Pix4Point: Image Pretrained Transformers for 3D Point Cloud Understanding

Guocheng Qian, Xingdi Zhang, Abdullah Hamdi, Bernard Ghanem

Abstract
Pure Transformer models have achieved impressive success in natural language processing and computer vision. However, one limitation with Transformers is their need for large training data. In the realm of 3D point clouds, the availability of large datasets is a challenge, which exacerbates the issue of training Transformers for 3D tasks. In this work, we empirically study and investigate the effect of utilizing knowledge from a large number of images for point cloud understanding. We formulate a pipeline dubbed Pix4Point that allows harnessing pretrained Transformers in the image domain to improve downstream point cloud tasks. This is achieved by a modality-agnostic pure Transformer backbone with the help of tokenizer and decoder layers specialized in the 3D domain. Using image-pretrained Transformers, we observe significant performance gains of Pix4Point on the tasks of 3D point cloud classification, part segmentation, and semantic segmentation on ScanObjectNN, ShapeNetPart, and S3DIS benchmarks, respectively. Our code and models are available at: https://github.com/guochengqian/Pix4Point.

1 Introduction
Point cloud representation is one of the most essential 3D representations, with wide applications in semantic segmentation, object detection, registration, etc. Due to breakthrough progress made by deep learning (Krizhevsky, Sutskever, and Hinton 2012; He et al. 2015; Qi et al. 2017a,b), neural networks have become the de facto technique for point cloud understanding and have been attracting increasing interest in recent years. Despite impressive performance gains led by learning-based approaches, the demand for massive labeled training data limits the possible applications of deep neural networks in point clouds. Unfortunately, labeled point clouds are quite expensive to acquire, while 3D data cleaning and dense annotation might cost even more (Hu et al. 2021).

In contrast to expensive point clouds, 2D images are much cheaper and easier to access. For comparison, ModelNet (Wu et al. 2015), the widely used point cloud classification dataset, consists of only 12,311 CAD models. On the other hand, ImageNet (Deng et al. 2009) contains more than a million (1,331,167) images, which is more than 100 times larger than ModelNet. In addition, there are an uncountable number of public images on the Internet. Therefore, this work aims to investigate how to take advantage of cheaper images to understand more expensive point clouds.

Previous attempts to leverage image knowledge for point clouds mainly rely on two approaches: (i) multi-view methods (Su et al. 2015; Hamdi, Giancola, and Ghanem 2021a) that project 3D point clouds into images and subsequently process the 2D images instead for point cloud perception; and (ii) kernel inflation (Xu et al. 2021), i.e. inflating 2D convolution kernels into 3D kernels for voxel-based point cloud processing. However, both approaches lead to performance degradation. Multi-view methods lead to misinformed views, and a loss of geometric information (Alcorn et al. 2019; Hamdi, Giancola, and Ghanem 2021a), while kernel inflation only shows insignificant performance gains due to the gap between 2D and 3D kernels (Xu et al. 2021). Therefore, in this work, we propose a promising question: is it possible to pretrain a point cloud network backbone on cheap images directly?

Our answer to this question is Pix4Point, which is a simple yet effective pipeline that allows utilizing an image-pretrained Transformer to understand point clouds in their native formats. As illustrated in Fig. 1, Pix4Point pretrains a Transformer backbone on a large amount of tokenized (i.e. patchified) images (e.g. from ImageNet (Deng et al. 2009)) and then finetunes this image-pretrained Transformer in point-cloud tasks with tokenized point clouds as input. To the best of our knowledge, our Pix4Point is the first pipeline to directly
leverage the weights learned in the image domain to improve the performance for 3D point cloud tasks without any change in the network’s backbone or using image projections.

Contributions: (i) We propose Pix4Point, a novel, simple, yet effective Transformer-based pipeline that facilitates image pretraining for point cloud understanding. This pipeline enables the processing of images and point clouds in their native formats. (ii) Pix4Point image pretraining allows for performance improvement in 3D understanding tasks on standard benchmarks: ScanObjectNN, S3DIS, and ShapeNetPart.

