CLIP for Point Cloud Understanding

by

Shuvozit Ghose

A thesis submitted to
The Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
July 2023

Thesis advisor

Yang Wang

Author

Shuvozit Ghose

CLIP for Point Cloud Understanding

Abstract

Contrastive Vision-Language Pre-training (CLIP) based point cloud classification model has added a new direction in the point cloud classification research domain. In this thesis, we propose two novel methods for CLIP-based point cloud classification. First, we propose a Pretrained Point Cloud to Image Translation Network (PPCITNet) that produces generalized colored images along with additional salient visual cues to the point cloud depth maps for CLIP based point cloud classification. In addition, we propose a novel viewpoint adapter that combines the view feature processed by each viewpoint as well as the global intertwined knowledge that exists across the multi-view features. Next, we propose a novel meta-episodic learning framework for CLIP-based point cloud classification. In addition, we introduce dynamic task sampling within the episode based on performance memory. The experimental results demonstrate the superior performance of the proposed model over existing state-of-the-art CLIP-based models on ModelNet10, ModelNet40, and ScanobjectNN datasets.

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Yang Wang, for his unwavering encouragement, continuous support, and invaluable mentorship throughout my master's research. His vast knowledge and insightful comments were instrumental in the completion of my thesis. It has been an honor to work under his guidance.

I am also profoundly grateful to Dr. Yiming Qian and Dr. Manyi Li for their invaluable guidance and advice during the research process. Their willingness to dedicate their time and expertise to our project was truly remarkable and greatly enhanced the quality of our work.

Furthermore, I would like to extend my sincere appreciation to Dr. Lorenzo Livi and Dr. Carson Kai-Sang Leung for their invaluable contributions as members of my thesis committee. Their constructive feedback and valuable suggestions played a pivotal role in refining my thesis to its highest standard.

I would like to express my heartfelt thanks to Dr. Yang Wang and the Faculty of Graduate Studies at the University of Manitoba for the financial assistance and unwavering support provided throughout my academic journey. I would like to extend my thanks to the support staff in the Department of Computer Science for their help.

Last, but not least, I am deeply grateful to my family for their love and encouragement. Their words of encouragement, understanding, and acts of kindness have meant the world to me, and I am truly grateful for their presence in my life.

This thesis is dedicated to my parents and my sister for their unconditional love and endless support.

Publications

Some of the ideas, materials, and figures in this thesis have appeared previously in the following submitted manuscripts:

- **Shuvozit Ghose**, Manyi Li, Yiming Qian, and Yang Wang. CLIP-based Few Shot Point Cloud Classification via Point Cloud to Image Translation. *British Machine Vision Conference* (BMVC), 2023. (Under review)

- **Shuvozit Ghose**, and Yang Wang. Meta Episodic Learning with Dynamic Task Sampling for CLIP-based Point Cloud Classification. *IEEE Winter Conference on Computer Vision* (WACV), 2024. (Under review)

Chapter 1

Introduction

Point cloud understanding refers to the process of extracting meaningful information from 3D point clouds, which are sets of 3D coordinates representing the surface geometry of objects or scenes. The goal of point cloud understanding is to analyze and interpret the data contained in the point cloud in order to understand the objects or scenes that it represents. Point cloud understanding has various applications in the real world, such as stereo reconstruction [31; 44], indoor navigation [61], autonomous driving [9; 3; 23], augmented reality and robotics perception [7; 10; 14] etc. Although both 2D image understanding and point cloud understanding involve analyzing visual data, compared to the 2D image understanding [38], 3D point cloud understanding [28; 54] is more challenging. A 2D image consists of a dense and regular pixel array. In contrast, a 3D point cloud only consists of sparse and unordered points in the 3D space. Moreover, point clouds often lack the rich texture and image information available in 2D images.

The success of deep learning in computer vision has also accelerated deep learning-

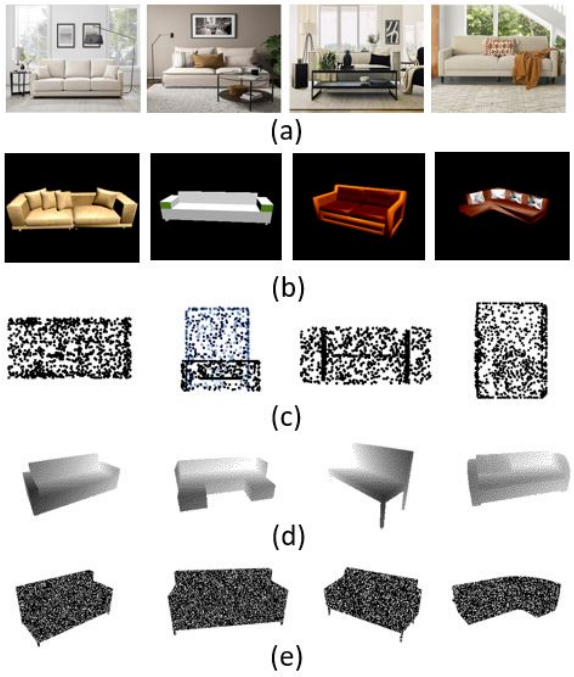

Figure 1.1: Example of different image representations: (a) natural RGB images; (b) rendered RGB images; (c) point cloud depth maps; (d) 3D depth maps; (e) processed binary mask images.

based point cloud understanding and 3D-related research. Early deep learning methods focus on designing advanced architectures for point clouds, such as PointNet [36], PointNet++ [37], RSCNN [24], DGCNN [51], CurveNet [53]. Recently, the success of Contrastive Language-Image Pre-training (CLIP) [38] model has added a new direction in the context of computer vision. Compared to the traditional deep learning models which require large amounts of labeled training data to achieve good performance on a specific task, CLIP [38] is trained in a more generalizable manner from unlabeled image-caption supervision and requires minimal additional training to adapt a specific downstream task. Recently following the CLIP's success on image [17; 13] and natural language domain [15], several works [59; 60; 16] have been proposed to generalize pre-trained CLIP to 3D recognition. Some of these works focus on

designing a small adapter network to CLIP [59; 16] and fine-tuning it for the downstream task. Other works focus on LLM-assisted 3D prompting and realistic shape projection [60] and cross-modal training framework [58] to bridge the gap between 2D image and point cloud. In general, the pipeline is as follows. Given a point cloud, the point cloud is first projected as a depth map. The depth map is then processed by the pre-trained CLIP visual encoder [38]. A small network called an adapter is added and fine-tuned for the downstream task.

Although these methods show some promising performance, they have certain limitations. CLIP [38] is trained on RGB images, whereas these models utilize point cloud depth maps for point cloud understanding. Inherently, RGB images and depth maps are quite different from one another as depicted in Fig. 1.1. Point cloud depth maps represent depth information which is useful for 3D reconstruction, robotics navigation, manipulation, etc. In contrast, RGB images consist of color channels that are important for tasks such as classification and recognition. The image information missing in the depth maps can lead to the degrading performance of the state-of-the-art CLIP [38] based point cloud models.

To transfer the image information to the CLIP [38] based point cloud models, one intuitive solution is to train a network that maps depth maps to the corresponding natural RGB images. During testing, we can use this network to convert a depth image to an RGB image as the input to CLIP. The challenge is that there does not exist any dataset that has depth maps and their corresponding natural RGB images. Alternatively, there exist datasets with depth maps and corresponding rendered RGB images. So it is possible to train a network that maps depth maps to the corresponding

rendered RGB images. However, directly using such a network does not work well due to several factors. Firstly, CLIP [38] is trained on natural RGB images. Rendered images differ from natural images in terms of realism, lighting, and Complexity as depicted by Fig. 1.1 (a,b). Secondly, for a single depth map, there can be many possible corresponding rendered RGB images. For example, for a depth map of a sofa, the synthetic color changes in various parts as depicted in Fig. 1.1 (b) in multiple rendered image instances. Finally, a 3D model depth map differs from a point cloud depth map as showed by [40; 19; 35]. A 3D model depth map typically represents depth information as a grayscale image shown in Fig. 1.1(d), with darker regions indicating greater distance from the viewer. In contrast, a point cloud depth map represents depth information as a set of 3D points in space, with each point having an (x, y, z) coordinate as depicted by Fig. 1.1(c).

In order to transfer image information to the CLIP [38] based point cloud model, we propose a novel Pretrained Point Cloud to Image Translation Network (PPCIT-Net) in this thesis that produces generalized colored images along with additional salient visual cues to the point cloud depth maps. Here, the salient visual cues refer to additional color concentration to prominent or distinctive parts like an additional color concentration in the head and legs of a person (see Fig. 3.3). The target of our PPCITNet is to provide image information to the CLIP [38] model so that it can achieve promising performance on point cloud classification and understanding. To pre-train this Point Cloud to Image Translation Network (PCITNet), we utilize the binary mask images of the rendered RGB images. Binary mask images and point cloud depth maps are similar geometrically because of discrete and compact represen-

tation. But visually, they are slightly different (see Fig. 1.1 (d, e)). To further bridge the gap, we preprocess the binary mask images by multiplying the binary image with a noise image to make the binary image sparse. The noise image is composed of 50% white pixel and 50% of black pixel sampled randomly. Through PPCITNet, the depth features of the point cloud can then be well aligned with the visual CLIP features. To further adapt our network to the few-shot learning, we propose a novel viewpoint adapter that combines the view feature processed by each viewpoint as well as the global intertwined knowledge that exists across the multi-view features.

