applied sciences

MDPI

Article

PointSwin: Modeling Self-Attention with Shifted Window on Point Cloud

Cheng Jiang [1], Yuanxi Peng [1], Xuebin Tang [1], Chunchao Li [1] and Teng Li [2,3,*]

[1] The State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China
[2] Beijing Institute for Advanced Study, National University of Defense Technology, Beijing 100020, China
[3] College of Advanced Interdisciplinary Studies, National University of Defense Technology, Changsha 410073, China
* Correspondence: liteng09@nudt.edu.cn

Abstract: As a pioneering work that directly applies deep learning methods to raw point cloud data, PointNet has the advantages of fast convergence speed and high computational efficiency. However, its feature learning in local areas has a certain defect, which limits the expressive ability of the model. In order to enhance the feature representation in the local area, this paper proposes a new point cloud processing model, which is called PointSwin. By applying the Self-Attention with Shifted-Window mechanism to learn the correlation between mixed features and points, PointSwin encourages features to enhance their interactions with each other to achieve the effect of feature enhancement. At the same time, PointSwin also achieves a better balance between higher accuracy results and less time overhead by adopting the Mask mechanism to reduce redundant computations. In addition, this paper also proposes an efficient model called PointSwin-E. It can maintain good performance while greatly reducing the computational overhead. The results of the comparative experiments on ModelNet40 dataset show that PointSwin and PointSwin-E are better than PointNet and PointNet++ in terms of accuracy, and the effectiveness verification experiments on the Self-Attention with Shifted-Window mechanism also prove the superiority of this model.

Keywords: 3D point cloud processing; deep learning; self attention; classification

Citation: Jiang, C.; Peng, Y.; Tang, X.; Li, C.; Li, T. PointSwin: Modeling Self-Attention with Shifted Window on Point Cloud. *Appl. Sci.* **2022**, *12*, 12616. https://doi.org/10.3390/app122412616

Academic Editor: Byung-Gyu Kim

Received: 2 August 2022
Accepted: 2 December 2022
Published: 9 December 2022

1. Introduction

Point cloud data are the most popular data representation in the current 3D data representation domain, and it naturally possesses properties such as arrangement invariance and rigid transformation invariance [1]. The early point cloud data are usually converted into voxel data [2,3] and processed by deep learning methods such as Convolutional Neural Networks (CNN) [4,5] because of the non-standardization problem that its data storage form does not match the physical meaning. However, starting from PointNet [6] pioneering the use of deep learning methods to directly process raw point cloud data, more and more new methods have been extended and developed on the basis of this Pointwise MLP type of method [7] to pursue a better understanding of point cloud data.

The main body of PointNet [6] is the alignment network, the MultiLayer Perceptron (MLP) layer, and the MaxPooling operator. After correcting the input data through the alignment network, the MLP layer will combine the MaxPooling operator to approximate and fit the data distribution law. Finally, the MLP classifier is used to output the category score to predict the category to complete the classification work.

PointNet has excellent performance in learning global features, with high accuracy and fast computation speed. However, it has certain limitations in the ability to learn local features [8]. There is actually an overfitting phenomenon in the process of model training. That is to say, after the PointNet model reaches a certain accuracy rate in training, the accuracy rate on the test set will no longer improve. If the training continues, the network

will continue to fit the training set strongly, but such fitting does not learn the more general data distribution in local details.

Therefore, in order to better utilize the natural spatial location information of raw point cloud data, we hope to improve the feature extraction ability but control the computational cost of the model. The Self-Attention mechanism, which originated from the Transformer model [9], establishes the connection between elements by calculating the correlation between each element in the receptive field. This mechanism is known for its good perception of context information [10], and it just can complete the task goal of point cloud processing, which intends to establish connections between independent points. Considering the excellent gain effect of the Self-Attention mechanism on feature expression [11], the method in SwinTransformer [12] to achieve better performance with less computational overhead is more in line with our needs. SwinTransformer uses an advanced version of Self-Attention, MultiHead Self-Attention, which is calculated in the divided window to strengthen the connection between features. We use the Shifted-Window [12–14] method to move the calculation window to achieve the effect of window overlapping to reduce the amount of calculation.

We hope to use the new Self-Attention method proposed by SwinTransformer to help improve the problem of insufficient local detail learning ability of PointNet so as to achieve higher accuracy of point cloud classification task. Thus, we apply the Self-Attention with Shifted-Window mechanism to the point cloud feature map to form our model PointSwin.