2 Related Work

2.1 3D Point Cloud Understanding

Point Cloud Networks. Due to breakthrough progress in deep learning technology (Krizhevsky, Sutskever, and Hinton 2012; He et al. 2015; Qi et al. 2017a,b), current prevailing point cloud processing methods are entirely data-driven and consist of three main approaches: view-based (Su et al. 2015; Hamdi, Giancola, and Ghanem 2021a; Goyal et al. 2021), voxel-based (Maturana and Scherer 2015-09; Graham, Engelcke, and Van Der Maaten 2018; Choy, Gwak, and Savarese 2019), and point-based (Qi et al. 2017a,b; Wang et al. 2019a; Li et al. 2021; Qian et al. 2022) that directly take point clouds as input and process in their unstructured format. Among the aforementioned approaches, point-based methods are receiving increasing attention since there is no information loss. In this work, we pay attention to point-based methods, and specifically, Transformer-based networks (Vaswani et al. 2017), that have been proposed as a new paradigm of processing point clouds (Zhao et al. 2021; Guo et al. 2021; Lai et al. 2022). Unlike previous studies that propose new Transformer-like architectures, we instead keep the Transformer architecture unchanged, while focusing on the overall pipeline that transfers image knowledge to point clouds.

2D to 3D Transfer learning. The simplest solution is to use view-based methods, as aforementioned. As a specific example, PointCLIP (Zhang et al. 2022) showed that a pretrained CLIP (Radford et al. 2021) model can be used directly for zero-shot point cloud classification without any finetuning. More advanced methods leverage point-pixel correspondence (Liu, Qi, and Fu 2021; Hamdi, Giancola, and Ghanem 2021b) between point clouds and multi-view images, where pointwise pseudo-labels can be generated from multi-view images to (pre)train a point cloud network. More recently, Image2Point (Xu et al. 2021) presents a kernel inflation technique that expands 2D kernels of a 2D convolutional neural network into 3D kernels and further applies them to voxel-based point cloud understanding. In this work, we show a novel pipeline that can directly utilize the same backbone Transformer pretrained in images for point cloud applications, without multi-view image projection or kernel inflation.

Self-supervised learning on point clouds. One prominent way to tackle the need for large labeled data is by relying on self-supervised learning (SSL) (Chen et al. 2020; Caron et al. 2021). Previous works in SSL for point clouds rely on pretext tasks (Achlioptas et al. 2018; Sauder and Sievers 2019) or auto-encoders (Li, Chen, and Lee 2018; Achlioptas et al. 2018; Wang et al. 2021; Yan et al. 2022). PointContrast (Xie et al. 2020) proposes to generate two views of the point cloud with random transformations and ask the neural network to minimize the distance between matched points. On the other hand, Yu et al. (Yu et al. 2022) propose a BERT-like Transformer and achieves equal state-of-the-art performance on point cloud tasks. Furthermore, Point-MAE (Pang et al. 2022) propose a masked auto-encoder to learn useful representation for 3D classification.

2.2 Transformers

Specialized Transformers Architectures. Based on the multi-head attention block, the Transformer (Vaswani et al. 2017) has been believed to be the most successful architecture for the natural language attention process (NLP). Vision Transformer (ViT) explores the direct application of the Transformer on image patches to solve vision tasks (Dosovitskiy et al. 2021). There is also an increasing interest in Transformer-like architectures for point cloud processing. PointASNL (Yan et al. 2020) and Point Transformer (Zhao et al. 2021) introduce a self-attention mechanism to extract features from local groups of points. PointGMM (Hertz et al. 2020) introduces attention splits to shape embedding and manipulation. Engel, Belagiannis, and Dietmayer (2021) and Guo et al. (2021) integrate Transformer blocks into point convolutions. Hui et al. (2021) proposes a pyramid transformer for point cloud processing. Unlike these previous works, we do not aim to investigate a specialized architecture for processing point cloud. Instead, we are interested in harnessing the knowledge from cheaper images to improve downstream performance in 3D point clouds.

Transformer as a General Backbone. (Lu et al. 2021) proposes a pretrained Transformer as a universal backbone for protein prediction and language understanding. GATO (Reed et al. 2022) is proposed to be a generalist agent that can perform tasks ranging from image captioning and language tasks to robot manipulation and reinforcement learning. Most recently, Omnivoire (Girdhar et al. 2022) proposes a vision Transformer model that works on multiple modalities such as videos and RGBD data, showing the benefit of 2D training to improve performance on different modalities. In our work, we utilize the 2D image pretrained Transformers. But unlike Omnivoire and GATO, we investigate image-pretrained Transformers on point cloud tasks.

