remote sensing

MDPI

Article

Advanced Feature Learning on Point Clouds Using Multi-Resolution Features and Learnable Pooling

Kevin Tirta Wijaya [1,†], Dong-Hee Paek [2,†] and Seung-Hyun Kong [2,*]

[1] Computer Graphics Department, Max Planck Institute for Informatics, 66123 Saarbrücken, Germany; kwijaya@mpi-inf.mpg.de
[2] CCS Graduate School of Mobility, Korea Advanced Institute of Science and Technology, Daejeon 34051, Republic of Korea; donghee.paek@kaist.ac.kr
* Correspondence: skong@kaist.ac.kr
† These authors contributed equally to this work.

Abstract: Existing point cloud feature learning networks often learn high-semantic point features representing the global context by incorporating sampling, neighborhood grouping, neighborhood-wise feature learning, and feature aggregation. However, this process may result in a substantial loss of granular information due to the sampling operation and the widely-used max pooling feature aggregation, which neglects information from non-maximum point features. Consequently, the resulting high-semantic point features could be insufficient to represent the local context, hindering the network's ability to distinguish fine shapes. To address this problem, we propose PointStack, a novel point cloud feature learning network that utilizes multi-resolution feature learning and learnable pooling (LP). PointStack aggregates point features of various resolutions across multiple layers to capture both high-semantic and high-resolution information. The LP function calculates the weighted sum of multi-resolution point features through an attention mechanism with learnable queries, enabling the extraction of all available information. As a result, PointStack can effectively represent both global and local contexts, allowing the network to comprehend both the global structure and local shape details. PointStack outperforms various existing feature learning networks for shape classification and part segmentation on the ScanObjectNN and ShapeNetPart datasets, achieving 87.2% overall accuracy and instance mIoU.

Keywords: point cloud; feature learning; shape classification; part segmentation

check for updates

Citation: Wijaya, K.T.; Paek, D.-H.; Kong, S.-H. Advanced Feature Learning on Point Clouds Using Multi-Resolution Features and Learnable Pooling. *Remote Sens.* **2024**, *16*, 1835. https://doi.org/10.3390/rs16111835

Academic Editors: Felicia Norma Rebecca Teferle, Abdul Awal Md Nurunnabi, Meida Chen and Yan Xia

Received: 26 March 2024
Revised: 13 May 2024
Accepted: 19 May 2024
Published: 21 May 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The point cloud has become one of the most popular representations of 3D objects in recent years [1–3]. The ability of point cloud data to represent highly complex 3D objects with low memory requirements enables real-time 3D vision applications for resource-limited agents. This is contrary to voxel-based representations [4,5], where the memory requirement is cubically proportional to the spatial resolution. In addition, the point cloud is the native data representation of most 3D sensors; thus, performing 3D vision directly on point clouds minimizes the pre-processing complexity. The advantages indicate that the point cloud can be the prime data representation for fast and accurate neural networks for 3D vision.

Unfortunately, there are several challenges in applying the matured 2D deep learning-based feature learning techniques to the point cloud, such as the irregular and disordered nature of the point cloud. These issues are addressed in the pioneering works of Qi et al. [1] and Qi et al. [6], which present the multilayer perceptron-based (MLP-based) PointNet and PointNet++, respectively. In the PointNet++ framework, a sequence of keypoint sampling, neighborhood grouping, neighborhood-wise feature learning, and feature aggregation is repeated several times to produce high-semantic point features. The relatively simple framework of PointNet++ is widely used in the literature. For example,

PointMLP [7] enhances the framework by incorporating residual connections and constructs a 40-layer MLP-based network that achieves state-of-the-art classification performance on several datasets.

Despite the promising results, the final high-semantic point features of the PointNet++ framework lose granular information due to the repeated keypoint sampling, where each of the surviving point features at the deeper layer of the network represents a larger spatial volume in the point cloud. Moreover, the max pooling function that is used for feature aggregation may exacerbate the loss since it completely neglects information from non-maximum point features [8]. Such a compounded loss of information concerning granularity and non-maximum point features could substantially harm the ability of the point features to deliver local context information, such as the detailed shapes of objects in point clouds [9].

Considering the problem of loss of granular and non-maximum point features information, we present two hypotheses: (1) It is advantageous for the task-specific head to have access to point features from all levels of resolution. This enables the network to extract high semantic information while preserving the granularity to a certain degree. (2) A generalized pooling function that combines information from all point features could improve the representation capacity of the aggregated point features, since the loss of information from non-maximum point features is minimized.

Based on the hypotheses, we propose a novel MLP-based network for feature learning on point clouds, PointStack, with multi-resolution feature learning and learnable pooling (LP). PointStack collects point features from various resolutions that are already available in the multiple layers of PointNet++. The collected multi-resolution point features are then aggregated and fed to the task-specific head. Therefore, the task-specific head has access to both high-semantic and high-resolution point features. Moreover, PointStack utilizes the LP that is based on the multi-head attention (MHA) mechanism [10] with learnable queries for both single-resolution and multi-resolution feature aggregation. The LP is a permutation invariant and generalized pooling function that does not neglect information from non-maximum point features, but rather calculates the weighted sum of the multi-resolution point features according to their attention scores. Consequently, PointStack is capable of producing high-semantic point features with minimal loss of information concerning granularity and non-maximum point features, such that both global and local contexts of a point cloud can be effectively represented. As a result, the network head can comprehend the global structure well and distinguish fine shapes of a point cloud, enabling PointStack to advance the state of the art of feature learning from point clouds.

Specifically, we observe that PointStack exhibits strong performance compared to various existing feature learning networks on two popular tasks: shape classification that requires global context understanding and part segmentation that requires both global and local context understanding. On the shape classification task with the ScanObjectNN dataset, PointStack outperforms existing feature learning networks by 1.5% and 1.9% in terms of the overall and class mean accuracy, respectively. On the part segmentation task with the ShapeNetPart dataset, PointStack outperforms existing feature learning networks by 0.4% in terms of the instance mean intersection over union metric. The two results demonstrate the superiority of the proposed PointStack, not only for tasks that require global context, but also for tasks that require local context.

In a summary, our contributions are as follows:

- In the proposed PointStack, we employ a multi-resolution feature learning framework for point clouds. Leveraging point features of multiple resolutions provides both high-semantic and high-resolution point features to the task-specific heads. Therefore, the task-specific heads can obtain high-semantic information without substantially losing granularity.
- We propose a permutation invariant learnable pooling (LP) for point clouds as an advancement over the widely-used max pooling. LP is a generalized pooling method compared to max pooling, since it combines information from multi-resolution point

features through the multi-head attention mechanism, as opposed to only preserving the highest-valued features.
- We demonstrate that PointStack outperforms various existing feature learning networks for point clouds on two popular tasks that include shape classification on the ScanObjectNN dataset and part segmentation on the ShapeNetPart dataset.