Another issue of CLIP-based models is that although these methods show promising performance for some classes like airplanes, desks, guitars, etc, the performance for some classes like the cup, flower pot, sink, nightstand, etc is still far from satisfactory as depicted by Fig 4.1 (b). This is due to the fact that the adapter of CLIP-based models is trained using randomly sampled N-way K-shot data in the standard supervised learning setup. But, for some classes point cloud instances differ drastically in terms of shape and structure as depicted in Fig. 1.2. In the CLIP-based point cloud classification few-shot setup, the adapter tries to learn these various shape and structure variations through only k-shot data but often falls short, thus overfitting on the training set and generalizing poorly on evaluation because it is not possible for a single adapter model to perform well on every class-specific point cloud instance. However, another naive approach can be fine-tuning the adapter to the unseen class-specific point cloud instances. But it requires thousands of backpropagation gradients updates, thus requiring considerable time and computation to get the desired result. Moreover, although fine-tuning can solve the unseen class-specific point cloud adap-

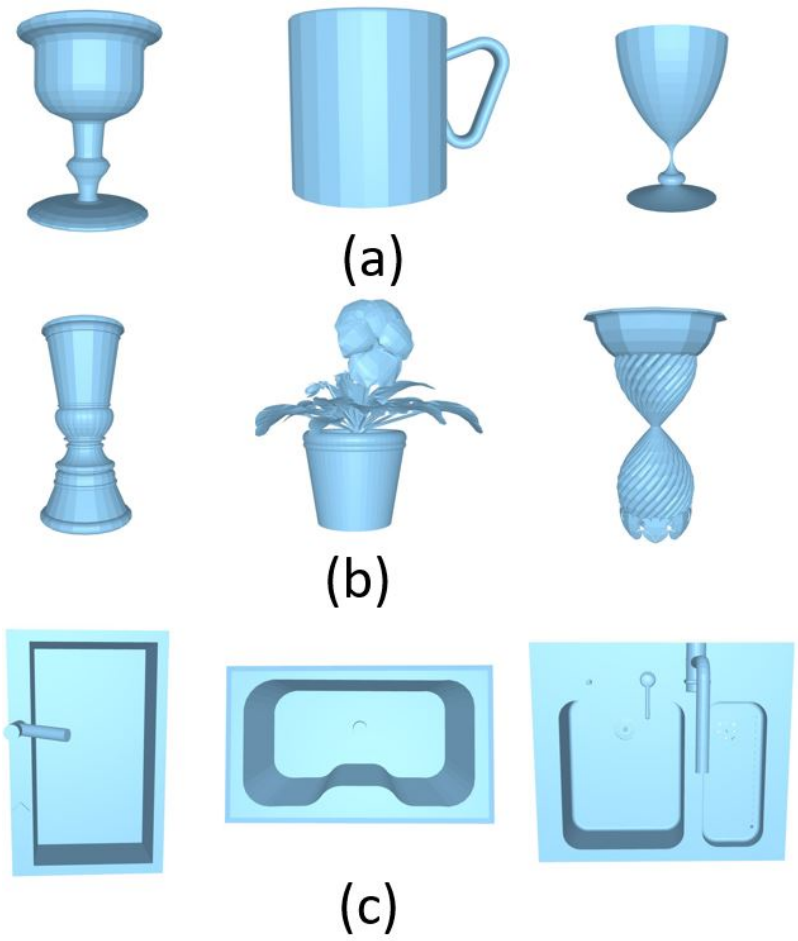

Figure 1.2: (a) cup (side view) (b) flower pot (side view) (c) sink (top view). We visualize the 3d object of the corresponding point cloud for better visual understanding.

tation problem to some extent, it is not possible to fine-tune the adapter with every unseen class-specific point cloud instance.

To solve this issue, one potential solution can be the introduction of a meta-learning formulation for CLIP-based point cloud classification. Unlike supervised learning, meta-learning learns novel tasks with only a few examples, in a similar way to human beings. Just like humans learn by seeing some examples and then using domain-specific knowledge in practical scenarios, meta-learning adapts to a

specific task by observing a few examples. The motivation of meta-learning is to absorb information from one task and generalize that information to unseen tasks proficiently by quickly adapting to the new tasks from a small set of training examples given during the testing phase. Especially, Model Agnostic Meta-learning (MAML) [12] has shown great impact. MAML [12] tries to find a good initialization across shared knowledge, so that small updates with task-specific data can adapt the model to that task, thus boosting performance.

However, the standard MAML [12] has several problems in this context. Firstly, the standard meta-learning paradigm is used to learn good initialization for the base model which has increased model capacity, enhanced feature extractor, and a large receptive field. In the case of CLIP based model, the adapter is a small model with a smaller receptive field. In the standard MAML [12], the base model learns the generalized knowledge from the randomly sampled task, whereas for CLIP-based models, some visual knowledge has already been captured by the CLIP Visual encoder as depicted by the Fig. 4.1 (a). Finally and most importantly, the standard meta-learning gives equal importance to all the classes and samples the N-way K-shot data accordingly. However, in the CLIP-based models, some visual information has already been learned by the CLIP visual encoder (e.g. airplanes, desks, guitars, etc), and the main target of our meta-training is to encode unknown generalized class information (e.g. cup, flower pot, sink, nightstand, etc) to the adapter.

For this purpose, we propose a novel meta-episodic learning framework with dynamic task sampling for CLIP-based point cloud classification in this thesis. While meta-episodic learning combines these two concepts by employing episodic training

within a meta-learning framework, dynamic task sampling ensures the sampling of unknown classes in the related task within an episodic. For sampling dynamic tasks within an episode, we propose a novel performance memory. The performance memory keeps track of the class-wise performance within an episode and samples the task according. In our approach, the CLIP-based model is trained on episodes of related tasks, where each episode contains a set of related examples mostly sampled from unknown classes. By learning from these episodes, the adapter acquires a generalization ability that enables it to quickly adapt and learn from new tasks.

1.1 Contributions

The contributions presented in this thesis are:

- We propose a novel Pretrained Point Cloud to Image Translation Network (PPCITNet) that transfers image information to the point cloud depth maps so that it can achieve promising performance on point cloud classification and understanding. In addition, we propose a novel viewpoint adapter that combines the view feature processed by each viewpoint as well as the global intertwined knowledge existing across the multi-view features.

- We, to the best of our knowledge for the first time, propose a novel meta-episodic learning framework for CLIP-based point cloud classification, addressing the challenges of limited training examples and sampling unknown classes. Additionally, we propose a novel dynamic task sampling technique within the episode based on performance memory. This sampling strategy effectively ad-

dresses the challenge of sampling unknown classes, ensuring that the model learns from a diverse range of classes and promotes the exploration of under-represented categories. By dynamically updating the performance memory, we adaptively prioritize the sampling of classes based on their performance, enhancing the model's ability to handle challenging and real-world scenarios.

1.2 Thesis Organization

We organize the remainder of this thesis as follows. First, in Chapter 2, we give an overview of the related works. More specifically, in this chapter, we discuss deep learning methods in point cloud classification, CLIP-based point cloud models, meta-learning methods, and sampling in few-shot learning. In Chapter 3, we outline our proposed point cloud to image translation network for CLIP-based point cloud classification. In Chapter 4, we propose our meta-episodic learning with dynamic task sampling for CLIP-based point cloud classification. Finally, in Chapter 5, we conclude this thesis.

Chapter 2

Related Work

In this chapter, we provide a brief overview of the works that are closely related to our proposed approaches.

2.1 Deep Learning in Point Cloud Classification

Deep learning has revolutionized the field of point cloud classification and understanding. Categorically, deep learning-based models are divided into three sections, including multi-view based methods [11; 46; 26], volumetric-based methods [21; 27; 52] and point-based methods [36; 37; 24; 53; 24]. Early works on deep learning primarily focused on multi-view-based methods [11; 46; 26], where the 2D image models are utilized for point cloud classification. In volumetric-based methods [21; 27; 52], point clouds are treated as voxel data, and 3D convolution-based models are used for classification and segmentation. The state-of-art models are point cloud-based methods [36; 37; 24; 53; 24], where the raw points are processed and passed

through the model without any transformation. PointNet [36] is the first point-based model that has encoded each point with a multi-layer perception. PointNet++ [37] further utilizes the max pooling operation to ensure permutation invariance. Recently, the success of CLIP for the downstream tasks on 2D has motivated the use of pre-trained CLIP for point cloud classification. Zhang et. al. [59] propose PointCLIP which generalizes pre-trained CLIP to 3D recognition.