3 Methodology

The primary assumption in our work is that 3D point clouds are expensive to acquire and other rich domains (e.g., images) should be exploited. This condition contrasts with most 3D point cloud pretraining pipelines that assume that only 3D point clouds can facilitate rich representations for point clouds themselves (Xie et al. 2020; Yu et al. 2022; Pang et al. 2022). This section introduces the concept of Pix4Point and presents a practical framework on how to learn the network backbone from the 2D image domain and transfer knowledge to the domain of 3D point clouds. The full pipeline of Pix4Point is illustrated in Fig.2.
3.1 A Generalist Network for 2D and 3D
Pix4Point shows a generic network for processing images and point clouds in their native formats. This generic network consists of three stages: domain tokenizer, Transformer backbone, and task decoder. While the backbone architecture is the same standard Transformer network for both 2D and 3D, the tokenizer is specified by different modalities, and the decoder is different from task to task.

**Domain tokenizer** \( t \). The tokenizer is the first module in our network that groups and projects the raw input into a number of tokens in space \( \mathbb{R}^C \) shared by 2D and 3D. The tokenizer is domain-specific and is constructed by different modules for different modalities. The image tokenizer could be a flattening linear layer of image patches such as the tokenizer in ViT (Dosovitskiy et al. 2021). Regarding 3D point clouds, we consider the input \( X \in \mathbb{R}^{N \times (3+C_{in})} \), representing \( N \) points in 3D and \( C_{in} \) input features (e.g. colors, normals) per point. The point cloud tokenizer \( t: \mathbb{R}^{N \times (3+C_{in})} \rightarrow \mathbb{R}^{N_C \times C} \) is composed of two stages: (i) a downsampling layer to sample \( N_C \) center points, and (ii) local aggregation layers that query neighbors for each center point and aggregate the locality. The \( N_C \times C \) point patches obtained are named point tokens. In Pix4Point, we use the farthest point sampling to sample \( N_C = N/16 \) center points and graph convolution to extract features. The graph convolution in our point cloud tokenizer \( t \) is defined in Eqn. 1:

\[
\begin{align*}
    x_{ij} &= h_1(\theta)((p_j - p_i; x_j - x_i)), \\
    x_i &= h_2(\theta)([x_{ij}; \text{MAX}_j (x_{ij})];
\end{align*}
\]

where \( p_i, x_i \) denotes the coordinates and features of the \( i \)-th center point. The index \( j \) defines the neighbor of the point \( i \). The proposed graph convolution extracts the features for each center point from the relative positions and relative features. The maxpooled features inside each neighborhood, \( \text{MAX}_{j(i,j)\in\mathcal{N}}x_{ij} \), is further utilized to ensure the tokens obtained contain local information. The final output of the tokenizer is the coordinates \( p \) of the center points and their updated features \( x \).

**Transformer backbone** \( F \). Typical 3D point cloud networks \( F(X) \) are defined as a composition of learned operations on individual points with shared weights followed by the aggregation (e.g. maxpooling) to extract features. In Pix4Point, \( F: \mathbb{R}^{(N_c+1) \times C} \rightarrow \mathbb{R}^{(N_c+1) \times C} \) is a pure Transformer network that accepts a set of \( N_c \) tokens as input. Note that \( (N_c + 1) \) denotes an additional [CLS] token is appended to the \( N_c \) point tokens. The Transformer is a generalist model, where the input tokens can be from any domain. ViT (Dosovitskiy et al. 2021) is a specialized Transformer-based model for the image domain where each token is a linear embedding of different image patches. In the Pix4Point network, the input point cloud is projected into point tokens and the same standard Transformer backbone as ViT is leveraged to process these tokens. Due to the fact that the same Transformer backbone can be used for both images and point clouds, we can use the image-pretrained Transformer as an initialization for the Transformer backbone in Pix4Point. This enables transferring image knowledge to point clouds without extra operations (e.g. kernel inflation, view projection). Through-
out experiments, Pix4Point uses the standard Transformer in ViT-S, the small variant of the Vision Transformer (Dosovitskiy et al. 2021), as the backbone by default, which consists of 12 self-attention layers with 6 heads and a channel size equal to 384.