The remainder of this paper is organized as follows. Section 2 discusses existing works that are related to feature learning. Section 3 describes the proposed PointStack in detail. Section 4 shows the experimental results with extensive discussions. Section 5 concludes this work with a summary.

2. Related Work

2.1. Feature Learning on Point Clouds

As shown in Table 1, modern feature learning neural networks for point cloud data have been conducted on classification tasks using the ModelNet40 and ScanObjectNN datasets, as well as part segmentation tasks using the ShapeNetPart dataset [1,6,7,11–27]. Most studies originate from the pioneering work on PointNet by Qi et al. [1]. In PointNet, a sequence of point-wise multilayer perceptron (MLP) blocks is applied to the raw point cloud to produce high-dimensional point features. The point features are then aggregated through the max pooling operation, which results in a fixed-length global feature vector. PointNet++ [6] refines PointNet by considering local structures of the point via sampling, grouping, and local group feature aggregation. First, a collection of keypoints is obtained through farthest-point sampling. Then, neighboring points around each keypoint are grouped, and a PointNet operation is applied to each group, resulting in a neighborhood-wise global feature vector for each keypoint.

Table 1. Comparison of various models on ModelNet40, ScanObjectNN, and ShapeNetPart. We show the overall accuracy (OA), class mean accuracy (mAcc), and instance mIoU (Inst. mIoU). The notation $x \pm y$ represents the mean and standard deviation of the results after multiple runs of training. Bold indicates the best performance in each column.

Model	Year	ModelNet40		ScanObjectNN		ShapeNetPart
		OA (%)	mAcc (%)	OA (%)	mAcc (%)	Inst. mIoU (%)
PointNet [1]	2017	89.2	86.0	68.2	63.4	83.7
PointNet++ [6]	2017	90.7	-	77.9	75.4	85.1
PointCNN [11]	2018	92.5	88.1	78.5	75.1	86.1
SpiderCNN [12]	2018	92.4	-	-	-	85.3
DGCNN [13]	2019	92.9	90.2	78.1	73.6	85.2
KPConv [14]	2019	92.9	-	-	-	86.4
MVTN [15]	2021	93.8	**92.2**	82.8	-	-
DRNet [16]	2021	93.1	-	80.3	78.0	86.4
GBNet [17]	2021	93.8	91.0	80.5	77.8	85.9
Simpleview [18]	2021	93.9	91.8	80.5	-	-
CurveNet [19]	2021	93.8	-	-	-	**86.8**
PointBERT [20]	2021	93.8	-	83.1	-	85.6
PRA-Net [21]	2021	93.7	91.2	82.1	79.1	86.3
PointMLP [7]	2022	**94.1**	91.5	85.4 ± 0.3	**83.9 ± 0.5**	86.1
Point-MAE [22]	2022	94.0	-	85.2	-	86.1
Point-TnT [23]	2022	92.6	-	83.5	81.0	-
DualMLP [24]	2023	93.7	-	**86.4**	-	-
IBT [25]	2023	93.6	91.0	82.8	80.0	86.2
APES [26]	2023	93.8	-	-	-	86.6
Point-LGMask [27]	2023	-	-	85.3	-	86.1

Since then, numerous research studies have been conducted to enable learning the fine-grained local geometric features of the point clouds. For example, Wang et al. [13] propose a method to learn the relationships between the points with graph-based EdgeConv. Wu et al. [28] introduce a convolution-based network that learns the appropriate convolution kernel via MLP networks and kernel density estimation. Hamdi et al. [15] present a multi-view approach, where the network regresses the optimal viewpoint of the objects for 3D recognition. Ma et al. [7] introduce PointMLP, a relatively deep MLP-based network for point clouds. The network is based on the original PointNet++ with additional residual connections and geometric affine modules. Owing to the residual connections, PointMLP manages to comprise deep layers, where the best-performing variant is composed of 40 layers. Recently, several advancements have been made in point cloud feature learning. DualMLP, introduced by Paul et al. [24], extends the architecture of PointMLP to address the trade-off between the number of input points and computational cost. It consists of a SparseNet for processing a small subset of points and a DenseNet for handling a larger number of points, effectively balancing computational efficiency and scene understanding while achieving improved performance on the ScanObjectNN dataset compared to PointMLP. Li et al. [25] proposed a novel Inductive Bias-aided Transformer (IBT) method to tackle the challenge of discovering inter-point connections for efficient high-dimensional feature extraction in point cloud processing. IBT learns 3D inter-point relations by considering both local and global attentions.

In contrast to point cloud networks, point cloud sampling remains less explored, with random sampling and farthest point sampling being the most common methods. Wu et al. [26] propose a non-generative Attention-based Point Cloud Edge Sampling method (APES) that captures salient points in the point cloud outline and demonstrates superior performance on common benchmark tasks. Furthermore, few-shot learning for point clouds has been explored. Tang et al. [27] introduce Point-LGMask, a novel self-supervised learning method for 3D point clouds that embeds both local and global contexts using multi-ratio masking and a compound loss function. Point-LGMask outperforms existing pre-training methods on few-shot classification tasks.

2.2. Deep Learning with Multi-Resolution Features

Multi-resolution features have been extensively explored in image-based computer vision. Various traditional image processing techniques, such as the ones introduced by Dalal and Triggs [29] and Lowe [30], utilize a feature pyramid that leverages features of various resolutions (scales) from multiple layers for the downstream task prediction. The feature pyramid framework is still widely used in deep learning, especially after the introduction of the Feature Pyramid Network (FPN) by Lin et al. [31]. In FPN, feature maps of multiple resolutions are downsampled or upsampled to match the output feature map size and concatenated together, resulting in an output feature map with both high-resolution and high-semantic information.

Recently, various deep learning-based studies utilizing multi-resolution features have been proposed. Guo et al. [32] proposed an Atrous Spatial Pyramid Pooling (ASPP) module that leverages multi-scale context, significantly improving semantic segmentation performance. Additionally, Zhao et al. [33] introduced the Pyramid Scene Parsing Network (PSPNet), which effectively integrates context information at various scales by utilizing multi-scale representations. Moreover, research on multi-resolution features combined with self-attention mechanisms has been actively conducted in recent years. Wang et al. [34] proposed a method that integrates information at different scales along the height and width directions of an image using axial attention. Furthermore, Liu et al. [35] introduced a cross-shaped window multi-head self-attention approach that considers both global and local contexts.

