P2P: Tuning Pre-trained Image Models for Point Cloud Analysis with Point-to-Pixel Prompting

Anonymous Author(s)
Affiliation
Address
email

Abstract

Nowadays, pre-training big models on large-scale datasets has become a crucial topic in deep learning. The pre-trained models with high representation ability and transferability are tuned on downstream tasks and achieve a great success in natural language processing and computer vision. However, it is non-trivial to promote such a pretraining-tuning paradigm to the 3D vision, given the limited training data that are relatively inconvenient to collect. In this paper, we propose a new perspective of leveraging pre-trained 2D knowledge in 3D domain to tackle this problem, tuning pre-trained image models with the novel *Point-to-Pixel prompting* for point cloud analysis. Following the principle of prompting engineering, we transform point clouds into colorful images with geometry-preserved projection and geometry-aware coloring to adapt to pre-trained image models, whose weights are kept frozen during the end-to-end optimization of point cloud analysis tasks. We conduct extensive experiments on object classification and part segmentation, demonstrating that the pre-trained knowledge from image models can be migrated to the 3D domain at a minor parameter cost while achieving competitive performance with the state-of-the-art point cloud recognition models.

1 Introduction

With the rapid development of deep learning and computing hardware, neural networks are experiencing explosive growth in model size and representation capacity. Nowadays, pre-training big models has become an important research topic in both natural language processing [12, 50, 4, 56] and computer vision [49, 51, 19, 53], and has achieved a great success when transferred to downstream tasks with fine-tuning [20, 9, 8, 2, 19] or prompt-tuning [49, 44, 63, 55, 29, 35] strategies. Fine-tuning is a traditional tuning strategy that requires a large amount of trainable parameters, while prompt tuning is a recently emerged lightweight scheme to convert downstream tasks into the similar form as the pre-training task. However, such prevalence of the pretraining-tuning pipeline cannot be obtained without the support of numerous training data in pre-training stage. Language pre-training leading work Megatron-Turing NLG [56] with 530 billion parameters is trained on 15 datasets containing over 338 billion tokens, while Vision MoE [53] with 14.7 billion parameters is trained on JFT-300M dataset [58] including 305 million training images.

Unfortunately, the aforementioned convention of pre-training big models on large-scale datasets and tuning on downstream tasks has encountered obstacles when we refer to the 3D vision realm. 3D visual perception is gaining more and more attention given its superiority in many emerging research fields including autonomous driving [28, 75], robotics vision [7, 73] and virtual reality [40, 67]. However, obtaining abundant 3D data such as point clouds from LiDAR is neither convenient nor inexpensive. For example, the largest object-level point cloud dataset ShapeNet [6] only contains 50 thousand synthetic samples. Therefore, pre-training fundamental 3D models with limited data

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Figure 1: **Images produced by our Point-to-Pixel Prompting.** We show the original point clouds (top line) and the projected colorful images produced by our P2P of synthetic objects from ModelNet40 (left five columns) and the objects in real-world scenarios from ScanObjectNN (right three columns) from two different projection views. Best viewed in colors.

remains an open question. There are some previous works [72, 64, 76] that attempts to develop specific pre-training strategies on point clouds with limited training data, such as Point Contrast [72], OcCo [64] and Point-BERT [76]. Although they prove that the pretraining-finetuning pipeline works well in the 3D domain, the imbalance between numerous trainable parameters and limited training data may lead to insufficient optimization or overfitting problems.

Different from the previous works that directly pre-train models on 3D data, we propose to transfer the pre-trained knowledge from image field to 3D domain with appropriate prompting engineering, since images and point clouds display the same visual world and share some common knowledge. In this way, we address the data-starvation problem in the 3D domain, given that the pre-training strategy is well-studied in the 2D field with abundant training data and that prompt-tuning on 3D tasks does not require much 3D training data. To the best of our knowledge, we are the first work to transfer knowledge in pre-trained image models to 3D vision with a novel prompting approach. More specifically, we propose an innovative Point-to-Pixel Prompting mechanism that transforms point clouds into colorful images with geometry-preserved projection and geometry-aware coloring. Examples of produced colorful images are shown in Figure 1. Then the colorful images are fed into the pre-trained image model with frozen weights to extract representative features, which are further deployed to downstream task-specific heads. The end-to-end optimization pipeline and the strategy of freezing the pre-trained image model promote the bidirectional knowledge flow between points and pixels. The geometric information from point clouds is mostly retained in projected images via our geometry-preserved projection, while the color information of natural images from the pre-trained image model is transmitted back to colorless point clouds via the cooperation between the geometry-aware coloring module and the fixed pre-trained image model.