2.2 CLIP-based Point Cloud Models

Recently several works [59; 60; 16; 58] have been proposed to generalize pre-trained Contrastive Language-Image Pre-Training (CLIP) to point cloud understanding tasks. For example, Zhang et. al. first proposed PointCLIP [59] by extending the CLIP [38] for handling 3D point cloud data. In addition, they presented an inter-view adapter to capture the feature interaction between multiple views. In this direction, Zhu et. al.[60] further introduced an efficient cross-modal adaptation method called PointCLIP V2 by proposing LLM-assisted 3D prompting and realistic shape projection. Next, Huang et. al. [16] presented a novel Dual-Path adapter and contrastive learning framework to transfer CLIP knowledge to the 3D domain. Yan et. al. [58] presented PointCMT, an point cloud cross-modal training framework that utilized the merits of color-aware 2D images and textures to acquire more discriminative point cloud representation and formulated point cloud analysis as a knowledge distillation problem.

2.3 Meta Learning

Meta-learning [45; 50; 12], also known as learning to learn, aim to acquire prior knowledge or meta-knowledge from the distribution of tasks and use this knowledge to facilitate learning on new, unseen tasks. Categorically, meta-learning is composed of three groups of methods. They are memory-based methods [42; 32], metric-based methods [45; 48], and optimization-based methods [12; 29; 47; 1]. While memory-based meta-learning [42; 32] algorithms store and leverage previous experiences in the form of memories or prototypes, metric-based meta-learning [45; 48] algorithms aim to learn a similarity metric or distance metric that allows for efficient generalization to new tasks. On the other hand, optimization-based meta-learning [12; 29; 47; 1] algorithms focus on learning an optimization procedure or policy that can quickly adapt model parameters to new tasks. These methods learn how to update the model's parameters in a task-specific manner, such that it can efficiently adapt to new tasks with minimal training iterations. However, the optimization-based models have gained prominence and achieved state-of-the-art performance in the meta-learning context because of their model-agnostic nature. In this direction, MAML [12] aims to learn a good initialization of model parameters that can be easily adapted to new tasks with a few gradient steps. The key idea behind MAML [12] is to optimize the model's parameters in a way that they can be fine-tuned quickly to minimize the loss of new tasks. MAML++ [1] is an extension of the original MAML [12] algorithm that introduces several improvements to enhance its performance. One key enhancement is the use of a second-order approximation for computing gradients, which takes into account the curvature of the loss landscape. By considering second-order information,

MAML++ [1] can capture more nuanced relationships between model parameters and achieve better adaptation to new tasks. MetaSGD [22] is another optimization-based meta-learning algorithm that addresses the challenges of learning an effective optimization algorithm itself. The main idea behind MetaSGD [22] is to learn a meta-optimizer that dynamically adjusts the learning rate or optimization strategy for different tasks. Instead of learning model parameters directly, MetaSGD [22] aims to learn the update rule or optimization algorithm that adapts the model's parameters.

2.4 Sampling in Fewshot Learning

Sampling plays a crucial role in few-shot learning by shaping the training and evaluation processes, enabling effective adaptation and generalization from limited labeled examples. Several works have been proposed in the last few years in this direction. For example, Pezeshkpour et. al. [34] investigated the effectiveness of active instance selection in the context of few-shot learning and proposed a framework where instances were selected based on their informativeness, relevance, and diversity. Arnold et.al. [2] highlighted the importance of considering episode difficulty and proposed a uniform sampling strategy to address this factor in few-shot learning. Le et. al. [20] introduced a method called Poodle that aimed to enhance few-shot learning performance by addressing the challenge of out-of-distribution samples. Xu et. al. [55] proposed a method for generating representative samples to improve the few-shot classification. Xu et. al. [56] addressed the issue of sample selection bias in few-shot learning and proposed a method to alleviate this bias by removing the

projection to the centroid. Roy et. al. [41] introduced a method called FeLMi that improved few-shot learning by incorporating a technique called hard mixup. The hard mixup technique involved selecting difficult samples from the support set and blending them with other samples from different classes.

Chapter 3

Pretrained Point Cloud to Image Translation Network

In this chapter, we first briefly revisit PointCLIP [59] for few-shot 3D classification (Sec. 3.1). Then we introduce our Pretrained Point Cloud to Image Translation Network (PPCITNet) framework (Sec.3.2) that aligns image information to the point cloud depth map. Next, we describe our proposed few-shot learning framework for few-shot point cloud classification (Sec. 3.3) and finally we provide an overview of the experimental setup (Sec. 3.4). The overall overview of our method is depicted in Fig. 3.1.

3.1 Revisit of PointCLIP

Similar to CLIP [38] which matches images and text by contrastive learning, PointCLIP [60] consists of one visual encoder and a textual encoder. For K class

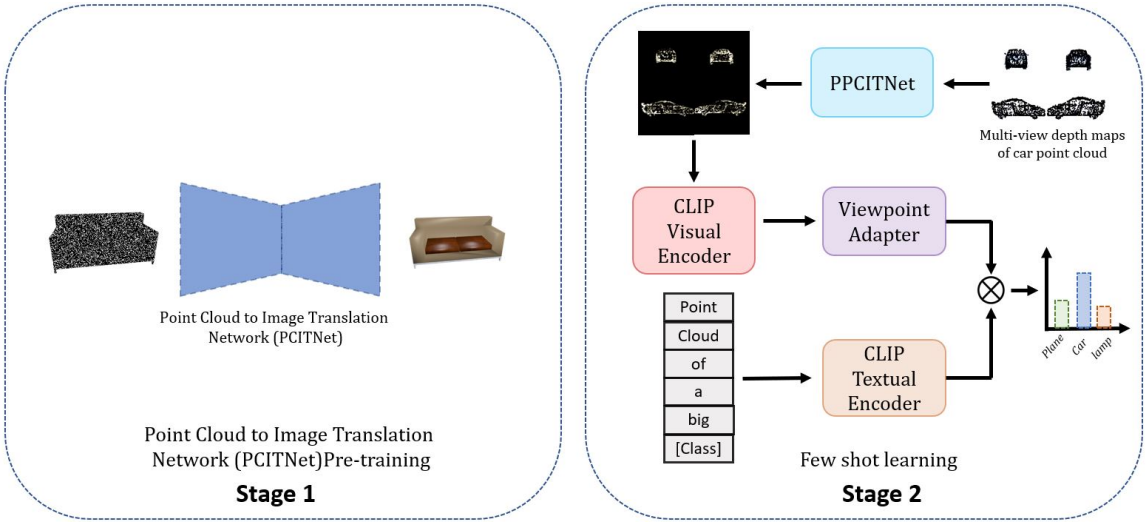

Figure 3.1: The training of our approach is composed of two stages. In the first stage, we pre-train our PCITNet using the processed binary mask and RGB pairs. In the second stage, we perform a few shot learning on a viewpoint adapter utilizing PPCITNet and pre-trained CLIP.

classification, PointCLIP uses a pre-defined template: "point cloud depth map of a [CLASS]" and the textual encoder outputs $P \in R^{K \times C}$, where C is the channel of the text embedding. To feed point clouds to the CLIP's visual encoder [38], point clouds are first projected onto depth maps $\{f_1, f_2, \ldots, f_M\}$. Here M denotes the number of views and $f_i \in W^{H \times W \times C}$ denotes each view of the point cloud, where H and W denote height and width respectively. Given the input $\{f_1, f_2, \ldots, f_M\}$, the visual encoder in PointCLIP generates visual feature $\{F_1^I, F_2^I, \ldots, F_M^I\}$, where $F_i^I \in R^{1 \times C}$ and C is the channel dimension of the embedding.

For few-shot point cloud classification, PointCLIP proposes an inter-view adapter. The inter-view adapter extracts the global visual representation by combining the multi-view features produced by the visual encoder of PointCLIP. The global representation is then added back to the adapted features F_i^I. Thus, the adapter can be

formulated as follows:

$$G = f_2(ReLU(f_1(concat(F_i^{I\,M}{}_{i=1})))) \tag{3.1}$$

$$F^g = ReLU(GW^T) \tag{3.2}$$

$$\widehat{y} = softmax(\sum_{i}^{M} \alpha_i((F_i^I + F^g)\{P^T\}^T)) \tag{3.3}$$

where P denotes textual information, α_i is a hyper-parameter that denotes importance of view i, W denotes learnable weights, and f_1, f_2 are MLP layers.