In the point cloud domain, some recent works have started to explore the potential of multi-resolution features and self-attention mechanisms. For instance, Hui et al. [36] propose a transformer-based feature extractor that learns multi-scale feature maps for

large-scale place recognition. This work demonstrates the effectiveness of incorporating multi-resolution features in point cloud processing tasks. However, compared to the extensive research and advancements in image-based computer vision, the application of multi-resolution features in the point cloud domain is still in its early stages. While the success of these techniques in image-based tasks has been well-established, their full potential in point cloud processing has not yet been thoroughly explored.

3. PointStack: Multi-Resolution Feature Learning with Learnable Pooling

In this section, we first introduce an overview of the proposed multi-resolution feature learning implemented on a deep MLP-based network, PointStack. Following the overview, we introduce learnable pooling that is permutation invariant.

3.1. Multi-Resolution Feature Learning

The concept of multi-resolution feature learning is widely used for various downstream tasks in computer vision [31,37,38]. The principal approach is to construct a feature pyramid from semantic features from all levels of resolution. As a result, the feature pyramid has both high-semantic and high-resolution information that is often needed to recognize objects of various scales.

In the 3D point cloud domain, the potential benefits of utilizing multi-resolution features arise from the fact that 3D shapes are significantly more complex compared to 2D images. Important textures or curves of the 3D shapes may be only observable at the highest level of granularity. As constructing high-semantic features comes at a cost of losing granularity in the existing approaches, the finer details of the 3D shapes may be obscured. Therefore, multi-resolution point features can be a solution for both collecting sufficient semantic information and preserving granularity to a certain degree.

Unlike PyramNet [39], which uses multiple convolutions with different kernel sizes on the same point features to create a multi-scale feature map, we opt to leverage multiple point features from m different resolutions that are already available in the existing MLP-based networks, as shown in Figure 1. PointStack first learns the underlying representations of the points with the m repeated residual blocks, where the output of each block has lower resolution but higher semantic information compared to the corresponding input. We use the residual blocks instead of the transformer blocks as in Hui et al. [36], since residual blocks are more efficient in terms of the memory requirements. This is because the self-attention mechanism in each of the transformer blocks has $\mathcal{O}(n^2)$ memory complexity with respect to the input size n.

After learning the appropriate representations, PointStack performs single-resolution pooling on each output point feature, as shown in the bottom-left dashed box in Figure 1. That is, from each output, PointStack pools point features (PF_i of N_i feature vectors) at the i-th layer to produce PF_i^{pooled} of a fixed length N_m, where PF^{pooled} contains important features on the specific level of resolution.

Following single-resolution pooling, PointStack concatenates all PF_i^{pooled} to form and process the stacked, pooled point features, stacked-PF^{pooled}, through multi-resolution pooling (top-right dashed box in Figure 1) to produce a global feature vector. Since the global feature vector is obtained from the features of m resolutions, it contains information from both high-semantic and high-resolution features. Therefore, the task-specific heads have access to high-semantic information with minimal loss of granularity.

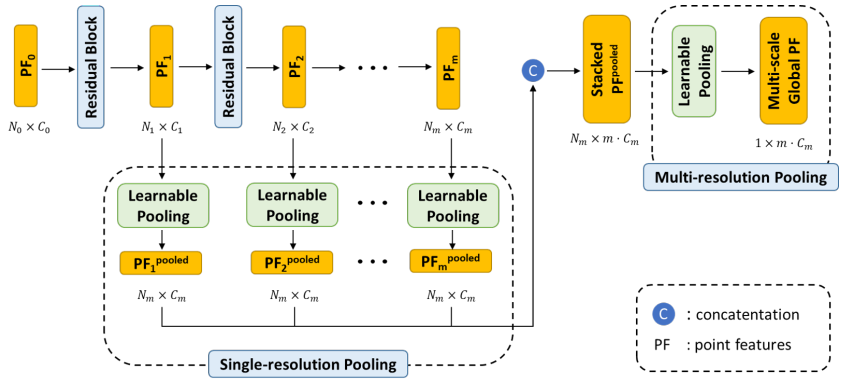

Figure 1. Feature learning backbone of PointStack. The residual block (one stage of PointMLP, Ma et al. [7]) learns the underlying representation of the input point features and outputs point features of a reduced length. For m repeated residual blocks, the output point features of each block are pooled by learnable pooling (LP) and concatenated to form a stacked point feature. The final LP is then applied to obtain the multi-resolution features, which can be used by the network heads.

Note that the multi-resolution feature learning framework can be realized without fixing the length of the output features of the single-resolution pooling. However, we find empirically that the fixed-length single-resolution pooling substantially improves the classification performance. Such a phenomenon may originate from the fact that point features of m different resolutions have different numbers of entries. That is, the highest-resolution point feature, PF_1, has significantly more feature vectors compared to the lowest-resolution point feature, PF_m. The disparity between the number of feature vectors may adversely affect the multi-resolution LP. Therefore, we incorporate the single-resolution pooling process to produce the same number of feature vectors from m resolution levels. This explanation is supported by the experimental results shown with the ablation study in Section 4.

3.2. Learnable Pooling

Recent works on feature learning from point clouds often utilize pooling functions. The pooling function is an important trick to produce fixed-length global features from input points of an arbitrary size. Since a 3D shape can be represented by the same set of points in a different order, the pooling function should be permutation invariant. A natural choice for such requirements is the max pooling function. Unfortunately, the max pooling function only preserves the highest-valued point features and completely neglects non-maximum point features, which results in a substantial amount of information loss.

To prevent this problem, we propose a generalized pooling function—learnable pooling (LP)—that aggregates by calculating the weighted sum of all point features according to the correlation between the point features and learnable parameters. Since the LP does not neglect information from non-maximum point features, it can be used for aggregating both single-resolution and multi-resolution point features without loss of information.

Structurally, LP utilizes the multi-head attention (MHA) [10] mechanism that can be seen as an information retrieval process, where a set of queries is used to retrieve information from the values based on the correlation between the queries and the keys. We set both keys and values to originate from the same point feature tensor, while the queries are learnable parameters. In this setting, we can consider that the network can learn the appropriate queries so that the retrieved point features (values) are highly relevant to the learning objectives. As the queries are directly supervised by the learning objectives, and the values are obtained through the weighted sum of all point features, the proposed LP is capable of producing representative aggregated point features with minimal loss of information compared to the max pooling function that completely neglects non-maximum point features.

The structure of the proposed LP is shown in Figure 2. The module architecture of the proposed LP is inspired by the Pooling by Multihead Attention (PMA) module introduced by Lee et al. [40], but ours is designed as a more compact form. That is, we only utilize linear transforms to match the channel size of the input point features to the desired output channel size and the multi-head attention mechanism. Note that, in this setting, the LP is a symmetric function so that the function is permutation invariant to the points of the point cloud.