Task decoder g. The goal of the task decoder g is to specialize the general features learned by the Transformer backbone for a specific downstream task. In the case of classification, the decoder g produces global logits to define the probability of each class. These global logits are obtained through two components: (1) a global aggregation layer that aggregates all point tokens through global maxpooling and concatenates the [CLS] token; and (2) three linear layers with batch normalization, ReLU, and dropout inbetween that process the two concatenated global information and yield final outputs. For segmentation, the decoder gradually interpolates the token features to the same size as the raw input to obtain the individual representation for each point. Global representations by global maxpooling and the [CLS] token are appended to each point. This global representation appending is also shown to be beneficial for the segmentation task (refer to Appendix for the ablation study). The output of the global representation appending for every point \( x_i \) in the segmentation task can be described as follows: \( [x_i; \text{MAX} (\{g(x)\})]: [\text{CLS}] \).

3.2 Pix4Point Training

We illustrate the pipeline of Pix4Point, pretraining image Transformers for point clouds, in Fig. 2. Pix4Point pipeline consists of three stages: (1) pretraining on images, (2) transferring image pretrained weights to the Pix4Point network, and (3) finetuning on point cloud tasks.

Pretraining stage. The goal of this stage is to utilize the large number of available images to pretrain the potential of Transformer backbone of Pix4Point. By default, we pretrain ViT-S using the training strategies for ImageNet-21K (Deng et al. 2009) dataset. Note that our Pix4Point is quite flexible with the pretrained stage. Besides the aforementioned supervised pretraining, there are many other applicable strategies, e.g., self-supervised training through DINOS (Caron et al. 2021). Furthermore, pretraining epochs and dataset size might also affect downstream performance in point cloud tasks. Sec. 5 shows the effects of different pretraining strategies.

Weights transferring stage. The goal of this stage is to transfer the image-pretrained weights to a point cloud network. Unlike previous methods that require view projections or kernel inflation, in Pix4Point, this stage is as simple as using the weights of the pretrained Transformer as an initialization for the backbone of our Pix4Point network. Experiments in sec. 5 also show that the image-pretrained Transformer can still recognize point clouds by training only the tokenizer and decoder in the downstream task, while keeping the Transformer backbone frozen.

Finetuning stage. The finetuning stage finetunes the image-pretrained Transformer backbone F in point cloud tasks. The domain-specific tokenizer t and the task-specific decoder g are also jointly trained from scratch during finetuning. The loss for Pix4Point can be described as follows:

\[
\arg \min_{\theta_t, \theta_F, \theta_g} \sum_i^B L \left( g(F(t(X))), y_i \right),
\]

where \( \{y_i\}_{i=1}^P \) defines the labels of a mini-batch of input point clouds \( X \) and \( L \) is a Cross-Entropy (CE) loss.

4 Experiments and Results

We conduct extensive experiments on various benchmarks, including real-world point cloud classification on ScanObjectNN (Uy et al. 2019), semantic segmentation on S3DIS (Armenti et al. 2016), and part segmentation on ShapeNetPart (Chang et al. 2015) to verify the strength of Pix4Point.

4.1 Training Setup

Pretraining setup. We pretrain ViT-S (Dosovitskiy et al. 2021) using the training strategies provided in DeIT (Touvron et al. 2021) without distillation in ImageNet-21k (Deng et al. 2009) dataset. The weights of the ViT-S backbone from the

| Method       | OA  | mAcc |
|--------------|-----|------|
| PointNet (2017a) | 68.2 | 63.4 |
| PointNet++ (2017b) | 77.9 | 75.4 |
| PointCNN (2018)     | 78.5 | 75.1 |
| DGCNN (2019b)       | 78.1 | 73.6 |
| PointMLP (2022)     | 86.4±1.3 | 83.9±1.5 |

| Method       | mAcc |
|--------------|------|
| Point-BERT (2022) | 83.1metal |
| Point-MAE (2022) | 85.2metal |
| Pix4Point (scratch) | 85.2±0.2 | 83.3±0.2 |
| Pix4Point (pretrained) | 86.8±0.2 (±1.6) | 84.9±0.6 (±1.6) |

Table 1: 3D Object Classification on ScanObjectNN. Pix4Point (scratch) denotes training the entire network (domain tokenizer, Transformer backbone, task decoder) from random initialization, while Pix4Point (pretrained) denotes training with a random initialized tokenizer and decoder and an image-pretrained Transformer backbone. Methods are divided into two categories. Bottom: standard Transformer-based methods. Top: other methods. Results of mean±std of three random runs are provided. Improvements using image pretraining are highlighted in green color.