The highlight of our method is the adoption of the Self-Attention with Shifted-Window mechanism to promote the interaction of local features with each other, enhance the feature learning ability, and maximize the advantages and benefits of the global features of PointNet.

The core of PointSwin is the backbone network and Self-Attention with Shifted-Window module. The whole PointSwin is constructed by the backbone network and Self-Attention with Shifted-Window part. In the backbone network, we follow the backbone network part of PointNet. That is, we continue to use the Joint Align Network to calibrate the input data, use MLP as the feature extraction operator, and use the MaxPooling operator as the global feature aggregation tool. In the Self-Attention with Shifted-Window section, we stack the Self-Attention with Shifted-Window modules and combine these modules with the downsampling action to form a hierarchical structure to enhance the interaction of local features. In the Self-Attention with Shifted-Window module, with the help of the Mask mechanism, MultiHead Self-Attention scores are calculated in the shifted window to improve computational efficiency.

Moreover, we have an efficient model named PointSwin-E. The model is also constructed by the backbone network and the Self-Attention with Shifted-Window part. The difference is that the Self-Attention with Shifted-Window part of PointSwin-E has only two parallel Self-Attention with Shifted-Window modules. By receiving the low-order MLP output feature map in the early stage of the backbone network, the Self-Attention with Shifted-Window module promotes the interaction of features with each other to achieve the effect of influencing the decision making of the model to build global features.

We conduct experiments on the ModelNet40 dataset and ShapeNetPart dataset, respectively, and compare them with other methods. PointSwin and PointSwin-E exhibit excellent overall accuracies of 90.88% and 98.39% on the ModelNet40 dataset and ShapeNetPart dataset, respectively. Experimental results show that our proposed PointSwin method outperforms PointNet as the baseline method on both the overall accuracy (OA) and the mean of accuracy on different categories (mACC). Compared with PointNet++ [15], which uses the farthest point sampling method (FPS) [16] combined with the Ball Query(BQ) [15] grouping method on the PointNet network to solve the problem of insufficient local feature capture ability, PointSwin shows higher OA data results.

We also conduct experiments to verify the effectiveness of the Self-Attention with Shifted-Window mechanism. The effectiveness of MultiHead Self-Attention, the input

data format, the hierachical architecture and the Feed-Forward Network [9] (FFN) in the Self-Attention with Shifted-Window part is demonstrated.

These experimental results fully demonstrate the superiority of the PointSwin model and also demonstrate the feasibility of the Self-Attention with Shifted-Window mechanism applied to point cloud data processing tasks.

In the future, we will also turn our attention to more point cloud processing tasks.

The main contributions of our work are as follows:

- We propose a new model, PointSwin, which effectively improves the prediction accuracy of point cloud classification tasks. In addition, we also propose an efficient model PointSwin-E, which can effectively reduce the computational cost while ensuring the prediction accuracy of the model.
- In our proposed model, the Self-Attention with Shifted-Window mechanism achieves excellent feature enhancement by increasing the effective interaction of features with each other. In addition, the Mask mechanism effectively helps the model to reduce the amount of redundant computation.
- In the classification task on the ModelNet40 dataset, the superior experimental results on PointNet and PointNet++ validate the effectiveness of our proposed model.

2. Related Work

2.1. Processing with Raw Points

Since PointNet [6] pioneered the use of the Pointwise MLP method, models that directly apply deep learning methods to raw point cloud data have emerged one after another. PointNet can easily complete tasks such as 3D object detection, part segmentation, scene semantic parsing, etc. using only the combination of MLP and Max Pooling [17]. However, it cannot capture the local feature information between points. Therefore, PointNet++ [15] proposes a hierarchical network to enhance the feature extraction capability of fine-grained regions by stacking multiple abstraction levels.

Point-BERT [18] performs self-supervised training by assigning the Masked Point Modeling task to the point cloud to improve the generalization performance of the model. PointMLP [19] further improves the robustness and performance of the model by introducing a very simple feedforward residual network combined with the MLP module. PointNext [20] optimizes the training and scaling strategy of PointNet++ [15] to realize the full potential of the model.