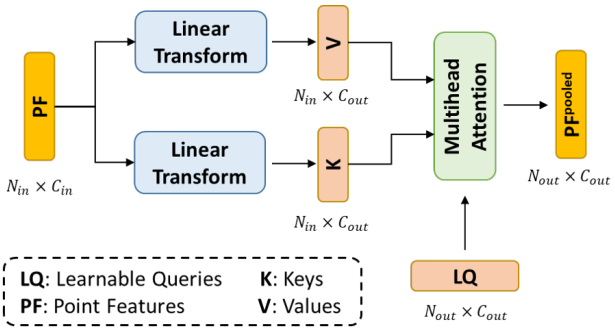

Figure 2. The structure of the learnable pooling (LP) module. Given an input of point features, LP transforms the features such that the channel size of the features match the channel size of the learnable queries. A multi-head attention (MHA) mechanism is then used to produce the fixed-length pooled point features. For the MHA, we set the input point features as the source of the keys and values, and the learnable queries as the queries.

Property 1. *The proposed learnable pooling is a symmetric function that is invariant to the permutation of the points of the point cloud. The proof can be found in Appendix A.*

The key to the permutation invariant property of the LP is the use of the point-wise shared MLP and the fact that both keys and values originate from the same row-permuted feature matrix. Since both keys and values are row-permuted by the same permutation matrix, and since the permutation matrices are orthogonal, the scaled dot-product attention mechanism becomes permutation invariant. In addition to the theoretical proof in Appendix A, we also show the empirical results in Section 4 to demonstrate the similarity between the standard deviations of the PointStack with LP and PointStack with max pooling outputs for various permutations of the input points.

4. Experiment and Discussion

In this section, we describe the dataset, network details, and training setup used for the experiments. Then, we show and discuss the experimental results.

4.1. Implementation Details

4.1.1. Dataset

We evaluate the proposed PointStack on two tasks, 3D shape classification and point-wise part segmentation, with three different datasets: ModelNet40 [41], ScanObjectNN [42], and ShapeNetPart [43]. We chose the two tasks because they represent the two extremes of the downstream tasks widely studied for point cloud data. That is, classification requires learning the global context of the overall point cloud, while segmentation additionally requires learning the local context of each point. In the following experiments, the number of input points is set to 1024 for the classification task and 2048 for the segmentation task. Note that the hardest variant of ScanObjectNN (PB_T50_RS) is used in the experiments, where objects are perturbed with translation, rotation, and scale transformations.

4.1.2. Network

For all experiments, we employ four residual blocks for the feature learning backbone of the PointStack, followed by an additional task-specific head. We set the hyperparameters

for the residual blocks according to Ma et al. [7]. The task-specific head is made up of only MLP blocks, where each block consists of an affine transformation, batch normalization [44], ReLU non-linearity, and dropout [45] layers. Each head has a final affine transformation layer to match the shape of the output tensors with the task-specific requirements. For learnable pooling, we set the size of learnable queries to 64×1024 and 1×4096 in single-resolution pooling and multi-resolution pooling, respectively. Since there are four residual blocks, we use four separate learnable queries for the four levels of resolution in the single-resolution pooling.

4.1.3. Training Setup

We train the networks using the PyTorch library [46] on RTX 3090 GPUs. Networks are optimized using the SGD optimizer with a cosine annealing scheduler [47] without the warm restart. The initial learning rate and minimum learning rate are set to be 0.01 and 0.0001, respectively, and we incorporate label smoothing [48] to the cross-entropy loss. We perform data augmentation by applying random translation to all datasets and random rotation to the ScanObjectNN dataset. For the shape classification task on ModelNet40 and ScanObjectNN, we set the maximum epochs to 300 and 200, respectively, and the batch size to 48. For the part segmentation task on ShapeNetPart, we set the batch size to 24 and the maximum epoch to 400.

4.2. Shape Classification

We evaluate the proposed PointStack on the shape classification task with ModelNet40 and ScanObjectNN datasets. ModelNet40 is a synthetic dataset of 40 different shape categories in the 12,311 point clouds sampled from computer-aided design (CAD) meshes. ScanObjectNN, on the other hand, acquires point clouds from real-world object scans; thus, the samples contain background points and occlusions. There are about 15,000 point clouds of 15 different shape categories.

Experimental results in Table 2 show that PointStack outperforms the previous MLP-based network, PointMLP [7], on the real-world dataset (i.e., the ScanObjectNN dataset) by 1.5% and 1.9% for the mean OA and mAcc, respectively. PointStack also outperforms other existing works, such as the multiview projection-based MVTN [15], by 4.1%, and the transformer-based Point-TnT [23] by 3.4%. Note that PointStack reduces the gap between the OA and mAcc performance, proving that PointStack is less biased towards certain classes compared to existing works. The shape classification results prove that minimizing the loss of information concerning granularity and non-maximum point features through multi-resolution feature learning and LP is beneficial for tasks that rely on the global context of the point cloud.

Table 2. Comparison of various models and PointStack on ModelNet40, ScanObjectNN, and ShapeNetPart. We show the overall accuracy (OA), class mean accuracy (mAcc), and instance mIoU (Inst. mIoU). The notation $x \pm y$ represents the mean and standard deviation of the results after multiple runs of training. Bold indicates the best performance in each column.

Model	ModelNet40		ScanObjectNN		ShapeNetPart
	OA (%)	**mAcc (%)**	**OA (%)**	**mAcc (%)**	**Inst. mIoU (%)**
PointMLP [7]	**94.1**	91.5	85.4 ± 0.3	83.9 ± 0.5	86.1
MVTN [15]	93.8	**92.2**	82.8	-	-
Point-TnT [23]	92.6	-	83.5	81.0	-
CurveNet [19]	93.8	-	-	-	86.8
PointStack	93.3	89.6	**86.9 ± 0.3** best = 87.2	**85.8 ± 0.3** best = 86.2	**87.2**

We see that the overall accuracy performance of PointStack on the synthetic dataset (i.e., ModelNet40) stands competitively at 93.3%, which is not superior to existing works. We speculate that the underlying cause of this issue is the significantly smaller number of training samples available in ModelNet40. To support this speculation, we train PointStack and PointMLP on a small subset of the ScanObjectNN dataset, which we discuss in more detail in Section 4.6. Classification results on ModelNet40 are shown in Figure 3.