| Method       | mAcc  |
|--------------|-------|
| PointNet (2017a) | 49.0  |
| PointNet++ (2017b) | -    |
| DeepGCN (2019) | -     |
| PVCNN (2019)    | -     |
| KPCNN (2019)    | 72.8  |
| ASSANet-L (2021) | -     |
| PCTransformer (2021) | 67.7  |
| Point Transformer (2021) | 76.5  |

| Method       | mIoU  |
|--------------|-------|
| Pix4Point (scratch) | 68.4±0.5 | 62.3±0.7 |
| Pix4Point (pretrained) | 73.7±0.6 (±5.3) | 67.5±0.6 (±5.2) |

Table 2: Semantic Segmentation on S3DIS Area 5. Training Pix4Point with an image-pretrained Transformer backbone improves the performance considerably.
Table 3: Part Segmentation on ShapeNetPart.

| Method             | Ins. mIoU | cls. mIoU |
|--------------------|-----------|-----------|
| PointNet (2017a)   | 83.7      | 80.4      |
| PointNet++ (2017b)| 85.1      | 81.9      |
| DGCNN (2019b)      | 85.2      | 82.3      |
| KPConv (2019)      | 86.4      | 85.1      |
| CurveNet (2021)    | 86.8      | -         |
| ASSANet-L (2021)   | 86.1      | -         |
| PCTransformer (2021)| 86.4   | -         |
| Point Transformer (2021) | 86.6 | 83.7 |
| PointMLP (2022)    | 86.1      | 84.6      |
| StratifiedFormer (2022) | 86.6 | 85.1 |
| Point-BERT (2022)  | 85.6      | 84.0      |
| Point-MAE (2022)   | 86.1      | 84.2      |
| Pix4Point (scratch)| 86.3±0.1  | 84.2±0.2  |
| Pix4Point (pretrained)| 86.5±0.1 (+0.3) | 84.5±0.1 (+0.3) |

Figure 3: Qualitative Results of Pix4Point on S3DIS Area 5. Pix4Point with image pretraining (3rd column) achieves more precise segmentation results than Pix4Point trained from scratch (2nd column).

4.2 Classification

Dataset. ScanObjectNN (Uy et al. 2019) collects a total of 15,000 scanned objects for 15 classes. This real-world dataset presents challenges to classification tasks due to inherent scan noise and occlusion.

Results. Tab. 1 shows the effectiveness of Pix4Point in point cloud classification in the real-world dataset ScanObjectNN. Image pretraining improves both the overall accuracy (OA) and the mean accuracy (mAcc) of Pix4Point by 1.6. With
image pretraining, Pix4Point surpasses the most recent self-supervisely pretrained network Point-MAE (Pang et al. 2022) and the state-of-the-art PointMLP (Ma et al. 2022) in terms of OA and mAcc. This observation verifies our argument: even from different domains, image pretraining can be beneficial for point cloud understanding.

4.3 Semantic Segmentation

Dataset. S3DIS (Armeni et al. 2016) (Stanford Large-Scale 3D Indoor Spaces) dataset provides instance-level semantic segmentation for large scale scenes. S3DIS is collected from 6 large indoor areas that cover 271 rooms and 13 semantic categories. Following common practice, we leave area 5 as testing and other areas as training.

Results. We show the results of Pix4Point compared to the state-of-the-art methods in S3DIS area 5 in Tab. 2. The image-pretrained Transformer surprisingly improves Pix4Point by 5.3 in mean accuracy (mAcc) and 5.2 in mean IoU (mIoU). This achievement is simply due to initializing our Pix4Point backbone with an image-pretrained Transformer. We note that in the current landscape, the performance of a standard Transformer in large-scale benchmarks is still low compared to the classical point-based methods, since the Transformer is considered to be harder to optimize than convolutional neural networks. Despite this challenge, our Pix4Point performs better than many representative point-based methods such as PointNet++, PVCNN, and KPConv.

Qualitative Results. Fig. 3 shows the qualitative results of Pix4Point trained from scratch and Pix4Point finetuned with image pretraining. The latter yields more precise segmentation maps than the former. Closeups show that image pretraining helps Pix4Point successfully segments the board (1st raw), the clutter on the tables (1st and 2nd raw), and the bookcase (2nd row).

4.4 Part Segmentation

Dataset. ShapeNetPart (Yi et al. 2016) is a richly annotated, large-scale 3D dataset of 16 shape categories selected from the ShapeNet dataset, annotated with part-level semantic labels. It consists of 16,880 models, 2-6 parts for each category, and 50 part labels in total. Pix4Point learns a single model using a single head layer to predict all parts.