3.2 Point Cloud to Image Translation Network Pretraining

Instead of directly applying CLIP [38] visual encoder to depth maps, we propose to learn a Point Cloud to Image Translation Network (PCITNet) for aligning point cloud depth features with CLIP visual features. In other words, we expect the extracted features of a rendered point cloud depth map to be consistent with the CLIP visual features of the corresponding image. Then CLIP [38] textual prompts can be directly adopted to match the depth features. Let $S = \{B_i, R_i\}_{i=1}^{L}$ denotes a pretraining dataset with L instances. Here B_i is a binary mask image and R_i denotes its corresponding rendered RGB image. We would like to learn a network $F_\theta(\cdot)$ that maps from a binary mask image to a rendered RGB image as follows:

$$\widehat{R} = F_\theta(B) \tag{3.4}$$

Our goal is to learn the PCITNet F_θ that represents generalized image color distribution along with additional salient visual cues. As discussed earlier, binary

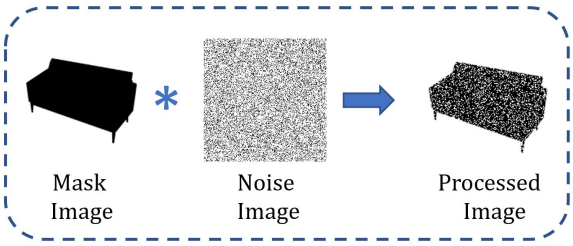

Figure 3.2: For a binary mask image, we multiply the binary image with a noise image to the make binary image sparse.

mask images and point cloud depth maps are similar geometrically because of their discrete and compact representations. But visually, they are slightly different. To further bridge the gap, we pre-process a binary mask image by multiplying the binary image with a noise image to make the binary image sparse as depicted in Fig. 3.2. The noise image is composed of 50% white pixel and 50% of black pixel sampled randomly. To learn the generalized image information along with additional salient visual cues, we optimize the following objective function:

$$\mathcal{L}_c = \frac{1}{L}\sum_{i=1}^{L}(R_i - \widehat{R}_i)^2 \tag{3.5}$$

Here, L is the total number of mask-RGB pairs in the dataset. The generalized image information along with additional salient visual cues information helps to encode a richer and more diverse set of visual features that can be used to discriminate between different objects. Without image information, CLIP [38] may have difficulty distinguishing between objects with similar shapes. For example, consider the task of classifying chairs based on their shape alone. Chairs have similar shape features to tables such as legs. Based on the shape alone, it is very difficult for CLIP [38] to distinguish between them. However, by incorporating image information in the classification process, we can identify additional features that can help differentiate

between chairs and tables as the image information provides additional cues for the CLIP [38] as described by Bramao et. al. [4].

3.3 Few-Shot Learning

Settings: Let $\rho \in R^{P\times 3}$ denote the point cloud, where P denotes the number of points of the point cloud sample from the $N \times K$ few shot data. Here, N is the total number of classes and each class has K instances of point cloud. Given the PPCITNet and pre-trained CLIP [38] network, the goal is to train the viewpoint adapter so that it can boost the performance of the CLIP-based point cloud classification network.

Feature Extraction: For each $\rho \in R^{P\times 3}$, we need to project 3D coordinates to 2D coordinates. Following [16], we get the point cloud depth maps $f^d = \{f_1, f_2, \ldots, f_M\}$. These depth maps are first passed through the PPCITNet, then the output feature is passed through CLIP's visual encoder. The goal of our PPCITNet is to provide generalized image information along with additional salient visual cues to the CLIP model so that it can achieve promising performance on point cloud classification and understanding.

$$f^c = F_\theta(f^d), i = 1, 2 \ldots M \tag{3.6}$$

$$f^v = F_V(f^c), i = 1, 2 \ldots M \tag{3.7}$$

where f_i^v denotes output for f_i^d depth map, F_θ and F_V denote the PPCITNet and CLIP's visual encoder [16] respectively.

Viewpoint adapter: We propose a novel viewpoint adapter that combines the view feature processed by each viewpoint as well as the global intertwined knowl-

edge that exists across the multi-view features. Given the extracted feature $f^v = \{f_1, f_2, \ldots, f_M\}$, the view-specific view information is calculated using M MLP layers. Thus,

$$f_i^l = \phi(f_i^v W_{li}) \tag{3.8}$$

where W_{li} is the weight of an MLP layer and ϕ denotes the activation function. f_i^l captures the view-specific fine-grained visual features and generalized image information along with additional salient visual cues that are relevant to a particular point cloud object. For example, to classify the point cloud of an airplane, the viewpoint that contains the wing information is more crucial than any other part. f_i^l encodes fine-grained wing information for the point cloud of the airplane. To get the global information of the M views, we perform the following operation:

$$f^g = \phi(concat(f_{i\ i=1}^{v\,M})W_{g1}^T)W_{g2}^T \tag{3.9}$$

where $f^g \in R^{1\times C}$ and W_{g1} W_{g2} denote the two-layer weights in the viewpoint adapter. Here, the global knowledge captures the overall structure and organization of point clouds and provides a more holistic understanding of the point cloud objects. Finally, the classification is performed as follows:

$$logits = softmax(\sum_{i}^{N} \alpha_i((f_i^l + f^g)\{P^T\}^T)) \tag{3.10}$$

Where α_i denotes the learnable weight, and P denotes textual information. Note that, Only the viewpoint adapter is trained in few-shot learning. The features learned by the viewpoint adapter provide complementary information about the overall structure and view-specific fine-grained features of point cloud objects combining both view and global information.

3.4 Experiments

3.4.1 Pre-training Datasets

To pre-train our PCITNet network, we use the DISN 2D dataset released by Wang et.al. [57]. This dataset is based on the ShapeNet Core dataset [5], which is a 3D dataset consisting of 13 object categories. While early work [8] of rendering this dataset utilizes 24 views with limited variation in terms of camera orientation for each model, DISN provides two types of settings: "easy" and "hard". The easy setting consists of 36 renderings with smaller variations, The hard setting is composed of 36 renderings with larger variations. To train our PCITNet network, we sample 100k data from the easy setting randomly. From the RGBA image, we sample the mask image.

Model	ModelNet10	ModelNet40	ScanObjectNN
CurveNet	82.45	76.55	34.76
SimpleView	84.15	71.17	37.44
PointNet	73.98	67.34	36.18
PointNet++	84.62	77.13	51.62
PointCLIP	89.33	83.80	54.37
CLIP2Point	90.21	85.10	57.49
PPCITNet	**94.30**	**88.93**	**63.22**

Table 3.1: Performance (%) of PPCITNet with other methods in 16-shot setup using prompt "point cloud of a big [CLASS]".

3.4.2 Downstream Datasets

Following PointCLIP [59], we evaluate our proposed model on three widely used benchmark datasets: ModelNet10 [52], ModelNet40 [52] and ScanObjectNN [49].

ModelNet10 and ModelNet40 have a training point cloud set of 3991 and 9,843 and a test point cloud set of 908 and 2,468 respectively. ScanObjectNN is a real-world point cloud dataset that includes 2,321 samples for training and 581 samples for testing the point cloud from 15 categories. Compare to the ModelNet, ScanObjectNN is more challenging because the CAD models are attached with backgrounds and partially presented. For all three datasets, we uniformly sample 1,024 points of each object as the PPCITNet's input.

3.4.3 Implementation Details

We use Unet architecture from [39] as our PCITNet network. To pre-train the PCITNet network, we resize the image to 224 x 224 and train our model in a 12 GB Nvidia Titan X GPU using PyTorch. In pre-training, we use the Adam optimizer [18] with decay of $1x10^{-4}$ and the initial learning rate of $1x10^{-3}$. Our pre-training task takes 100 epochs with a batch size of 16. For few-shot learning, we utilize AdamW optimizer [25] with decay of $1x10^{-4}$ and the initial learning rate of $1x10^{-3}$. The training batch size is 32 and it takes 100 epochs to train the network. Similar to [59; 16], we use the 6 orthogonal views: left, right, top, bottom, front, and back for few-shot learning.

3.4.4 Experimental Results

CLIP-based models [59; 16] are generally evaluated by comparing with state-of-the-art methods on few-shot learning and prompt engineering. In Table 3.4.1, we present the few shot performance of PPCITNet and compare it with state-of-the-art

Prompts	PointCLIP	PPCITNet
"a photo of a [CLASS]."	81.78	**86.63**
"a point cloud photo of a [CLASS]."	82.02	**87.33**
"point cloud of a [CLASS]."	82.10	**87.04**
"point cloud of a big [CLASS]."	83.80	**88.93**
"point cloud depth map of a [CLASS]."	81.58	**85.15**
"[Learnable Tokens] + [CLASS]"	69.23	**76.27**

Table 3.2: Performance (%) of PPCITNet with PointCLIP for different prompt designs on ModelNet40.

3D networks like PointNet [36], PointNet++ [37], CurveNet [53], SimpleView [6] as well as CLIP based models PointCLIP [59], CLIP2Point [16] on 16 shot setup. As we can see from the table, PPCITNet with a viewpoint adapter outperforms PointCLIP and CLIP2Point by a margin of 3-5 % for 16 shot setup for prompt "point cloud of a big [CLASS]" on all three datasets. To further evaluate the transfer ability of PPCITNet, we show the performance for 1,2,4,8,10,12,16 shots in Fig. 3.4.