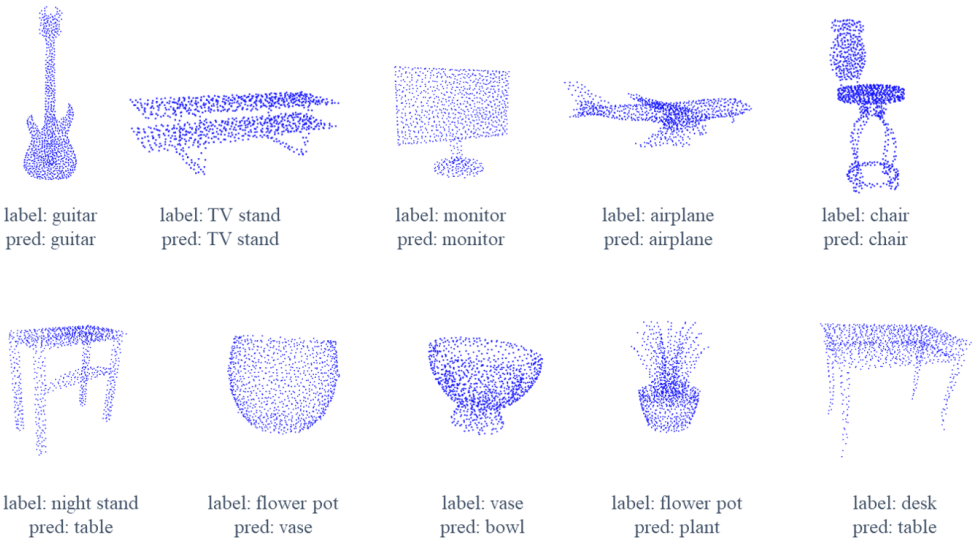

Figure 3. Visualization of classification results for PointStack on ModelNet40. As shown on the top row, PointStack demonstrates excellent 3D shape understanding across a diverse set of objects. We also show cases where PointStack outputs predictions that are different from the labels on the bottom row. We can see that failure cases occur when the labels are inherently ambiguous, but PointStack still manages to output predictions that are semantically similar to the labels, for example, desk vs. table, flower pot vs. plant.

4.3. Part Segmentation

We evaluate the proposed PointStack on the part segmentation task with the ShapeNet-Part dataset, a synthetic dataset derived from the ShapeNet dataset. It contains 16,881 pre-aligned point cloud shapes that can be categorized into 16 shape classes and a total of 50 segmentation classes.

From experimental results shown in Table 2, we observe that PointStack outperforms existing feature learning networks by at least 0.4%. Note that PointStack achieves such a high performance without using the voting strategy used by Xiang et al. [19], where each input point cloud is randomly scaled multiple times, and the predicted logits are averaged to produce the final class prediction. It is worth noting that PointStack achieves such performance with a simple MLP-based network, and the best performance of an existing MLP-based network [7] is 1.1% lower than PointStack. The part segmentation result, especially the significant improvement from the existing MLP-based network, testifies that minimizing the loss of information concerning granularity and non-maximum point features is crucial for tasks that require both global and local contexts. We visualize the part segmentation results in Figure 4 to demonstrate the high performance of PointStack.

We conduct additional experiments on semantic scene segmentation using the S3DIS dataset [49], which is essential for a comprehensive evaluation of our proposed method's potential. Both qualitative and quantitative results are provided in Appendix B.

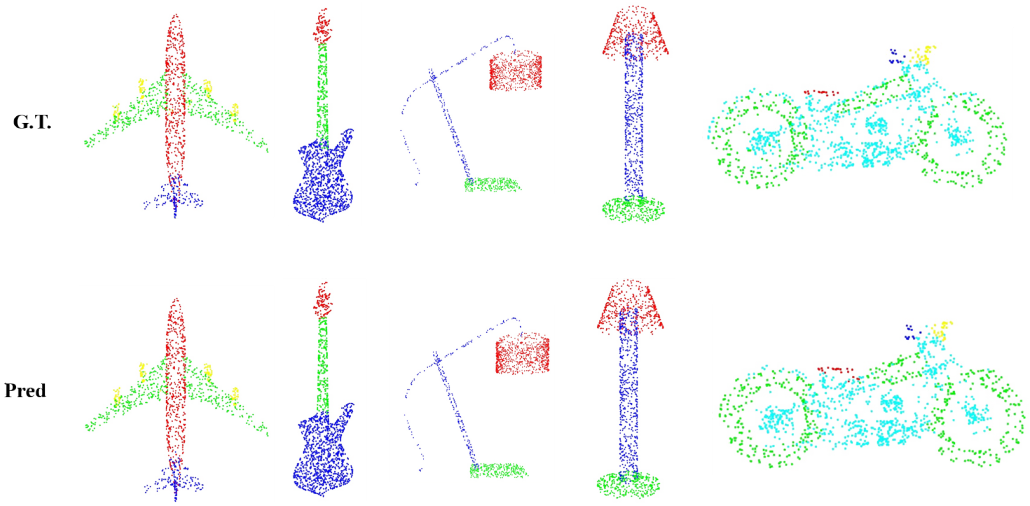

Figure 4. Visualization of the PointStack part segmentation ground truths (G.T.) and predictions (Pred). Qualitatively, the predictions are nearly identical to the ground truths.

4.4. Ablation Study

We conduct an ablation study with the ScanObjectNN dataset to investigate the effect of three major components of PointStack on the classification performance. The three major components are multi-resolution features, LP-based single-resolution pooling, and LP-based multi-resolution pooling.

First, we investigate the effect of multi-resolution features. We apply the max pooling function to the point features of four different resolution levels, which results in four single-resolution global feature vectors. As in PointStack, we concatenate the four single-resolution global feature vectors and then apply another max pooling operation, which produces the multi-resolution global feature vector. From Table 3, we observe that incorporating multi-resolution features improves the OA and mAcc performances by 0.4% and 0.5%, respectively. This result proves that preserving granularity through multi-resolution feature learning is beneficial for the classification performance of the network.

Table 3. Ablation study on the PointStack major components on the ScanObjectNN dataset. The notation $x \pm y$ represents the mean and standard deviation of the results after several runs of training. Bold indicates the best performance in each column.

Multi-Resolution Features	Single-Resolution LPs	Multi-Resolution LP	OA (%)	mAcc (%)
-	-	-	85.4 ± 0.3	83.9 ± 0.5
✓	-	-	85.8 ± 0.1	84.4 ± 0.1
✓	✓	-	86.5 ± 0.4	85.2 ± 0.2
✓	✓	✓	$\mathbf{86.9 \pm 0.3}$	$\mathbf{85.8 \pm 0.3}$
✓	-	✓	86.0 ± 0.7	84.9 ± 0.4

Second, we examine the effect of learnable pooling (LP). We replace the max pooling function in the single-resolution pooling process with LP. When LP is used for pooling feature vectors from each level of the four resolutions, the OA and mAcc scores are further improved by 0.7% and 0.8%, respectively. Subsequently, when LP is used for multi-resolution pooling, PointStack gains an additional 0.4% and 0.6% performance improvements for OA and mAcc, respectively. This result demonstrates that appropriately utilizing information from all point features in both single-resolution and multi-resolution poolings are crucial for producing relevant representations that benefit the classification performance of the network.