Results. The performance of Pix4Point on ShapeNetPart is given in Tab. 3. Compared to training from scratch, Pix4Point with image-pretraining achieves +0.2 in instance mIoU and +0.3 in terms of class mIoU. With image pretraining, Pix4Point is able to achieve a performance (86.5 instance mIoU) comparable to that of the state-of-the-art point-based method CurveNet.

5 Ablation Study and Analysis

In this section, we study the effects of pretraining strategies on the downstream point cloud tasks, including pretraining in different datasets in a supervised or self-supervised manner, and the size of pretraining dataset. We further ablate the design of the graph convolution (Eqn. 1). Additional ablation studies on the number of pretraining epochs and the decoder of the Pix4Point network are available in Appendix.

Figure 4: Ablation Study: the Effects of Pretraining Methods with and without a Frozen Transformer. We show curves of downstream performance in S3DIS area 5 (val mIoU) with the backbone pretrained using different methods: from scratch (random initialization), self-supervised pretraining in ShapeNet by Point-MAE, supervised pretraining in ImageNet-1K by DeiT (Pix4Point). Frozen Transformer results are included for reference. As observed, image-pretraining improves point cloud understanding more than the usual point cloud pretraining in ShapeNet.

Pretraining methods. Pix4Point pretrains the Transformer using DeiT on ImageNet-21K by default. Apart from this supervision in images, there are multiple other ways to pretrain the Transformer backbone. It can be pretrained in 3D or 2D datasets, in a supervised or self-supervised manner. In this work, we compare the following pretraining strategies: (1) supervised pretraining in ImageNet-1K using DeiT (Touvron et al. 2021) without distillation; (2) self-supervised pretraining in ImageNet-1K using DINO (Caron et al. 2021); (3) supervised pretraining the Pix4Point network in the shapeNet part segmentation task; (4) self-supervised pretraining in ShapeNet using Point-MAE (Pang et al. 2022).

Tab. 4 compares the finetuning performance in S3DIS of the aforementioned training methods. The proposed image-pretrained Transformers reach mIoUs greater than 65.6 for both DeiT supervised pretraining (65.8 mIoU) and DINO self-supervised pretraining (65.6 mIoU), outperforming the random initialized Transformer (62.3 mIoU) by more than 3.3 mIoU. More interestingly, the image-pretrained Transformers also surpass the point cloud pretrained Transformers (both supervised and self-supervised pretrained) by non-trivial margins. Fig. 4 (solid lines) shows the val mIoU during finetuning. It is observed that image-pretrained Transformer always outperforms random initialized Transformer and point cloud pretrained Transformer across finetuning epochs. These experiments clearly demonstrate the benefits of image-pretraining for point cloud understanding.

Effects of Image Pretraining without Finetuning Transformer Backbone. Previously, we show that image pretraining improves Pix4Point performance in various benchmarks by training the domain-specific tokenizer and decoder, as well as finetuning the pretrained backbone. Here, we keep the weights of the pretrained backbone frozen and only update
Table 4: Ablation Study on the Effects of Pretraining Methods. We show the downstream performance of the Pix4Point network in S3DIS area 5 using the Transformer backbone pretrained by (1) supervised pretraining in ShapeNet part segmentation task (2nd row), (2) self-supervised pretraining in ShapeNet by Point-MAE (Pang et al. 2022) (3rd row), (3) supervised pretraining in ImageNet-1K by DeiT (4th row), and (4) self-supervised pretraining in ImageNet-1K by DINO (5th row). The random initialized Transformer backbone (1st row) is provided for comparison.

| Method | Finetuning Entire Network | Finetuning with Frozen Backbone |
|--------|----------------------------|-------------------------------|
|        | mACC | mIoU | mACC | mIoU |
| random initialization | 68.4 ± 0.5 | 62.3 ± 0.7 | 51.8 ± 0.5 | 44.3 ± 0.2 |
| self-supervised pretraining in ShapeNet by Point-MAE | 69.3 ± 0.2 (±0.9) | 63.0 ± 0.4 (±0.7) | 53.9 ± 1.6 (±2.1) | 46.5 ± 1.8 (±2.2) |
| supervised pretraining in ShapeNet | 59.8 ± 1.1 (±8.6) | 53.1 ± 0.9 (±9.2) | 58.0 ± 1.6 (±6.2) | 50.1 ± 1.3 (±5.8) |
| self-supervised pretraining in ImageNet-1K by DINO | 71.8 ± 0.5 (±3.4) | 65.6 ± 0.5 (±3.3) | 58.8 ± 0.2 (±7.0) | 52.9 ± 0.0 (±7.7) |
| supervised pretraining in ImageNet-1K by DeiT (ours) | 72.1 ± 0.4 (±4.1) | 65.8 ± 0.3 (±3.5) | 58.5 ± 1.0 (±6.7) | 51.4 ± 1.2 (±7.1) |