We conduct extensive experiments to illustrate the effectiveness of the pre-training knowledge transfer paradigm and the ascendancy of our Point-to-Pixel Prompting. With much fewer trainable parameters, we achieve comparable results with the best object classification methods on both synthetic ModelNet40 [70] and real-world ScanObjectNN [61]. We also demonstrate the potential of our method to perform dense predictions like part segmentation on ShapeNetPart [74]. In conclusion, our Point-to-Pixel Prompting (P2P) framework explores the feasibility and superiority of transferring image pre-trained knowledge to the point cloud domain, bridging the gap between the two most decisive vision fields and promoting a new pre-training paradigm in 3D point cloud analysis.

2 Related Work

2.1 Visual Pre-training

Pre-training visual models has been studied thoroughly in the image domain and there are mainly three kinds of pre-training strategies: supervised, weakly-supervised and unsupervised. Supervised pre-training [14, 77, 5] on classification task with large-scale dataset is a traditional practice and is

stimulated by the boosting development of base vision models [21, 22, 14, 37]. From convolution-based networks [21, 22] to attention-based models [14, 37, 36], model size has grown rapidly and the supporting datasets has grown from ImageNet [27] with 14 million samples to JFT-3B [58] containing 3 billion images. Weakly-supervised pre-training methods [59, 3, 71, 45] use less annotations while unsupervised pre-training works [20, 9, 8, 2, 19, 16] introduces no task-related bias and brings higher transferability to various downstream tasks. Contrastive learning based methods [20, 9, 8, 16] and pretext-task-based approaches [2, 19, 13] are representative unsupervised pre-training works.

Different from the prosperity of pre-training image models, pre-training 3D models is still under development. Many works have developed self-supervised learning mechanisms with various pretext tasks such as solving jigsaw puzzles [54], orientation estimation [46], and deformation reconstruction [1]. Inspired by pre-training strategies in image domain, Point Contrast [72] adopts contrastive learning principle while OcCo [64] and Point-BERT [76] introduce reconstruction pretext tasks for better representation learning. However, the data limitation in 3D domain remains a large obstacle in developing better pre-training strategies.

2.2 Prompt Tuning

Prompt tuning is an important mechanism whose principle is to adapt downstream tasks with limited annotated data to the original pre-training task at a minimum cost, thus exploiting the pre-trained knowledge to solve downstream problems. It is first proposed in the natural language processing community [33], and has been leveraged in many vision-language models. At first, hand-crafted prompting methods like LAMA [44] and GPT-3 [4] are promoted, manually designing the cloze format. Later on, works like AdvTrigger [63] and AutoPrompt [55] develop an automated searching algorithm to select discrete prompt tokens within a large corpus. Recently, continuous prompting methods represented by Prefix-Tuning [31], PromptTuning [29] and P-Tuning [35, 34] are becoming the mainstream given their flexibility and high performance.

On the contrary, the development in prompting visual pre-training models lags behind. L2P [66] proposes a prompt pool for continual learning problem while VPT [23] takes the first step to propose continuous prompt tuning framework inspired by P-Tuning [35, 34] to traditional vision perception problems such as classification and segmentation, optimizing learnable prompt tokens with frozen pre-trained ViT [14]. In conclusion, prompt tuning in vision research field is still in the ascendant, and there are no previous works like this paper to discuss tuning pre-trained image models for point cloud analysis with an appropriate prompting mechanism.

2.3 Object-level Point Cloud Analysis

Given the unordered data structure of point clouds, early literature has developed two kinds of methods to construct structural representations for point cloud object analysis: voxel-based and point-based. Leading by VoxNet [42], voxel-based methods [42, 26, 52] partition the 3D space into ordered voxels and perform 3D convolutions for feature extraction. On the other hand, point-based methods directly process unordered points and introduce various approaches to aggregate local information, with PointNet [47] and PointNet++ [48] as pioneer works. Their successors are differentiated into multiple branches, including MLP-based [41, 11], convolution-based [32, 69, 60] and graph-based ones [65, 30]. Recently, attention-based Transformer [62, 17, 78] architecture has prevailed over other frameworks in vision community and works like PCT [17], Point Transformer [78] utilize the technique to achieve competitive performance in point cloud object analysis.

Besides the aforementioned methods that perform representation learning in the 3D space, there are projection-based works[57, 24, 68, 18, 15] that leverage multi-view images to represent 3D objects. Most methods are based on synthetics datasets with fixed multi-view images rendered from meshes in advance, developing various aggregation functions to merge features from multiple views. Recently, MVTN [18] introduces the differentiable rendering technique to build an end-to-end learning pipeline, rendering images online and regressing the optimal projection view. Different from theirs, our work designs a novel prompting engineering scheme, leveraging differentiable projection as the bridge between points and pixels. Thus we are able to utilize 2D color knowledge from pre-trained image models that is absent in colorless point clouds. Moreover, our framework is implemented in a faster single-view pattern, as we only select one random projection view during training and don't develop any aggregation strategy to explicitly fuse multi-view knowledge.