Additionally, we emphasize the importance of the single-resolution LP process. As mentioned in Section 3, the single-resolution LP enables PointStack to pool an equal

number of feature vectors from each level of the m resolutions. In Table 3, the performance of PointStack without single-resolution LP becomes 0.9% lower for both OA and mAcc, in addition to the higher variance. This result indicates that standardizing the number of feature vectors from each level of the m resolutions is indeed crucial for the multi-resolution LP to achieve high performance.

4.5. Permutation Invariant Property of Learnable Pooling

As mentioned in Section 3, the pooling function for any point cloud feature learning network should be permutation invariant. That is, the pooling function should be capable of producing the same output even if the order of the input points is changed.

To evaluate the permutation invariant property of learnable pooling, we compare two variants of the PointStack: one with max pooling and another with the proposed learnable pooling. Specifically, we train the two variants and evaluate the standard deviation of the OA for 10 random permutations of the input points.

From Table 4, we see that the network with learnable pooling has a similar standard deviation to the network with max pooling, where the standard deviation difference is only 0.04%. As the standard deviations are both small and similar, we confirm that the learnable pooling has a permutation invariant property similar to max pooling.

Table 4. Comparison of the standard deviation of the OA (σ OA) for shape classification on the ScanObjectNN dataset. We use 10 random permutations to calculate the σ OA values.

Pooling Function	**σ OA (%)**
Max Pooling	0.22
Learnable Pooling	0.26

4.6. Limitations on the Number of Training Samples

We observe that, although PointStack achieves state-of-the-art performance on the ScanObjectNN dataset, it does not outperform existing works on the ModelNet40 dataset. From this observation, we speculate that a potential cause of lower performance of the PointStack may be the insufficient training samples available in the ModelNet40 dataset. The ModelNet40 dataset has 9843 point clouds for training 40 different classes. In comparison, the main-PB_T50_RS variant of the ScanObjectNN dataset has over 11,000 point clouds for training just 15 classes.

To validate the necessity of a large number of training samples, we train PointStack and PointMLP (the state-of-the-art network for ModelNet40) on a small subset of the ScanObjectNN dataset. The subset is constructed such that the number of training samples for each class matches the average number of training samples for each class in ModelNet40. This translates to roughly 246 samples per class. During training, no augmentation method is applied. We note that we do not perform transfer learning from ScanObjectNN to ModelNet40. Instead, we simulate ModelNet40's lack of training samples using the ScanObjectNN dataset.

As shown in Table 5, the overall accuracy of PointStack is lower than the existing MLP-based network performance, PointMLP, when the number of training samples is insufficient. PointStack and PointMLP show 15.2% and 11.5%, respectively, lower performance than when they are trained with the full ScanObjectNN dataset. The results show the importance of sufficient training data size for PointStack to achieve a state-of-the-art performance. One possible explanation for such a requirement is that PointStack has a larger number of trainable parameters than the existing MLP-based networks due to multiple learnable pooling. However, we emphasize that PointStack still achieves a competitive performance when trained with a limited number of training samples and that modern datasets such as ScanObjectNN have sufficiently large training samples.

Table 5. Performance comparison on the ScanObjectNN dataset. OA_F and OA_S are the classification performances when trained on the full and subset ScanObjectNN dataset, respectively.

Model	OA_F (%) $\rightarrow$ OA_S (%)
PointMLP	$85.4 \pm 0.3 \rightarrow 73.9 \pm 0.3$
PointStack	$86.9 \pm 0.3 \rightarrow 71.7 \pm 0.3$

4.7. Runtime Performance Analysis

We observe that PointStack does not significantly increase the runtime compared to state-of-the-art models such as PointMLP [7]. Both multi-resolution feature learning and learnable pooling techniques are designed to be executed in parallel on the GPU, resulting in minimal impact on the overall computation time [10]. Regarding point cloud encoding of the ScanObjectNN dataset, PointMLP achieves an encoding speed of 112 samples per second on a Tesla V100 GPU [7], while PointStack processes 225 samples per second on an RTX3090 GPU. Considering that the RTX3090 has 1.62 times higher performance than the Tesla V100 [50], PointStack's ability to achieve more than double the encoding speed demonstrates that it does not introduce significant computational overhead. These results suggest that the proposed multi-resolution feature learning and learnable techniques can be efficiently incorporated into point cloud encoding architectures without compromising their time efficiency.

In addition, we evaluate the runtime performance of PointStack across various tasks. Specifically, we measure the point cloud encoding speed on the ModelNet40, ScanObjectNN, and ShapeNetPart datasets using an RTX3090 GPU. PointStack achieves encoding speeds of 222, 225, and 182 samples per second on these datasets, respectively. The results indicate that the part segmentation task (ShapeNetPart) requires additional runtime compared to the classification tasks (ModelNet40 and ScanObjectNN). This increased runtime can be attributed to the additional computation required for point-wise classification in the part segmentation task.

5. Limitations and Conclusions

5.1. Limitations

As demonstrated in Section 4, PointStack exhibits impressive performance, surpassing various existing feature learning networks for shape classification and part segmentation on the ScanObjectNN and ShapeNetPart datasets, achieving 87.2% overall accuracy and instance mIoU. However, it is important to note that PointStack requires a significant number of training parameters due to the learnable pooling operation, which introduces additional query parameters compared to other feature learning networks such as PointMLP [7]. While the proposed methods are efficient in terms of runtime, the increased parameter count can lead to a higher demand for GPU memory. To address this limitation and improve the practicality of PointStack for embedding applications, future work should focus on reducing the number of parameters without compromising the network's performance. This could involve exploring techniques such as parameter sharing, pruning, or more efficient attention mechanisms to minimize the memory footprint while maintaining the network's ability to capture both global and local contexts effectively.

5.2. Conclusions

Recent point cloud feature learning networks often use aggregated point features originating from the deepest layer when performing downstream tasks. The aggregated point features may contain high-semantic information, but there is a cost of losing information concerning granularity and non-maximum point features due to the sampling operation and max pooling, respectively. In this work, we have proposed a novel MLP-based feature learning network, PointStack, where the task-specific heads are given inputs of aggregated multi-resolution point features by the generalized pooling function called learnable pooling (LP). As a result, the aggregated point features could effectively represent both global

and local contexts and enable the network head to comprehend the global structure and local shape details of objects in the point cloud. Empirically, we observe that PointStack outperforms various existing feature learning networks for the shape classification and part segmentation tasks. In the future, it will be worthwhile to investigate the effectiveness of PointStack as the feature learning backbone network for other downstream tasks such as 3D object detection and shape completion.