We can see from the graph, our PPCITNet surpasses all by a reasonable good margin. This is due to the additional visual cues provided by PPCITNet and the view and global information encoding of the viewpoint adapter. The large performance gain on ScanObjectNN indicates the robustness of PPCITNet under noisy real-world scenes. The visualization in Fig. 3.3 further establishes our claims. While PointCLIP and CLIP2Point provide uniformly sampled point features to the CLIP's visual encoder, our PPCITNet produces generalized colored images along with additional salient visual cues. Here, the salient visual cues refer to additional color concentration to prominent or distinctive parts like an additional color concentration in the head and legs of the human in Fig. 3.3. In Table 3.4.3, we compare our PPCITNet with

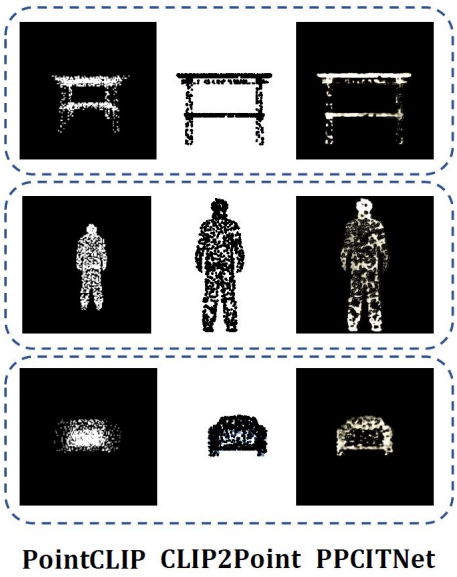

Figure 3.3: Input Visualization. Our PPCITNet produces generalized colored images along with additional salient visual cues.

PointCLIP for different prompt designs on ModelNet40, where [CLASS] represents the class token and '[Learnable Tokens]' refers to the prompts with a fixed length that are capable of being learned during training. The large performance gain indicates the generability of our PPCITNet over PointCLIP for various prompt designs.

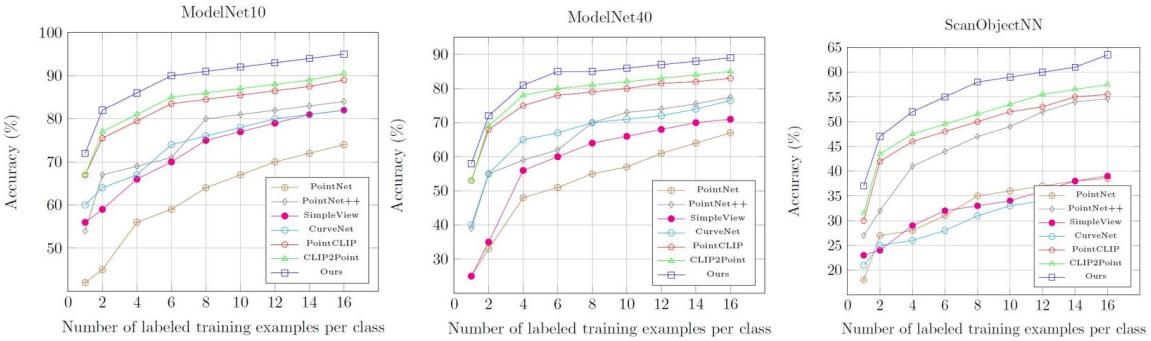

Figure 3.4: Few-shot performance comparison under 1, 2, 4, 8,10,12,14, and 16-shot settings.

3.4.5 Ablation Studies

In this section, we evaluate the effect of our PPCITNet and the effect of view information on the viewpoint adapter. To observe the effect of PPCITNet, we conduct an experiment with PPCITNet and without PPCITNet on ModelNet40 as shown in Fig. 3.5.

Model	Accuracy
Without PPCITNet	84.27
With PPCITNet	**88.93**

Figure 3.5: Effect of PPCITNet on ModelNet40

View info.	Global info.	Accuracy
✓	-	82.34
-	✓	87.60
✓	✓	**88.93**

Figure 3.6: Effect of view information for PPCITNet on ModelNet40

From the table, it is evident that incorporating PPCITNet on the few-shot pipeline improves accuracy by 4.6 %. To analyze the view feature, we conduct experiments with the only view feature, with only global information, and with both view information and global information on PPCITNet on ModelNet40. Although the performance drops significantly while utilizing only the view information, a combination of view and global information yields the best performance, specifically an improvement of 1.3% over global information as described in Fig. 3.6.

Chapter 4

Meta Episodic Learning with Dynamic Task Sampling

In this Chapter, we first briefly describe baseline CLIP-based point cloud models like PointCLIP [59], CLIP2Point [16] for point cloud classification (Sec. 4.1). Then we introduce our meta-episodic learning framework for point cloud classification (Sec.4.2). Finally, we describe our proposed dynamic task sampling technique based on performance memory for class instance adaptive point cloud classification (Sec. 4.3). Finally, we provide an overview of the experimental setup (Sec. 4.4). The overall overview of our method is depicted in Fig. 4.

4.1 Baseline CLIP-based Point Cloud Models

The CLIP-based point cloud models are the state-of-the-art point cloud classification methods in the few-shot setup. In this thesis, we select two CLIP-based

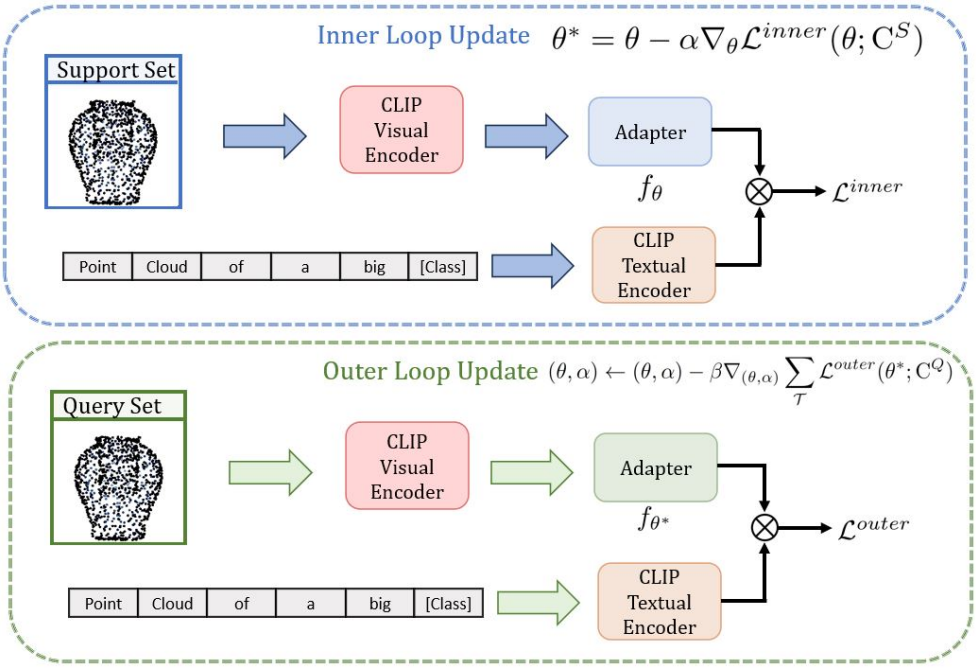

Figure 4.1: The adapter learns following a bi-level optimization process. While the adapter learns to recognize and discriminate the features of the inner loop update, the outer loop extracts meta-features of the point clouds that generalize across tasks. Additionally, we introduce dynamic task sampling within the episode based on performance memory to ensure underrepresented class sampling.

point cloud methods, PointCLIP [60] and CLIP2Point [16], to use as our baseline CLIP-based point cloud models.

For completeness, we briefly summarize the outline of the CLIP-based point cloud models. In general, they consist of three components: (a) a pre-trained CLIP visual Encoder, F_V (b) a pre-trained CLIP textual Encoder F_T (c) a small trainable network called an adapter F_θ. For N class classification, CLIP-based models use a pre-defined template: "point cloud depth map of a [CLASS]" and the textual encoder outputs $W_t \in R^{N\times C}$, where C is the channel of the text embedding. To feed the point cloud to the CLIP's visual encoder, point clouds are first projected into $\{f_1, f_2, \ldots, m\}$ depth

maps where m denotes the number of views and $f_i \in W^{1\times C}$ denotes each view of the point cloud. Given the input $\{f_1, f_2, \ldots, f_m\}$, the CLIP's visual encoder generates visual feature $\{F_1^I, F_2^I, \ldots, F_m^I\}$, where $F_i^I \in R^{1\times C}$, C being the channel information of the embedding. In the zero-shot setup, there is no training stage and no adapter is required. Each viewpoint generates a prediction by calculating the cosine similarity between the visual feature F_i^I and the textual feature F^T. The final prediction is a weighted sum of all viewpoint-wise predictions.

For few-shot point cloud classification, the adapter F_θ is trained using randomly sampled N-way K-shot data from the training dataset in the standard supervised learning setup. For this purpose, PointCLIP [59] proposes an interview adapter that extracts a global visual representation by combining multi-view features generated by the CLIP visual encoder. The global representation obtained from the inter-view adapter serves as a higher-level understanding of the visual input and encapsulates important information that is shared across different views. During inference, CLIP's visual feature $\{F_1^I, F_2^I, \ldots, F_m^I\}$ is passed through the trained adapter network F_θ and the final prediction is calculated as cosine similarity between the output adapter feature and the textual feature.