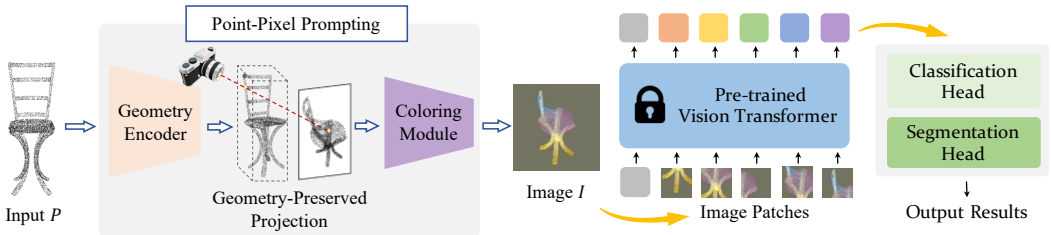

Figure 2: **The pipeline of our proposed P2P framework.** Taking a point cloud P as the input, we first encode the geometry information for each point. Then we sample a projection view and rearrange the point-wise features into an image-style layout to obtain the pixel-wise features with *Geometry-preserved Projection*. The colorless projection will be enriched to produce a colorful image I with the color information via a learnable *Coloring Module*. Our P2P framework can be easily transferred to several downstream tasks with a task-specific head with the help of the transferable visual knowledgae from the pre-trained image model.

3 Approach

3.1 Overview and Preliminaries

The overall framework of our P2P framework is illustrated in Figure 2. The network architecture consists of three components: 1) a Point-to-Pixel Prompting module to produce colorful images from point clouds, 2) a pre-trained image model to leverage pre-trained knowledge from image domain, and 3) a task-specific head to perform various kinds of point cloud tasks. We will introduce the Point-to-Pixel Prompting module and task-specific heads in detail in the following sections.

For the pre-trained image model, we choose the Vision Transformer (ViT) [14] considering its strong representation ability and it being the foundation of many developed image pre-training strategies. ViT splits input images into fixed-size patches and linearly embeds each of them into patch tokens. With additional absolute position embeddings, the patch tokens are fed to a standard Transformer encoder to extract token features via multi-head self-attention layers and FeedForward Networks. In our P2P framework, we select ViT pre-trained by MAE [19] given its strong transferability and high performance of fine-tuned results on ImageNet classification. During the training process, we froze the weights of the pre-trained ViT model to preserve the pre-trained knowledge from the image domain. Please refer to the original papers of ViT and MAE or our supplementary for more details.

With the proposed architecture that can be optimized in an end-to-end manner, we are able to exploit 2D pre-trained knowledge for point cloud analysis from two perspectives. In the forward process, the point clouds are projected into images with preserved geometry information and the resulting images can be recognized and handled by the pre-trained 2D ViT model. In the backward optimization, the frozen pre-trained ViT weights act as an anchor and guide the learnable coloring module to learn extra color knowledge for colorless point clouds, without explicit manual interference and only under the weak supervision from the overall target functions of downstream tasks. Therefore, the resulting colorful images are expected to mimic patterns in 2D images and to be distinguishable for the pre-trained ViT model in downstream tasks.

3.2 Point-to-Pixel Prompting

Following the principle of prompt tuning mechanism introduced in Section 2.2, we propose Point-to-Pixel Prompting to adapt point cloud analysis to image representation learning tasks, on which the 2D ViT model is initially pre-trained. As illustrated in Figure 2, we first perform point cloud feature encoding to extract geometric information, obtaining a rough perception of the input point cloud. Then we introduce geometry-preserved projection to transform the 3D point cloud into 2D images, rearranging 3D geometric features according to the projection correspondences. Finally, we propose a geometric-aware coloring module to dye projected images, transferring 2D color knowledge in the frozen ViT model to the colorless point cloud and obtaining more distinguishable images that can be better recognized by the pre-trained ViT model.

3.2.1 Point Cloud Feature Encoding

One of the most significant advantages of 3D point clouds over 2D images is that point clouds contain more spatial and geometric information that is compressed or even lost in flat images. Therefore, we first extract geometry features from point clouds for better spatial comprehension, implementing a lightweight DGCNN [65] to focus on the local region of each point.

Given an input point cloud $P \in \mathbb{R}^{N\times 3}$ with N points, we first locate k-nearest neighbors $\mathcal{N} \in \mathbb{R}^{N\times k\times 3}$ of each point. Then for each local region, we implement a small neural network h_Θ to encode the relative position relations between the central point p_i and the local neighbor points $\mathcal{N}_{p_i}$, resulting in geometric features $F = \{f_i, 0 \le i < N\} \in \mathbb{R}^{N\times C}$ with dimension C:

$$f_i = \mathrm{maxpool}_{\mathcal{N}_{p_i}}(h_\Theta(\mathrm{concat}_{\mathcal{N}_{p_i}}(\mathrm{x}_i, \mathrm{x}_j - \mathrm{x}_i))) \tag{1}$$

where $\mathrm{x}_i, \mathrm{x}_j$ are coordinates of p_i, p_j respectively, $\mathrm{maxpool}_{\mathcal{N}_{p_i}}$ and $\mathrm{concat}_{\mathcal{N}_{p_i}}$ stand for max-pooling and concatenation within all points p_j in local neighbor region $\mathcal{N}_{p_i}$ respectively.