Author Contributions: Conceptualization, K.T.W. and D.-H.P.; methodology, K.T.W. and D.-H.P.; software, K.T.W.; validation, K.T.W., D.-H.P. and S.-H.K.; formal analysis, D.-H.P.; investigation, D.-H.P.; resources, S.-H.K.; data curation, K.T.W.; writing—original draft preparation, K.T.W.; writing—review and editing, D.-H.P.; visualization, K.T.W.; supervision, S.-H.K.; project administration, S.-H.K.; funding acquisition, S.-H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by a National Research Foundation (NRF) of Korea grant funded by the Korea government (MSIT) (No. 2021R1A2C3008370).

Data Availability Statement: The original data (i.e., (1) ModelNet, (2) ScanObjectNN, and (3) ShapeNetPart) presented in the study are openly available at the following URLs: (1) https://modelnet.cs.princeton.edu/ (accessed on 13 May 2024), (2) https://hkust-vgd.github.io/scanobjectnn/ (accessed on 13 May 2024), and (3) https://shapenet.org/ (accessed on 13 May 2024), respectively. The code is available at https://github.com/kaist-avelab/PointStack (accessed on 13 May 2024).

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Proof for Property 1

Let F be the input point feature matrix to the learnable pooling function Ψ. Furthermore, suppose that Q, K, and V are the query, key, and value matrices, respectively, for a scaled dot-product attention mechanism Φ.

The learnable pooling Ψ can be formally defined as follows:

$$\Psi(Q, F) = \Phi(QW_q, FW_k, FW_v), \tag{A1}$$

where W_q, W_k, and W_v are the learnable weight matrices of linear transformations for the query, key, and value, respectively. Following the definition of the scaled dot-product attention mechanism [10], Equation (A1) becomes

$$\Phi(QW_q, FW_k, FW_v) = softmax\left(\frac{QW_q(FW_k)^T}{\sqrt{d_k}}\right)FW_v, \tag{A2}$$

where d_k is a scaling factor proportional to the feature dimension. Consider a case where F is row-permuted by a permutation matrix P. Then, the learnable pooling function becomes

$$\begin{aligned}\Psi(Q, PF) &= \Phi(QW_q, PFW_k, PFW_v)\\ &= softmax\left(\frac{QW_q(PFW_k)^T}{\sqrt{d_k}}\right)PFW_v.\end{aligned} \tag{A3}$$

Expanding the multiplications and considering that the permutation matrix does not scale the values such that performing the permutation before or after the softmax results in the same values, we obtain the following:

$$softmax\left(\frac{QW_q(PFW_k)^T}{\sqrt{d_k}}\right)PFW_v = softmax\left(\frac{QW_qW_k^TF^T}{\sqrt{d_k}}\right)P^TPFW_v. \tag{A4}$$

Since permutation matrices are orthogonal, i.e., $PP^T = P^TP = I$, where I is an identity matrix, Equation (A4) becomes

$$softmax\left(\frac{QW_qW_k^TF^T}{\sqrt{d_k}}\right)P^TPFW_v = softmax\left(\frac{QW_q(FW_k)^T}{\sqrt{d_k}}\right)FW_v. \quad \text{(A5)}$$

Since the right-hand side of Equation (A5) is equal to the right-hand side of Equation (A2), we prove that $\Psi(Q, F) = \Psi(Q, PF)$, and Property 1 in Section 3.2 holds.

Appendix B. Semantic Scene Segmentation on the S3DIS Dataset

Semantic scene segmentation aims to classify each point in a point cloud into predefined classes. Compared to the part segmentation task performed in Section 4.3, semantic scene segmentation is a more generalized and complex task due to the large number of points (approximately 1 million points in the S3DIS dataset [49]) and the requirement to classify points into 13 classes (e.g., doors, chairs, etc.). Therefore, we further validate the generalized local and global context encoding performance of PointStack on the widely used S3DIS dataset [49] for the semantic scene segmentation task. We train PointStack and PointMLP, which are used for distinguishing objects in indoor environments, using the same settings (70 epochs, SGD optimizer, 0.01 learning rate). The training results are shown in Figure A1. PointStack achieves 57.6% mIoU, a key metric for semantic scene segmentation, outperforming PointMLP by 1.5%. As depicted in Figure A1, both PointMLP and PointStack show classification results nearly identical to the ground truth for simple environments such as hallways (first row of Figure A1). Unlike PointMLP, PointStack robustly classifies objects such as boards and doors (depicted in green and orange colors in the second and third rows), even in complex environments such as offices and restrooms, demonstrating the effectiveness of the proposed LP and multi-resolution feature learning framework in semantic scene segmentation. The training code and weights are available at https://github.com/kaist-avelab/PointStack (accessed on 13 May 2024).

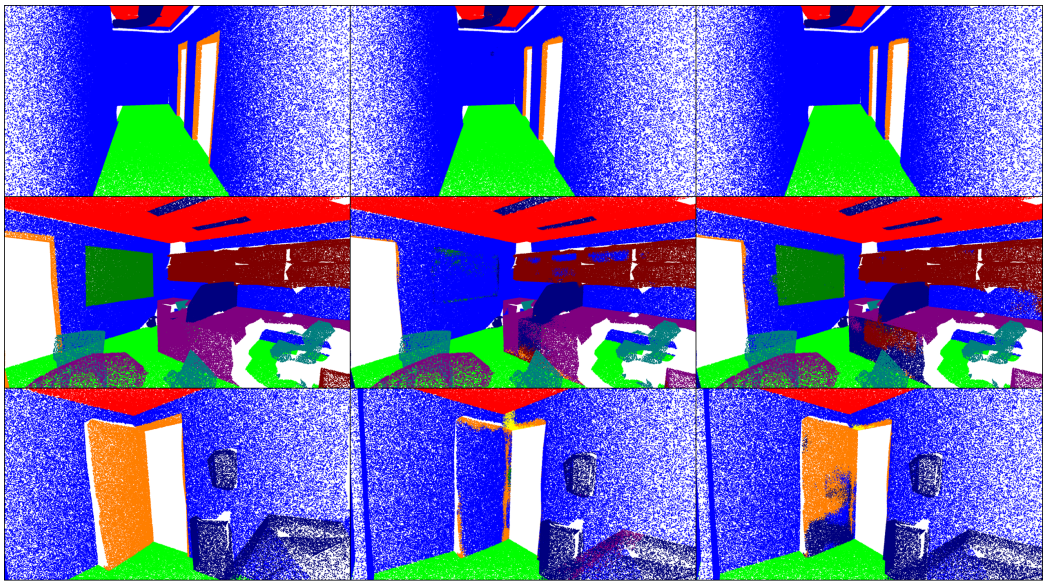

Figure A1. Visualization of (**a**) ground truth, (**b**) PointMLP, and (**c**) PointStack on the S3DIS dataset.

References

1. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
2. Yang, Z.; Sun, Y.; Liu, S.; Jia, J. 3dssd: Point-based 3d single stage object detector. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11040–11048.