Figure 5: Ablation Study: the Effects of Pretraining Dataset Size. We show curves of downstream performance in S3DIS area 5 (val mIoU and mAcc) vs the portion of sampled ImageNet-1K used in pretraining. As observed, image-pretraining improves point cloud understanding, and the improvement is increased with the size of the pretraining dataset.

Table 5: Ablation Study: the Effects of Relative Positions and Relative Features in the Tokenizer.

| Method | mAcc | mIoU |
|--------|------|------|
| p_j - p_i: x_j | 62.8 ± 0.7 | 56.4 ± 0.9 |
| p_j: x_j - x_i | 68.8 ± 0.2 | 62.3 ± 0.2 |
| p_j: x_j | 63.3 ± 0.4 | 56.6 ± 0.3 |
| p_j - p_i: x_j - x_i (Ours) | 71.8 ± 0.5 | 65.6 ± 0.5 |

the weights of the tokenizer and decoder during finetuning. As shown in Tab. 4 and Fig. 4 (dash lines), in the setting of freezing Transformer, the image-pretrained Transformers still reach more than 50% mIoU for both DeiT supervised pretraining and DINO self-supervised pretraining, outperforming the random initialized Transformer (51.8 mIoU) by ~ 7 mIoU. Image-pretrained Transformers also perform better than Transformers pretrained in the point cloud dataset, i.e. Point-MAE and supervised pretraining in ShapeNet.

Pretraining dataset size. We show the effects of the pretrained Transformer in ImageNet-21K by default. Here, we ablate the size of the pretrained dataset by pretraining Transformer using only a portion of ImageNet-1K. All models are trained using DeiT (Touvron et al. 2021) without distillation by 300 epochs. Fig. 5 shows that the downstream performance in point cloud segmentation in S3DIS increases with the size of pretraining dataset. Note that further increasing the size of the dataset from ImageNet-1K to ImageNet-21K (10 times larger than 1K) can still improve the performance further by 1.7 mIoU.

Tokenizer. We propose to use the concatenated relative positions and relative features [p_j - p_i: x_j - x_i] as input in our graph convolution (see Eq. 1). Tab. 5 shows an ablation study of the relative inputs on performance in S3DIS area 5 segmentation. Significant performance drops are observed when replacing the relative positions p_j - p_i with p_j or the relative features x_j - x_i with x_j. This ablation study demonstrates that both the relative positions and the relative features are essential for the tokenizer.

6 Conclusions and Future Work

This work presents an extensive empirical study on the image-pretrained Transformer for point cloud processing in their native formats. New observations and findings are demonstrated in this work: (1) the image-pretrained Transformer can be applied directly to point clouds and significantly improve performance in various 3D tasks, including classification, part segmentation, and semantic segmentation; (2) the downstream performance in the point cloud domain increases with the size of the pretraining dataset; (3) the ImageNet pretraining outperforms the ShapeNet pretraining. We believe that our findings will benefit the community and motivate future work in this direction.

While the scope of this work is to investigate the benefits of using image-based Transformers in the point cloud domain, a possible future work is to investigate cross-modality transfer learning / domain adaptation. For example, see whether language models can be used as an initialization for point clouds or images task and whether point clouds models can benefit language or image understanding? Another promising future work is to jointly optimize both images and point clouds, creating a generalist vision Transformer backbone that performs across the 2D and 3D domains.

Acknowledgement. The authors would like to thank Hani...
Itani for his insightful discussions. This work was supported by the KAUST Office of Sponsored Research (OSR) through the Visual Computing Center (VCC) funding.

A Additional Experiments

A.1 Shape Classification

Dataset. ModelNet (Wu et al. 2015) is a 3D dataset of 40 shape categories and 12,311 models.

Results. The performance of Pix4Point on ModelNet40 is given in Tab. I. Compared to training from scratch, Pix4Point with image-pretraining achieves +0.1 in mAcc and +0.2 in terms of overall Accuracy.