3.2.2 Geometry-Preserved Projection

Once obtaining geometric features $F \in \mathbb{R}^{N\times C}$ of the input point cloud P, we further rearrange them into an image-style layout $\hat{F} \in \mathbb{R}^{H\times W\times C}$ to prepare for producing colorful image I, where H, W are height and width of the target image I. We elaborately design a geometry-preserved projection to avoid information loss when casting 3D point clouds to 2D images.

The first step is to find spatial correspondence between point coordinates $\mathrm{X} \in \mathbb{R}^{N\times 3}$ and image pixel coordinates $\mathrm{Y} \in \mathbb{R}^{N\times 2}$. Since there is a dimensional diminishing during the projection process, we randomly select a projection view during training to construct a stereoscopic space with flat image components. Equivalently, we rotate the input point cloud with rotation matrix $R \in \mathbb{R}^{3\times 3}$ to get 3D coordinates $\tilde{\mathrm{X}}$ after rotation: $\tilde{\mathrm{X}} = \mathrm{X}R^T$. The rotation matrix R is constructed through two steps: first rotating around the axis $u_\theta = (0, 0, 1)$ by angle θ, then rotating around the axis $u_\phi = (\sin\theta, -\cos\theta, 0)$ by angle ϕ, where $\theta \in [-\pi, \pi]$ and $\phi \in [-\pi/2, \pi/2]$ are random rotation angles during training and fix-selected angles during inference. Then we just omit the final dimension $\tilde{\mathrm{X}}_{:,2}$ and evenly split the first two dimensions into 2D grids: $\mathrm{y}_{i,d} = \lfloor \tilde{\mathrm{x}}_{i,d}/g_d \rfloor$, where $0 \le i < N$ denotes point index, $d = 0, 1$ denotes coordinate dimension, g_d denotes grid size at dimension d.

The second step is to rearrange per-point geometric features F into per-pixel $\hat{F}$ according to coordinates correspondence between X and Y. If there are multiple points $\mathcal{S}_{h,w} = \{p_j\}$ falling in the same pixel at (h, w), which is a common situation, we add the features of these points altogether to produce the pixel-level feature: $\hat{f}_{h,w} = \sum_{p_j \in \mathcal{S}_{h,w}} f_j$. The summation operation brings two advantages related to our geometry-preserved design. On the one hand, we take all points in one pixel into consideration instead of the common practice that only keeps the foremost point according to depth and occlusion relation. In this way, we are able to represent and optimize all points in one image and produce images containing semitransparent objects with richer geometric information as shown in Figure 1. On the other hand, we conduct a summation operation instead of taking the average, resulting in larger feature values when there are more points in one pixel. Such design maintains the spatial density information of point clouds during the projection process, which is lacked in image representations and is critical in preserving geometry knowledge.

In conclusion, our proposed geometry-preserved projection is able to produce geometry-aware image feature map $\hat{F}$ that contains plentiful spatial knowledge of the object. Note that we only utilize one projection view during training and do not explicitly design any aggregation functions for multi-view feature fusion. Therefore, we follow a single-view projection pipeline which is much more efficient than its multi-view counterpart.

3.2.3 Geometry-Aware Coloring

Despite that 3D point cloud contains richer geometric knowledge than 2D images, colorful pictures embrace more texture and color information than colorless point clouds, which is also decisive in visual comprehension. The frozen ViT pre-trained on abundant images learns to perceive the visual world not only based on object shape and outlines, but also heavily relied on discriminative colors and textures. Therefore, the image feature map $\hat{F}$ that contains only geometry knowledge and lacks

color information is not most suitable for the pre-trained ViT to understand and process. In order to better leverage pre-trained 2D knowledge of the frozen ViT model, we propose to predict colors for each pixel, explicitly encouraging the network to migrate color knowledge in the pre-trained ViT model to $\hat{F}$ via the end-to-end optimization. Since the input $\hat{F}$ contains rich geometry information that will heavily affect the coloring process, the resulting images are expected to display different colors on different geometry parts, which has been verified in Figure 1. Therefore, we referred to our dyeing process as geometry-aware coloring.