3. Yu, X.; Rao, Y.; Wang, Z.; Liu, Z.; Lu, J.; Zhou, J. Pointr: Diverse point cloud completion with geometry-aware transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 12498–12507.
4. Graham, B.; Engelcke, M.; Van Der Maaten, L. 3d semantic segmentation with submanifold sparse convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9224–9232.
5. Yan, Y.; Mao, Y.; Li, B. Second: Sparsely embedded convolutional detection. *Sensors* **2018**, *18*, 3337. [CrossRef] [PubMed]
6. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
7. Ma, X.; Qin, C.; You, H.; Ran, H.; Fu, Y. Rethinking Network Design and Local Geometry in Point Cloud: A Simple Residual MLP Framework. *arXiv* **2022**, arXiv:2202.07123.
8. Yu, D.; Wang, H.; Chen, P.; Wei, Z. Mixed Pooling for Convolutional Neural Networks. In *Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2014; pp. 364–375. [CrossRef]
9. Zhang, X.; Sun, X.; Lian, Z. BoW Pooling: A Plug-and-Play Unit for Feature Aggregation of Point Clouds. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 2–9 February 2021; Volume 35, pp. 3403–3411. [CrossRef]
10. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
11. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2018; Volume 31.
12. Xu, Y.; Fan, T.; Xu, M.; Zeng, L.; Qiao, Y. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 87–102.
13. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph. (ToG)* **2019**, *38*, 1–12. [CrossRef]
14. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. Kpconv: Flexible and deformable convolution for point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 6411–6420.
15. Hamdi, A.; Giancola, S.; Ghanem, B. Mvtn: Multi-view transformation network for 3d shape recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 1–11.
16. Qiu, S.; Anwar, S.; Barnes, N. Dense-resolution network for point cloud classification and segmentation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2021; pp. 3813–3822.
17. Qiu, S.; Anwar, S.; Barnes, N. Geometric back-projection network for point cloud classification. *IEEE Trans. Multimed.* **2021**, *24*, 1943–1955. [CrossRef]
18. Goyal, A.; Law, H.; Liu, B.; Newell, A.; Deng, J. Revisiting Point Cloud Shape Classification with a Simple and Effective Baseline. In Proceedings of the International Conference on Machine Learning, Virtual Event, 18–24 July 2021.
19. Xiang, T.; Zhang, C.; Song, Y.; Yu, J.; Cai, W. Walk in the cloud: Learning curves for point clouds shape analysis. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 915–924.
20. Yu, X.; Tang, L.; Rao, Y.; Huang, T.; Zhou, J.; Lu, J. Point-BERT: Pre-training 3D Point Cloud Transformers with Masked Point Modeling. *arXiv* **2021**, arXiv:2111.14819.
21. Cheng, S.; Chen, X.; He, X.; Liu, Z.; Bai, X. Pra-net: Point relation-aware network for 3d point cloud analysis. *IEEE Trans. Image Process.* **2021**, *30*, 4436–4448. [CrossRef] [PubMed]
22. Pang, Y.; Wang, W.; Tay, F.E.; Liu, W.; Tian, Y.; Yuan, L. Masked Autoencoders for Point Cloud Self-supervised Learning. *arXiv* **2022**, arXiv:2203.06604.
23. Berg, A.; Oskarsson, M.; O'Connor, M. Points to Patches: Enabling the Use of Self-Attention for 3D Shape Recognition. *arXiv* **2022**, arXiv:2204.03957.
24. Paul, S.; Patterson, Z.; Bouguila, N. DualMLP: A two-stream fusion model for 3D point cloud classification. In *The Visual Computer*; Springer Nature: Berlin/Heidelberg, Germany, 2023. [CrossRef]
25. Li, Z.; Gao, P.; Yuan, H.; Wei, R.; Paul, M. Exploiting Inductive Bias in Transformer for Point Cloud Classification and Segmentation. In Proceedings of the 2023 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), Los Alamitos, CA, USA, 10–14 July 2023; pp. 140–145. [CrossRef]
26. Wu, C.; Zheng, J.; Pfrommer, J.; Beyerer, J. Attention-based point cloud edge sampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 5333–5343.
27. Tang, Y.; Li, X.; Xu, J.; Yu, Q.; Hu, L.; Hao, Y.; Chen, M. Point-LGMask: Local and Global Contexts Embedding for Point Cloud Pre-training with Multi-Ratio Masking. *IEEE Trans. Multimed.* 2023, *Early Access*. [CrossRef]
28. Wu, W.; Qi, Z.; Fuxin, L. Pointconv: Deep convolutional networks on 3d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9621–9630.
29. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
30. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]

31. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
32. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. Multi-scale context aggregation by dilated convolutions. *arXiv* **2018**, arXiv:1811.11922.
33. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid scene parsing network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2881–2890.
34. Wang, H.; Zhu, Y.; Green, B.; Adam, H.; Yuille, A.; Chen, L.C. Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 108–126.
35. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 10012–10022.
36. Hui, L.; Yang, H.; Cheng, M.; Xie, J.; Yang, J. Pyramid Point Cloud Transformer for Large-Scale Place Recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 6098–6107.
37. Ghiasi, G.; Lin, T.Y.; Le, Q.V. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7036–7045.
38. Kirillov, A.; Girshick, R.; He, K.; Dollár, P. Panoptic feature pyramid networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 6399–6408.
39. Zhiheng, K.; Ning, L. PyramNet: Point cloud pyramid attention network and graph embedding module for classification and segmentation. *arXiv* **2019**, arXiv:1906.03299.
40. Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A.; Choi, S.; Teh, Y.W. Set transformer: A framework for attention-based permutation-invariant neural networks. In Proceedings of the International Conference on Machine Learning. PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 3744–3753.
41. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
42. Uy, M.A.; Pham, Q.H.; Hua, B.S.; Nguyen, T.; Yeung, S.K. 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, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1588–1597.
43. Yi, L.; Kim, V.G.; Ceylan, D.; Shen, I.C.; Yan, M.; Su, H.; Lu, C.; Huang, Q.; Sheffer, A.; Guibas, L. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph. (ToG)* **2016**, *35*, 1–12. [CrossRef]
44. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning. PMLR, Lille, France, 7–9 July 2015; pp. 448–456.
45. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
46. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.
47. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
48. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
49. Armeni, I.; Sener, O.; Zamir, A.R.; Jiang, H.; Brilakis, I.; Fischer, M.; Savarese, S. 3D Semantic Parsing of Large-Scale Indoor Spaces. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1534–1543. [CrossRef]
50. Tesla V100 and RTX3090 Performance Comparison. Available online: https://www.techpowerup.com/gpu-specs/tesla-v100-pcie-16-gb.c2957 (accessed on 27 April 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.