Table I: Shape Classification on ModelNet40. Image pretraining improves classification performance on ModelNet40.

| Method             | mAcc   | OA    |
|--------------------|--------|-------|
| PointNet (2017a)   | 86.2   | 89.2  |
| PointNet++ (2017b) | -      | 91.9  |
| PointCNN (2018)    | 88.1   | 92.2  |
| PointConv (2019)   | -      | 92.5  |
| KPConv (2019)      | -      | 92.9  |
| DGCNN (2019b)      | 90.2   | 92.9  |
| DeepGCN (2021)     | 90.9   | 93.6  |
| ASSANet-L (2021)   | -      | 92.9  |
| Point Cloud Transformer (2021) | - | 93.2 |
| Point Transformer (2021) | 90.6 | 93.7 |
| CurveNet (2021)    | -      | 94.2  |
| PointMLP (2022)    | 91.4   | 94.5  |
| Point-BERT (2022)  | -      | 93.8  |
| Point-MAE (2022)   | -      | 94.0  |
| Pix4Point (scratch)| 90.0 ± 0.9 | 93.7 ± 0.5 |
| Pix4Point (pretrained) | 90.1 ± 0.2 (+0.1) | 93.9 ± 0.1 (+0.2) |

B Additional Ablation Studies

B.1 Additional ablation studies on pretraining strategies

Pretraining methods. As shown in Fig. 1a and Fig. 1b, random initialized Transformer achieves higher training mIoUs but lower validation mIoUs than Point-MAE self-supervised pretrained Transformer, showing the generalizability of self-supervised training. We also notice that the ShapeNet supervised pretrained Transformer yields even worse performance than random initialization, mostly because the former overfits to ShapeNet.

Fig. II further shows the Pix4Point training curves with and without image pretraining in the ShapeNetPart dataset. As observed, image pretraining improves training and validation performance.

Pretraining epochs. Here we study the effects of pretraining epochs on the downstream performance. We train DeiT using the entire training set of ImageNet-1K. We leverage the pretrained Transformers from different epochs and test their performance in S3DIS area 5. Fig. III shows that the downstream performance drops at first mainly because the Transformer does not converge to good minima yet. The pretrained Transformer starts to outperform the random initialized Transformer (62.3 mIoU) at around epoch 50, gradually increases to reach the peak performance (~ 66 mIoU) at epoch 500, and begins to decline from epoch 600 mainly due to overfitting image data.

B.2 Additional Ablation Study on Pix4Point Network

Decoder. We append global representations to the segmentation head, which takes \([x_i; \text{MAX}_i(g(x_i)); \text{[CLS]])\) as input. Here, we ablate the performance of Pix4Point without global representations in Tab. II. Appending the global information of the \([\text{CLS}]\) token and the globally maxpooled features can improve the performance in terms of mIoU.

Table II: Ablation Study: Choices of the Global Representation on 3D Segmentation of S3DIS Srea 5.

| Method               | mAcc   | mIoU  |
|----------------------|--------|-------|
| NONE                 | 73.0 ± 0.3 | 66.8 ± 0.5 |
| CLS                  | 74.0 ± 0.9 | 67.3 ± 0.6 |
| CLS;MAX (ours)       | 73.7 ± 0.6 | 67.5 ± 0.9 |
| CLS;MAX;AVG          | 73.6 ± 0.5 | 66.9 ± 0.1 |
Figure I: Training Plots: The Effects of Pretraining Methods and Freezing the Transformer. We show curves of downstream performance in S3DIS area 5 (train and val mIoU) vs. finetuning epochs of Pix4Point pretrained using different pretraining strategies including no pretraining (random initialization) in black curve, ShapeNet supervised pretraining in green, ShapeNet self-supervised pretraining by Point-MAE in blue, supervised pretraining in ImageNet-1K by DeiT in red, self-supervised pretraining in ImageNet-1K by DINO in purple. We show two cases: (a) full pipeline finetuning, (b) finetuning with frozen Transformer.

Figure II: Finetuning Epoch vs. Downstream Performance on Shape-Net Parts dataset. The curves correspond to training from scratch are colored black, while the curves of Pix4point are colored in red.

Figure III: Ablation Study: The Effects of Pretraining Epochs. We show curves of downstream performance in S3DIS area 5 (val mIoU and mAcc) vs the epochs used in pretraining the Transformer on sampled ImageNet-1K. As observed, image-pretraining improves point cloud understanding, and the improvement is increased with the amount of pertaining the Transformer on ImageNet.
Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 16259–16268.