More specifically, we design a lightweight 2D neural network g_Φ to predict RGB colors $\mathrm{C} = \{c_{h,w}\} \in \mathbb{R}^{H \times W \times 3}$ for each pixel (h, w): $c_{h,w} = g_\Phi(\hat{f}_{h,w})$. We implement several 3×3 convolutions in g_Φ for image smoothing, as the initial projected image feature $\hat{F}$ are relatively discontinuous due to the sparsity of the original point cloud. Therefore, the smoothing operation is critical in producing more realistic images that the pre-trained ViT can recognize. The resulting colorful images are then prepared for further image-level feature extraction through the pre-trained ViT model.

3.3 Optimization on Downstream Tasks

The outputs from the pre-trained ViT model are image token features $\bar{F} \in \mathbb{R}^{N_t \times C_t}$ and one class token feature $\bar{f}_{\text{cls}} \in \mathbb{R}^{1 \times C_t}$, where N_t is the number of image patches and C_t is the token feature dimension. For different downstream tasks, we design different task-specific heads and optimization strategies, which are introduced in the following part.

Object Classification For object classification, we follow the common protocol in image Transformer models to utilize the class token $\bar{f}_{\text{cls}}$ as the input to the classifier CLS implemented as one linear layer: $p = \mathrm{softmax}(\mathrm{CLS}(\bar{f}_{\text{cls}})))$. We use the CrossEntropy loss as the optimization target.

Part Segmentation For part segmentation, we rearrange the token features $\bar{F}$ into image layouts and upsample them to $H \times W$. Then we design a lightweight 2D segmentation head SEG based on SemanticFPN [25] to predict per-pixel segmentation logits: $p_{h,w} = \mathrm{softmax}(\mathrm{SEG}(\bar{f}_{\mathrm{h,w}}))$. Instead of re-projecting 2D segmentation results back to 3D points and supervising with 3D labels, we project 3D labels into 2D image-style labels, exactly as how the point cloud is projected. Then we implement a per-pixel multi-label CE loss as there may be points from multiple classes projected to the same pixel: $\mathcal{L}_{\text{seg}} = \sum_{h,w} \sum_k -y_{h,w,k} \log p_{h,w,k}$. The values of multi-hot 2D label y are assigned according to projection correspondences, satisfying $\sum_k y_{h,w,k} = 1$. Supervision in 2D domain speeds up the training procedure without much information loss, since we keep all features of points in one pixel and the optimization target is accordingly based on their category distributions. During inference, we select multiple projection views and re-project 2D per-pixel segmentation results back to 3D points, fusing multi-view predictions. Therefore, the per-point segmentation is decided by the most evident predictions from the most distinguishable projection directions.

4 Experiments

4.1 Datasets and Experiment Settings

Datasets. We conduct object classification experiments on ModelNet40 [70] and ScanObjectNN [61] datasets, while ShapeNetPart [61] is utilized for object-level part segmentation. **ModelNet40** is the most popular 3D synthetic dataset for object classification, which contains 12,311 CAD models from 40 categories. We follow the common protocol to split 9,843 objects for training and reserve 2,468 objects for validation. **ScanObjectNN** is a more challenging point cloud object dataset sampled from real-world scans with background and occlusions. It contains 2,902 samples from 15 categories, and we conduct experiments on the perturbed (PB-T50-RS) variant. **ShapeNetPart** samples from the synthetic ShapeNet dataset and annotates each object with part-level labels. It consists of 16,881 objects from 16 shape categories and their parts are split into 50 classes.

Implementation Details. We implement our proposed P2P architecture with PyTorch [43] and utilize AdamW [39] optimizer with learning rate $5e^{-4}$ and weight decay $5e^{-2}$, accompanied by the CosineAnnealing learning rate scheduler [38]. We optimize the parameters of Point-to-Pixel

Table 1: Results on the ModelNet40 dataset. We use Tr. Param. to represent the number of trainable parameters.

Method	Pretrain	Tr. Param.	Acc. (%)
PointNet++ [48]	N/A	1.4M	90.7
KPConv [60]	N/A	15.2M	92.9
DGCNN [65]	N/A	1.8M	92.9
PointMLP-elite [41]	N/A	0.68M	93.6
PointMLP [41]	N/A	12.6M	94.1
DGCNN-OcCo [64]	3D	1.8M	93.0
Point-BERT [76]	3D	21.1M	93.2
P2P (ViT-S-MoCo)	2D	0.14M	91.9
P2P (ViT-B-MoCo)	2D	0.17M	92.5
P2P (ViT-B-MAE)	2D	0.17M	93.2

Table 2: Results on the ScanObjectNN(PB-T50-RS). We use Tr. Param. to represent the number of trainable parameters.