On the other hand, CLIP2Point [16] proposes a Dual-Path Adapter (DPA) that combines the benefits of pre-training with instance-level depth maps and category-level discrimination from CLIP pre-training knowledge. The DPA module consists of a dual-path structure, utilizing two encoders: the pre-trained depth encoder and the CLIP visual encoder. The combination of the two encoders allows for a comprehensive representation of the visual data, incorporating both instance-level depth information

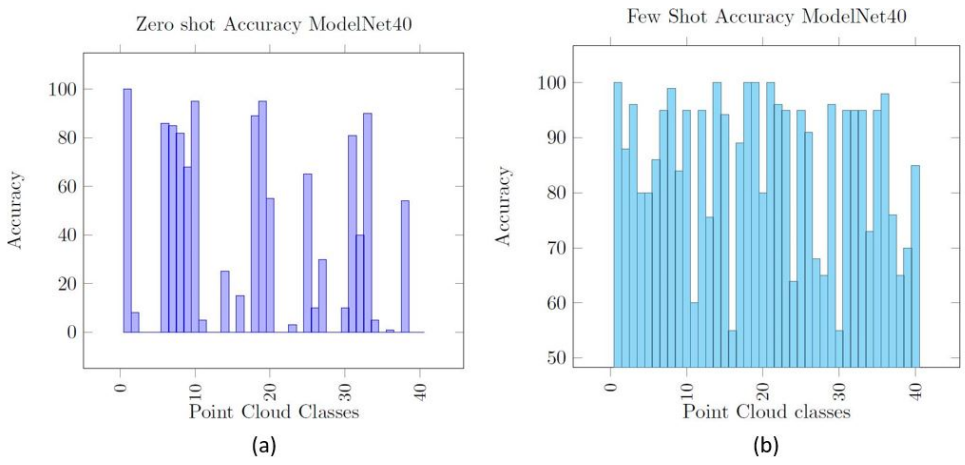

Figure 4.2: (a) Zero-shot result of PointCLIP [59]. We can see that CLIP's Visual encoder has already captured some classes like airplanes, desks, guitars, etc. (b) Few-shot result of PointCLIP [59]. Some classes like the cup, flower pot, sink, nightstand, etc perform poorly in the few-shot setup.

and category-level discrimination from CLIP pre-training. By combining the outputs of the two encoders, the DPA module generates the final logits. We refer the reader to [60; 16] for further architectural details. In this thesis, the adapter indicates the interview adapter for PointCLIP and the dual-path adapter for CLIP2Point and denotes the complete set of parameters as θ.

4.2 Meta Episodic Learning for Point Cloud Classification

Traditionally, the CLIP-based point cloud model inputs a point cloud ρ and generates its corresponding class label Y. Conventional CLIP-based point cloud models learn from N way K shot data, where N denotes the total number of classes in the dataset and K notes the total number of instances per class. Due to the data

instance-specific training, it ignores the class-specific shape and structure variation distribution. Henceforth, the performance deteriorates on some point cloud classes because of poor generalization on diverse point cloud class instances.

In contrast, we take a meta-episodic learning approach for the few-shot learning of the adapter along with dynamic tasks to address the challenge of CLIP-based point cloud classification. It combines the concepts of meta-learning and episodic training to enable efficient adaptation to new tasks with limited training examples. The goal is to encode unknown generalized class information into the adapter, allowing it to achieve satisfactory results. Let's consider W_{tr} and W_{test} denote the disjoint training and testing point cloud set respectively, i.e., $W_{tr} \cap W_{test} = \phi$. The training and testing set are denoted as $\mathcal{C}^{tr} = \{\mathcal{C}_1^{tr}, \mathcal{C}_2^{tr}, \ldots, \mathcal{C}_{|W_{tr}|}^{tr}\}$ and $\mathcal{C}^{test} = \{\mathcal{C}_1^{test}, \mathcal{C}_2^{test}, \ldots, \mathcal{C}_{|W_{test}|}^{test}\}$. Each i-th point cloud class in the training set has its own total of M labeled point cloud instances as $\mathcal{C}_i = \{(\rho_1, Y), (\rho_2, Y), \cdots, (\rho_M, Y)\}$. In the meta-episodic training, let each episode consists of S tasks, where each $\mathcal{T}$ consists of a support set C^S and a query set C^Q. Intuitively, our approach tries to find a good initialization of parameters θ, representing the prior or generalized knowledge of point clouds, so that a few updates of θ using C^S can make large improvements by reducing the error measures and boosting the performance in C^Q. To learn this optimal initialisation parameter θ, we first adapt (task-specific) f_θ using C^S by fine-tuning:

$$\theta^* = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{C}^S) \tag{4.1}$$

Where $\mathcal{L}^{inner}$ denotes loss in the inner loop and α denotes inner loop learning rate. In the inner loop update, the adapter learns to recognize and discriminate the patterns, features, and characteristics relevant to the point cloud classification task.

Evaluation of the adapted model is performed on unseen examples sampled from the same task C^Q, to measure the generalization of f_{θ^*}. This acts as feedback for MAML [12] to adjust its initialization parameters θ to achieve better generalization on any $\mathcal{T}$ (across-task):

$$(\theta, \alpha) \leftarrow (\theta, \alpha) - \beta \nabla_{(\theta,\alpha)} \sum_{\mathcal{T}} \mathcal{L}^{outer}(\theta^*; \mathrm{C}^Q) \tag{4.2}$$

Where $\mathcal{L}^{outer}$ denotes loss in the outer loop and β denotes outer loop learning rate. The primary objective of the outer loop is to extract meta-knowledge or meta-features that generalize across tasks. However, the standard meta-learning has several limitations for the CLIP-based point cloud classification. To address these challenges, we propose novel dynamic sampling within the episode based on performance memory.

4.3 Dynamic Task Sampling

The standard Model Agnostic Meta-learning (MAML) [12] is not well-suited for CLIP-based models due to several reasons. Firstly, the standard meta-learning paradigm is used to learn good initialization for the base model which has increased model capacity, enhanced feature extractor, and a large receptive field. In the case of CLIP based model, the adapter is a small model with a smaller receptive field. In the standard MAML [12], the base model learns the generalized knowledge from the randomly sampled task, whereas for CLIP-based models, some visual knowledge has already been captured by the CLIP visual encoder as depicted by the Fig. 4.1 (a). Finally and most importantly, the standard meta-learning gives equal importance to all the classes and samples the N-way K-shot data accordingly. However, in the CLIP-based models, some visual information has already been learned by the CLIP

visual encoder (e.g. airplanes, desks, guitars, etc), and the main target of our meta-training is to encode unknown generalized class information (e.g. cup, flower pot, sink, nightstand, etc) to the adapter.

Algorithm 1 Training of our approach

Input: Training dataset $\mathcal{C}^{tr} = \{\mathcal{C}_1^{tr}, \mathcal{C}_2^{tr}, \ldots, \mathcal{C}_{|W_{tr}|}^{tr}\}$; β as learning rate.
Requires: Pre-trained CLIP visual encoder F_V, Pre-trained CLIP textual encoder F_T
Initialise: Initialise adapter θ, inner learning rate α
Output: Optimised meta-parameters $\{\theta, \alpha\}$
for each episode **do**
  Initialize Performance Memory P with keys as the class labels and values as 0
  for $task = 1, 2, \ldots, S$ **do**
    Sort P based on values and decide classes to sample
    Sample task $\mathcal{T} = \{\mathrm{C}^S, \mathrm{C}^Q\}$ from the decided classes
    for each task $\mathcal{T}$ **do**
      Evaluate inner objective: $\mathcal{L}^{inner}(\theta; \mathrm{C}^S)$
      Adapt: $\theta^* = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{C}^S)$
      Compute outer objective: $\mathcal{L}^{outer}(\theta^*; \mathrm{C}^Q)$
    end for
    Update meta-parameters: $(\theta, \alpha) \leftarrow (\theta, \alpha) - \beta \nabla_{(\theta,\alpha)} \sum_{\mathcal{T}} \mathcal{L}^{outer}(\theta^*; \mathrm{C}^Q)$
    Update the class-wise accuracy in P
  end for
end for

To address these challenges, we propose novel dynamic sampling within the episode based on performance memory. Dynamic task sampling is a crucial component of the proposed meta-episodic learning framework for CLIP-based point cloud classification and performance memory plays a critical role in guiding the task sampling process. It aims to address the challenge of effectively sampling unknown classes within episodes, allowing the model to learn from diverse and relevant data. Let P denote performance memory and $P[i]$ denote $i-th$ class in the performance memory. At the onset of each episode, we initialize all the $P[i]$ equal to 0. For task sampling within an episode, we employ a specific procedure. Initially, we sort the performance memory in ascending order based on the recorded values. This sorting enables us to prioritize the classes based on their performance. Consequently, we select the first N classes from the performance memory, facilitating an N-way task sampling. Throughout the episode, the performance memory is updated dynamically as follows:

$$P[i] = \frac{P[i] + A_i}{2} \tag{4.3}$$

Where A_i denotes $i-th$ class accuracy in the query set C^{Q} of a task. This updating process allows the performance memory to reflect the evolving performance of each class within the episode. By incorporating dynamic sampling within the episode based on performance memory, our proposed approach ensures that the model is exposed to a diverse range of classes during training. It prioritizes the sampling of classes that have performed relatively poorly, promoting the model's ability to learn from challenging and underrepresented classes.