Method	Pretrain	Tr. Param.	Acc. (%)
PointNet++ [48]	N/A	1.4M	77.9
DGCNN [65]	N/A	1.8M	78.1
PRANet [11]	N/A	2.3M	82.1
MVTN [18]	N/A	14.0M	82.8
PointMLP-elite [41]	N/A	0.68M	83.8
PointMLP [41]	N/A	12.6M	85.4
Point-BERT [76]	3D	21.1M	83.1
P2P (ViT-S-MoCo)	2D	0.11M	80.2
P2P (ViT-B-MoCo)	2D	0.13M	84.2
P2P (ViT-B-MAE)	2D	0.13M	84.5

Prompting module, task-specific head and normalization layers in ViT, freezing the remaining pre-trained weights of the ViT model. The model is trained for 300 epochs with a batch size of 64. During training, the rotation angle θ is randomly selected from $[-\pi, \pi]$ and ϕ is randomly selected from $[-0.4\pi, -0.2\pi]$ to keep the objects standing upright in the images. During inference, we evenly choose 10 values of θ and 5 values of ϕ to produce 50 views for majority voting. Please refer to the supplementary material for architectural details.

4.2 Object Classification

4.2.1 Experimental Results

Results on ModelNet40. The object classification results on the synthetic ModelNet40 are shown in Table 1. For baseline comparisons, we select architecture-based methods [48, 60, 65, 41] that don't involve any pre-training strategies and traditional pre-training works [76, 64] in 3D domain. For our P2P framework, we provide comparisons among different pre-trained image models on scales (small, base) and pre-training strategies (MAE [19], MoCo [10]). Note that MAE is a relatively better pre-training strategy, as its fine-tuning result on ImageNet is better than that of MoCo.

From the results we can draw two conclusions. Firstly, our P2P outperforms traditional 3D pre-training methods, and achieves comparable performance with architecture-based works which require much more trainable parameters than ours. This suggests that the pre-trained knowledge from 2D domain is useful for solving 3D recognition problems and is better than directly pre-training on 3D datasets with limited data. Secondly, increasing the scale of the image model and using a better pre-training strategy both bring higher performance on 3D recognition tasks, which is consistent with the observations in 2D domain. Therefore, both indications verify that our P2P framework fully exploits the potential of pre-training knowledge from image domain for point cloud analysis.

Results on ScanObjectNN. The object classification results on the real-world ScanObjectNN dataset are shown in Table 2. We make similar baseline comparisons and image model comparisons as in ModelNet40 experiments. According to the results, we surpass most of the previous methods on the PB-T50-RS variant with much fewer trainable parameters. The only method that outperforms ours is PointMLP [41], which is trained with nearly $100\times$ more parameters than ours. When considering lightweight settings, our P2P (ViT-B-MAE) exceeds PointMLP-elite with about 20% trainable parameters. Moreover, the trend of increasing ViT scale and selecting a better pre-training strategy agree with conclusions from ModelNet40, additionally strengthening our statements.

Visualization Analysis. The visualizations of our projected colorful images are shown in Figure 1. The first line shows several point cloud samples from the synthetic ModelNet40 dataset and real-world ScanObjectNN dataset, while the second and third lines illustrate the colorful images from Point-to-Pixel Prompting module from different projection views. Two conclusions can be drawn from the illustration. Firstly, our geometry-preserved projection design maintains most spatial information, resulting in images of semitransparent objects that avoid occlusion problems, such as the chair leg in the second row 5th column. Secondly, the colors of the objects are reasonable and

Table 3: **Ablation studies on the ModelNet40 dataset.** We report trainable parameters as Tr. Param. and accuracy as Acc..(a) shows the effect of pre-trained knowledge, where we train ViT from scratch (A, B) without loading image pre-trained weights. (b) shows different tuning strategies of pre-trained ViT, including tuning all parameters (C), tuning only normalization layers (P2P) and keeping all ViT weights frozen (D). (c) shows different Point-to-Pixel Prompting types, discussing whether to explicitly produce images (E) and whether to predict pixel colors (F). (d) shows the deficiency of implementing VPT [23] prompting to one of the best point-based models PointMLP-Tiny (PT).

(a) Pre-trained Knowledge.

Model	Pre-train	ViT Type	Tr. Param.	Acc.(%)
A	✗	Base	82.0 M	83.8 (-9.4)
B	✗	Tiny	5.38 M	89.9 (-3.3)
P2P	✓	Base	0.17 M	93.2

(b) Tuning Strategies.

Model	All	Norm	Tr. Param.	Acc.(%)
C	✓	✓	82.0 M	91.8 (-1.4)
D	✗	✗	0.13 M	92.3 (-0.9)
P2P	✗	✓	0.17 M	93.2

(c) Point-to-Pixel Prompting.

Model	Prompt Type	Color	Tr. Param.	Acc.(%)
E	Feature	–	12.2 M	91.3 (-1.9)
F	Image	✗	0.07 M	89.6 (-3.6)
P2P	Image	✓	0.17 M	93.2

(d) Point-based Prompting.

Model	ViT	VPT	Tr. Param.	Acc.(%)
PT	✗	✗	0.17 M	92.3
PT-A	✓	✗	0.40 M	91.7 (-0.6)
PT-B	✓	✓	0.44 M	92.3 (-0.0)

even suggest part-level distinctions to some extent, making projected images more similar to natural images that can be better processed by the pre-trained image model.

Further Discussions. From the comparisons with previous literature on ModelNet40 and ScanObjectNN datasets, our P2P framework performs better on real-world ScanObjectNN dataset than other methods. This may be caused by the data distribution of ScanObjectNN dataset being more similar to the pre-trained ImageNet dataset, as they both contain visualizations of objects from the natural world. This prosperity indicates that our model successfully adapts pre-trained knowledge from image domain and reveals its potential in real-world applications.

4.2.2 Ablation Studies

Introducing 2D Pre-trained Knowledge to 3D Analysis. In Section 4.2.1, we demonstrate that leveraging pre-trained knowledge from the image domain with appropriate prompting can obtain comparable performance with state-of-the-art methods. Therefore, we conduct ablation studies to discuss whether the pre-trained weights are crucial in our P2P framework, as shown in Table 3a. In model A, we keep the same architecture as our P2P framework and train all parameters without loading pre-trained ViT weights. However, the classification accuracy decreases dramatically due to the imbalance between 82.0M trainable parameters and the limited training data of ModelNet40 dataset. Therefore, in model B, we additionally replace ViT-base with ViT-tiny to decrease the model scale. However, its classification accuracy is still far worse than P2P with pre-training knowledge, even though model B still contains much more trainable parameters than P2P. Therefore, the pre-trained knowledge from the image domain is the key component of our P2P framework.

Tuning Extent of the Pre-trained ViT Model. During the training process of our P2P framework, we froze most of the ViT parameters and only tune weights of Normalization layers. Therefore, we also conduct ablation studies on the extent of tuning ViT model: (1) Model C: tuning all ViT parameters, which is the most severe tuning degree. (2) P2P: tuning only normalization parameters, which is a modest tuning solution. (3) Model D: fix all ViT parameters without any tuning, which is the weakest tuning degree. According to the results in Table 3b, tuning normalization parameters is the most suitable solution, avoiding 2D information lost during massive tuning (model C) while adapting the model to point cloud data distribution that model D variant fails to accomplish.

Point-to-Pixel Prompting. In Section 3.2, we propose Point-to-Pixel Prompting mechanism, which is much different from the prompting approaches in NLP field. Therefore, we conduct ablations on prompting choices to illustrate that the proposed Point-to-Pixel Prompting is the most suitable prompting method for adapting pre-trained image knowledge to point cloud analysis. As the baseline experiment, we implement one of the best architecture-based works PointMLP [41] with

Table 4: **Part segmentation results on the ShapeNetPart dataset**. We report the mean IoU across all part categories mIoU_C (%) and the mean IoU across all instance mIoU_I (%) , and the IoU (%) for each category.

Model	mIoU_C	mIoU_I	aero plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor bike	mug	pistol	rocket	skate board	table
PointNet [47]	80.4	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [48]	81.9	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
DGCNN [65]	82.3	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
P2P (ViT-B-MAE)	81.7	85.0	82.6	85.2	83.8	75.5	90.7	80.9	91.2	87.4	84.8	95.5	62.0	91.2	78.9	61.6	72.5	82.9

the same linear classification head as ours and lower the feature dimension of the third block, obtaining a PointMLP-tiny (PT) version with similar trainable parameters as our P2P framework. Then we add ViT-base model with pre-trained weights from image domain and fine-tune only normalization layers as P2P, denoted as model PT-A. Finally, in model PT-B, we implement the advanced continuous prompting mechanism P-Tuning [35, 34] that adds learnable prompt tokens to the input of Transformer blocks, which has also been adopted in VPT [23]. Comparing results between model PT and model PT-A in Table 3d, we find that adding pre-trained ViT to PointMLP-tiny without any prompting mechanism will decrease the performance due to the difference between point clouds and natural images. Comparing results between model PT and model PT-B, we conclude that VPT does not work well when tuning the pre-trained 2d ViT model to point cloud classification, probably because the high-level features encoded by PointMLP-tiny cannot be understood by pre-trained ViT even with the learnable prompt tokens. Therefore, our proposed Point-to-Pixel Prompting is a better solution to tuning pre-trained 2D knowledge for point cloud analysis.