In summary, our meta-episodic learning framework leverages dynamic sampling

Algorithm 2 Inference of our model

Input: Testing dataset $\mathcal{C}^{test} = \{\mathcal{C}_1^{test}, \mathcal{C}_2^{test}, \ldots, \mathcal{C}_{|W_{test}|}^{test}\}$; number of gradient updates n.

Requires: Pre-trained CLIP visual encoder F_V, Pre-trained CLIP textual encoder F_T, Optimised meta-parameters $\{\theta, \alpha\}$

Sample task $\mathcal{T} = \{\mathrm{C}^S, \mathrm{C}^Q\}$ from $\mathcal{C}^{test}$

for n steps **do**

    Evaluate inner objective: $\mathcal{L}^{inner}(\theta; \mathrm{C}^S)$

    Adapt: $\theta_c = \theta - \alpha \nabla_\theta \mathcal{L}^{inner}(\theta; \mathrm{C}^S)$

end for

Return class instance specialised Adapter params. $\theta_{\mathbf{c}}$.

within the episode, guided by the performance memory. This approach effectively addresses the challenge of sampling unknown classes, enabling the model to learn from diverse and relevant data. The updating of the performance memory ensures that the model adapts to the evolving performance of different classes, enhancing its ability to handle real-world scenarios and achieve improved performance in CLIP-based point cloud classification tasks. The training and inference process of our approach is summarised in Algorithm 1 and 2, respectively.

4.4 Experiments

4.4.1 Datasets

We evaluate our proposed approach on two widely used benchmark datasets ModelNet40 [52] and ScanObjectNN [49]. ModelNet40 is a synthetic indoor 3D dataset that consists of 40 classes of point cloud objects. It has a training set of 9,843 and a testing set of 2,468 point clouds respectively. As the original ModelNet40 is not aligned in orientation, in our approach we use the aligned version of ModelNet 40 [43]. ScanObjectNN is a real-world point cloud dataset that has a training set of 2,321 and a testing set of 581 samples from 15 point cloud classes. Following PointCLIP [59] and CLIP2Point [16], we train our approach on the ModelNet40 training set and test on ModelNet40 and ScanObjectNN testing set.

4.4.2 Implementation Details

We implemented our framework in PyTorch [33] and conducted experiments on a 12 GB Nvidia Titan X GPU. We train the adapter from scratch with 100 epochs, and each epoch contains 50 meta-training episodes according to algorithm 1. Each episode consists of 20 tasks and each task 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. In our approach, we use a 3-way 5-shot 5-query for our tasks. Once the meta-training is ended, we test the network according to algorithm 2. For meta-training and meta-testing, we used a one-step gradient update. We use ADAM as a meta-optimizer with outer-loop learning rate β as 0.0001, while the inner-loop

learning α is meta-learned during training. We use accuracy as our evaluation metric.

4.4.3 Competitors

To the best of our knowledge, no prior research has specifically addressed task-specific few-shot learning for CLIP-based Point cloud models. However, in order to validate our approach, we have designed four strong baselines from the perspective of optimization-based meta-learning. These baselines are as follows: **(i) MAML [12]:** In this baseline, we train CLIP-based models using standard random task sampling, without employing episodes. **(ii) Reptile [30]:** This baseline involves directly performing inner loop updates using a randomly sampled support set of tasks. The models converge to initialization by accumulating changes made during these updates across multiple tasks. **(iii) MetaSGD [22]:** Here, CLIP-based models are trained within a meta-episodic learning framework along with random task sampling, where each parameter has its own learning rate. **(iv) Ours:** We train CLIP-based models using a MAML approach within a meta-episodic learning framework along with dynamic task sampling.

Methods	**ModelNet40**		**ScanobjectNN**	
	Baseline	**Ours**	**Baseline**	**Ours**
PointCLIP	83.80	**86.93**	54.37	**58.72**
CLIP2Point	85.10	**88.64**	57.49	**63.65**

Table 4.1: Comparison (%) of baseline and our approach on PointCLIP and CLIP2Point using prompt "point cloud of a big [CLASS]".

4.4.4 Result Analysis and Discussion

Table 4.1 compares the baseline results with our proposed approach on PointCLIP [59] and CLIP2Point [16] models using the prompt "point cloud of a big [CLASS]." For ModelNet40, our approach has outperformed the baseline model by 3.13% and 3.54% on PointCLIP [59] and CLIP2Point [16] respectively. Similarly, on the ScanobjectNN dataset, the performance improves by 4.35% and 6.16% on PointCLIP [59] and CLIP2Point [16] respectively. This result indicates the robustness of our meta-episodic approach compared to the standard few-shot approach used by CLIP-based point cloud models.

Methods	**ModelNet40**	**ScanobjectNN**
PointCLIP Baseline	83.80	54.37
PointCLIP + MAML	84.78	55.69
PointCLIP + Reptile	84.10	55.04
PointCLIP + MetaSGD	84.92	55.85
PointCLIP + Ours	**86.93**	**58.72**
CLIP2Point Baseline	85.10	57.49
CLIP2Point + MAML	86.34	58.56
CLIP2Point + Reptile	85.83	58.24
CLIP2Point + MetaSGD	86.36	58.68
CLIP2Point + Ours	**88.64**	**63.65**

Table 4.2: Performance (%) analysis with different Baselines on ModelNet40 and ScanobjectNN using prompt "point cloud of a big [CLASS]".

Table 4.2 presents a performance analysis with different Competitors on ModelNet40 and ScanobjectNN datasets. The Competitors include PointCLIP [59] and CLIP2Point [16] with various meta-learning algorithms such as MAML, Reptile, and MetaSGD, as well as our proposed approach. For PointCLIP [59], our proposed approach outperforms MAML by 2.15% and 3.03% on the ModelNet40 and ScanobjectNN datasets, respectively. Similarly, our approach surpasses Reptile by 2.83% and

3.68%, and MetaSGD by 2.01% and 2.87% on the same datasets. For CLIP2Point [16], our approach demonstrates superior performance compared to the baselines. It outperforms MAML by 2.3% and 5.09% on the ModelNet40 and ScanobjectNN datasets, respectively. Moreover, our model surpasses Reptile by 2.81% and 5.41%, and MetaSGD by 2.28% and 4.97% on the respective datasets. These results highlight the effectiveness of our proposed approach in achieving better performance compared to the other competitors in the task-specific few-shot learning setting for CLIP-based point cloud models.

Prompts	PointCLIP Baseline	PointCLIP + Ours
"a photo of a [CLASS]."	81.78	**84.55**
"a point cloud photo of a [CLASS]."	82.02	**86.12**
"point cloud of a [CLASS]."	82.10	**86.82**
"point cloud of a big [CLASS]."	83.80	**86.93**
"point cloud depth map of a [CLASS]."	81.58	**85.97**

Table 4.3: Comparison (%) of Baseline and our approach on PointCLIP for different prompt designs on ModelNet40.

Table 4.4.4 focuses on different prompt designs for the PointCLIP [59] model on the ModelNet40 dataset. It compares the baseline accuracy with our approach's accuracy for various prompt designs. Our approach consistently outperforms the baseline in all prompt designs.

4.4.5 Ablation Study

The Effect of Dynamic Task Sampling: To observe the effect of dynamic task sampling, we conduct an experiment with dynamic task sampling and random task

sampling using the prompt "point cloud of a big [CLASS]" on ModelNet40 as shown in Tab. 4.4. From the table, it is evident that incorporating dynamic task sampling on the few-shot pipeline improves accuracy by 2.15 %. and 2.3 % for PointCLIP [60] and CLIP2Point [16] respectively. This is because dynamic task sampling prioritizes the sampling of classes that have performed relatively poorly, promoting the model's ability to learn from challenging and underrepresented classes.

	PointCLIP	CLIP2Point
Random task sampling	84.78	86.34
Dynamic task sampling	**86.93**	**88.64**

Table 4.4: Effect of dynamic task sampling on ModelNet40 using prompt "point cloud of a big [CLASS]".