Image Coloring. In our Point-to-Pixel Prompting, we propose to produce colorful images to adapt to the pre-trained image ViT. In Section 3.2.3, we have discussed our motivation and advantages of obtaining colorful images. Here we further prove the statement via ablation studies in Table 3c. Model E processes per-pixel feature $\hat{F}$ from Section 3.2.2 to directly generate image tokens, which are the inputs to the ViT blocks. In this variant, we bypass the explicit image generation process and directly adopt patch embedding layers on feature map $\hat{F}$. Model F generates binary black-and-white images conditioned only on the geometric projection from the point cloud, without predicting pixel colors as in P2P. According to the results from Table 3c, Model E introduces much more trainable parameters due to the trainable patch embedding projection convolution layer with kernel size 16, while producing inferior classification results than P2P. On the other hand, even though Model F requires fewer trainable parameters, its performance lags far behind. Therefore, producing colorful images as the prompting mechanism can best communicate knowledge between the image domain and point cloud domain, fully exploiting pre-trained image knowledge from the frozen ViT model.

4.3 Part Segmentation

The quantitative part segmentation results on ShapeNetPart dataset are shown in Table 4. Our P2P framework can achieve comparable performance with classical point-based methods. Even though our P2P framework with 2D segmentation head does not achieve state-of-the-art performance on part segmentation, we demonstrate its potential in performing 3D dense prediction tasks based on 2D pre-trained image models. We leave it for future work to develop advanced segmentation heads and supervision strategies to better leverage pre-trained 2D knowledge in object-level or even scene-level point cloud segmentation.

5 Conclusion

In this paper, we propose a point-to-pixel prompting method to tune pre-trained image models for point cloud analysis. The pre-trained knowledge in image domain can be efficaciously adapted to 3D tasks at a low trainable parameter cost and achieve competitive performance compared with state-of-the-art point-based methods, solving the limited data problem in point cloud field that has been an obstacle for massive 3D pre-training researches. The proposed Point-to-Pixel Prompting builds a bridge between 2D and 3D domains, preserving the geometry information of point clouds in projected images while transferring color information from the pre-trained image model back to the colorless point cloud. Experimental results on object classification and part segmentation demonstrate the superiority and potential of our proposed P2P framework.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *WACV*, 2021.

[2] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[3] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *NeurIPS*, 2019.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020.

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

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

[7] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9044–9053, 2021.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*. PMLR, 2020.

[9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[10] Xinlei Chen*, Saining Xie*, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.

[11] Silin Cheng, Xiwu Chen, Xinwei He, Zhe Liu, and Xiang Bai. Pra-net: Point relation-aware network for 3d point cloud analysis. *TIP*, 2021.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

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

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 2020.

[17] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 2021.

[18] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *ICCV*, 2021.

[19] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[22] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[23] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022.

[24] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *CVPR*, 2018.

[25] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019.

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

[27] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.

[28] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019.

[29] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[30] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *AAAI*, 2018.

[31] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[32] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *NeurIPS*, 2018.

[33] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

[34] Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[35] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[36] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. *arXiv preprint arXiv:2111.09883*, 2021.

[37] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[38] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[39] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[40] Shugao Ma, Tomas Simon, Jason Saragih, Dawei Wang, Yuecheng Li, Fernando De La Torre, and Yaser Sheikh. Pixel codec avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 64–73, 2021.

[41] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. In *ICLR*, 2022.

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

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.

[44] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[45] Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V Le. Meta pseudo labels. In *CVPR*, pages 11557–11568, 2021.

[46] Omid Poursaeed, Tianxing Jiang, Han Qiao, Nayun Xu, and Vladimir G Kim. Self-supervised learning of point clouds via orientation estimation. In *3DV*. IEEE, 2020.

[47] 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.

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

[49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[50] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[51] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[52] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *CVPR*, 2017.

[53] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *NeurIPS*, 2021.

[54] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *NeurIPS*, 2019.

[55] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[56] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

[57] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015.

[58] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.

[59] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *NeurIPS*, 2017.

[60] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019.

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

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[63] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*, 2019.

[64] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. Unsupervised point cloud pre-training via occlusion completion. In *ICCV*, 2021.

[65] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ToG*, 2019.

[66] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. *arXiv preprint arXiv:2112.08654*, 2021.

[67] Shih-En Wei, Jason Saragih, Tomas Simon, Adam W Harley, Stephen Lombardi, Michal Perdoch, Alexander Hypes, Dawei Wang, Hernan Badino, and Yaser Sheikh. Vr facial animation via multiview image translation. *ACM Transactions on Graphics (TOG)*, 38(4):1–16, 2019.

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

[69] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019.

[70] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[71] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020.

[72] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, 2020.

[73] Wei Yang, Chris Paxton, Maya Cakmak, and Dieter Fox. Human grasp classification for reactive human-to-robot handovers. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11123–11130. IEEE, 2020.

[74] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ToG*, 2016.

[75] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021.

[76] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *CVPR*, 2021.

[77] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *CoRR*, 2021.

[78] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 4.3.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] There isn't any potential negative societal impacts of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Section 4.1 and supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.1 and supplemental material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We follow the common practice of the previous paper, where they didn't report the error bars.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See supplemental material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.1.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A] We doesn't use any new assets.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]