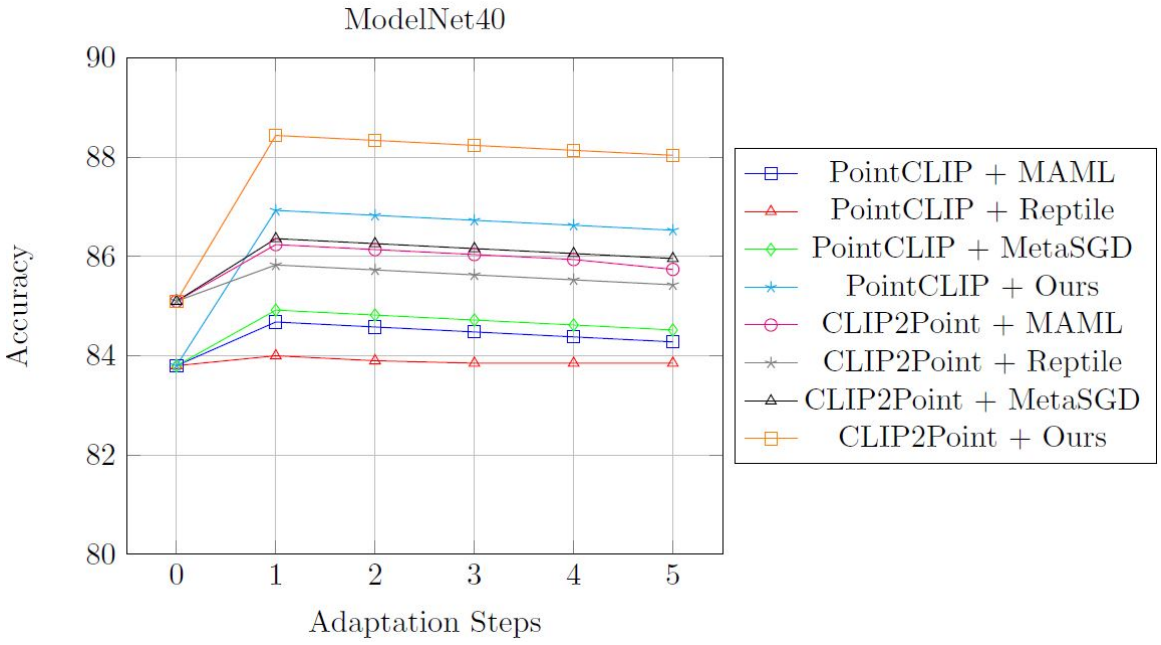

Figure 4.3: Accuracy of competitors with varying adaptation steps on ModelNet40 using prompt "point cloud of a big [CLASS]".

Number of Adaptation Steps: In Fig. 4.3, we experiment with varying the number of adaptation steps during the inference process. we observe that using just a

single gradient step update, which is the most common approach in our experiments, yields the highest performance gain. The reason behind this result could be that the inner loop, which handles the adaptation process, focuses too much on capturing unnecessary point cloud style details, causing it to forget the generic prior knowledge learned by the model. As a result, performing additional updates in the inner loop might lead to diminishing results, as the model becomes overfit to the specific style of the training examples and loses its ability to generalize well to new tasks.

Chapter 5

Conclusion and Future Work

In this thesis, we have presented effective deep-learning approaches for CLIP-based point cloud classification by proposing two novel methods. First, we present a novel pretrained point cloud to image translation network that transfers image information to the point cloud depth maps. In addition, we present a novel viewpoint adapter that combines the view feature processed by each viewpoint as well as the global intertwined knowledge existing across the multi-view features. Second, we have proposed a novel meta-episodic learning framework for CLIP-based point cloud classification, addressing the challenges of limited training examples and sampling unknown classes. Our framework combines meta-learning and episodic training, enabling the model to quickly adapt and generalize to new tasks. Additionally, we have introduced dynamic task sampling within the episode based on performance memory. This sampling strategy effectively addresses the challenge of sampling unknown classes, ensuring that the model learns from a diverse range of classes and promotes the exploration of underrepresented categories. By dynamically updating

the performance memory, we adaptively prioritize the sampling of classes based on their performance, enhancing the model's ability to handle challenging and real-world scenarios.

In terms of future direction, we would like to explore the application of advanced natural language processing techniques to identify the most effective prompts for enhancing the performance of CLIP-based point cloud classification. In this direction, our target is to move beyond fixed templates and embrace more descriptive prompts for point cloud classification. By employing descriptive prompts, we expect the model to gain a deeper understanding of the intricacies and attributes within point clouds, leading to more accurate and context-aware classifications.

Bibliography

[1] A. Antoniou, H. Edwards, and A. Storkey. How to train your maml. *ICLR*, 2019.

[2] S. Arnold, G. Dhillon, A. Ravichandran, and S. Soatto. Uniform sampling over episode difficulty. *Advances in Neural Information Processing Systems*, 34:1481–1493, 2021.

[3] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9297–9307, 2019.

[4] I. Bramão, A. Reis, K. M. Petersson, and L. Faísca. The role of color information on object recognition: A review and meta-analysis. *Acta psychologica*, 138(1):244–253, 2011.

[5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[6] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for

contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] C. Choy, J. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[8] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 628–644. Springer, 2016.

[9] Y. Cui, R. Chen, W. Chu, L. Chen, D. Tian, Y. Li, and D. Cao. Deep learning for image and point cloud fusion in autonomous driving: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):722–739, 2021.

[10] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[11] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. 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.

[12] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*, 2017.

[13] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, and Y. Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

[14] B. Graham, M. Engelcke, and L. Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018.

[15] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. *ACL*, 2020.

[16] T. Huang, B. Dong, Y. Yang, X. Huang, R. W. Lau, W. Ouyang, and W. Zuo. Clip2point: Transfer clip to point cloud classification with image-depth pre-training. *arXiv preprint arXiv:2210.01055*, 2022.

[17] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[18] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[19] E. Lachat, H. Macher, T. Landes, and P. Grussenmeyer. Assessment and calibration of a rgb-d camera (kinect v2 sensor) towards a potential use for close-range 3d modeling. *Remote Sensing*, 7(10):13070–13097, 2015.

[20] D. Le, K. D. Nguyen, K. Nguyen, Q.-H. Tran, R. Nguyen, and B.-S. Hua. Poodle: Improving few-shot learning via penalizing out-of-distribution samples. *Advances in Neural Information Processing Systems*, 34:23942–23955, 2021.

[21] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas. Fpnn: Field probing neural networks for 3d data. *Advances in neural information processing systems*, 29, 2016.

[22] Z. Li, F. Zhou, F. Chen, and H. Li. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

[23] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 641–656, 2018.

[24] Y. Liu, B. Fan, S. Xiang, and C. 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.

[25] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *ICLR*, 2019.

[26] C. Ma, Y. Guo, J. Yang, and W. An. Learning multi-view representation with lstm for 3-d shape recognition and retrieval. *IEEE Transactions on Multimedia*, 21(5):1169–1182, 2018.

[27] D. Maturana and S. 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.

[28] Q. Meng, W. Wang, T. Zhou, J. Shen, Y. Jia, and L. Van Gool. Towards a weakly supervised framework for 3d point cloud object detection and annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[29] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[30] A. Nichol and J. Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.

[31] Y. Nie, J. Hou, X. Han, and M. Nießner. Rfd-net: Point scene understanding by semantic instance reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4608–4618, 2021.

[32] B. Oreshkin, P. R. López, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018.

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[34] P. Pezeshkpour, Z. Zhao, and S. Singh. On the utility of active instance selection for few-shot learning. *NeurIPS HAMLETS*, 2020.

[35] S. Pini, G. Borghi, R. Vezzani, D. Maltoni, and R. Cucchiara. A systematic comparison of depth map representations for face recognition. *Sensors*, 21(3):944, 2021.

[36] C. R. Qi, H. Su, K. Mo, and L. 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.

[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[38] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[39] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[40] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International journal of computer vision*, 66(3):231–259, 2006.

[41] A. Roy, A. Shah, K. Shah, P. Dhar, A. Cherian, and R. Chellappa. Felmi: Few shot learning with hard mixup. In *Advances in Neural Information Processing Systems*, 2022.

[42] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, 2016.

[43] N. Sedaghat, M. Zolfaghari, E. Amiri, and T. Brox. Orientation-boosted voxel nets for 3d object recognition. *arXiv preprint arXiv:1604.03351*, 2016.

[44] Y. Siddiqui, J. Thies, F. Ma, Q. Shan, M. Nießner, and A. Dai. Retrievalfuse: Neural 3d scene reconstruction with a database. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12568–12577, 2021.

[45] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

[46] H. Su, S. Maji, E. Kalogerakis, and E. 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.

[47] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, 2019.

[48] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.

[49] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019.

[50] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016.

[51] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[52] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. 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.

[53] T. Xiang, C. Zhang, Y. Song, J. Yu, and W. Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 915–924, 2021.

[54] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *European Conference on Computer Vision*, pages 1–19. Springer, 2020.

[55] J. Xu and H. Le. Generating representative samples for few-shot classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9003–9013, 2022.

[56] J. Xu, X. Luo, X. Pan, Y. Li, W. Pei, and Z. Xu. Alleviating the sample selection bias in few-shot learning by removing projection to the centroid. *Advances in Neural Information Processing Systems*, 35:21073–21086, 2022.

[57] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann. Disn: Deep implicit

surface network for high-quality single-view 3d reconstruction. *Advances in neural information processing systems*, 32, 2019.

[58] X. Yan, H. Zhan, C. Zheng, J. Gao, R. Zhang, S. Cui, and Z. Li. Let images give you more: Point cloud cross-modal training for shape analysis. *arXiv preprint arXiv:2210.04208*, 2022.

[59] R. Zhang, Z. Guo, W. Zhang, K. Li, X. Miao, B. Cui, Y. Qiao, P. Gao, and H. Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8552–8562, 2022.

[60] X. Zhu, R. Zhang, B. He, Z. Zeng, S. Zhang, and P. Gao. Pointclip v2: Adapting clip for powerful 3d open-world learning. *arXiv preprint arXiv:2211.11682*, 2022.

[61] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. 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.