Convolutional Neural Network (CNN) [21,22] methods have also been applied in the field of point cloud understanding. PointCNN [23] integrates point information layer-by-layer by using a convolution-like structure. PointConv [24] directly uses the convolution operation to improve the recognition ability of the model after processing the coordinates and features of the point. KPConv [25] obtains point features with excellent representation ability by convolution operation on core points in a certain area. DGCNN [26] enhances feature representation by learning topological relationships between points to improve the accuracy of classification or segmentation tasks.

The above works have enhanced the learning ability of local region features from different perspectives. Our method enhances the expressive ability of local features from the perspective of enhancing the communication between features.

2.2. Self-Attention Mechanism

The Self-Attention mechanism originally originated in the field of NLP [10], and the emergence of the Transformer [9] model led to a new trend in the academic community. The two-dimensional image field was the first to innovate, and many methods began to use the Self-Attention method to replace the previous mainstream CNN method [27–30]. Then, the field of target detection also began to use the Self-Attention mechanism as the core feature extractor [15,31,32]. Afterward, researchers began to apply the Self-Attention mechanism to major processing tasks of point cloud data [10].

PointCloudTransformer (PCT) [33] completes the task of point cloud classification by obtaining the global context of the input and abandoning the downsampling operation. PATs [34] propose the Group Shuffle Attention (GSA) mechanism to replace the expensive overhead MultiHead Attention (MSA), which enhances the flow of information between multiple Heads by adding channel shuffling in MSA.

The models in the form of Local Transformer [35–39] strive to limit Self-Attention computation to local regions. LFT-Net [36] propose a trans-pooling model with four stacked transformer modules to strengthen the learning ability of fine-grained features. Pan et al. [37] adopt the Transformer blocks to extract global and local features and combine them for better 3D detection results. Lai et al. [39] convert the point cloud data into a voxelized grid style and perform local Self-Attention calculations in non-overlapping windows.

The models in the form of Point-Wise Transformer [35,40,41] aim to discover natural spatial relationships of point clouds. PointTransformer [35] combines the multi-scale idea with the Self-Attention method in the field of point cloud processing. It reduces the parameters in the Transformer hierarchy by downsampling the input step by step. Yu et al. [41] constructed a hierarchical point-wise structure for classification tasks and applied a local feature extraction module to complete the feature extraction work.

The models in the form of Channel-Wise Transformer [42–44] are dedicated to mining the similarities and associations between different dimensions. DT-Net [43] uses both Point-Wise Transformer and Channel-Wise Transformer for feature extraction and then splices the two for subsequent calculations. TCE [44] has combined both the coordinate channel and feature channel to generate a channel-wise transformer.

By using directed attention to learn points to achieve the effect of feature enhancement, the performance of point cloud downstream tasks can be improved [45,46].

The above works apply the Self-Attention mechanism to the processing tasks of point cloud data from the global or local perspective. Our work focuses on improving the perceptual ability of the model by applying the Self-Attention mechanism with the Shifted-Window.

3. Overall Architecture

We built two models, PointSwin and PointSwin-E. PointSwin-E is an efficient version of PointSwin, with faster model computation and less computational resource usage. Both models include two main modules. Our method has two main modules: one is the backbone network, and the other is Self-Attention with Shifted-Window module.

The backbone network module is built on the basis of the PointNet model structure. This module is designed to perform excellent feature extraction and feature aggregation works. Because of the advantage of the real-time performance of the PointNet network itself, this module can also quickly complete these tasks to balance the huge time overhead brought by the Self-Attention mechanism in the subsequent modules. We will introduce the backbone network in detail in Section 3.1.

The Self-Attention with Shifted-Window module is inspired by the SwinTransformer [12] network which the SOTA work has promoted on the model prediction capacity in an image classification field. This module aims to strengthen the interaction of local features with each other to compensate for the poor local neighborhood classification effect of the PointNet network. We will introduce Self-Attention with Shifted-Window module in detail in Section 3.2.

In PointSwin, the SASW module takes the output feature map in the middle of the backbone network as the starting input, and it completes the feature interaction work by combining with the downsampling operation and stacking four layers, as shown in Figure 1. The output features of this part are spliced with the original output features in the middle of the backbone network and re-input to the backbone network to complete the subsequent calculation.

However, in PointSwin-E, the two SASW modules will perform feature interaction on the output data of the two low-order hidden layers in the backbone network, respectively, and then splicing them with the output features in the middle of the backbone

network. Then, the spliced features are input back to the backbone network for subsequent calculations. We will introduce PointSwin-E in detail in Section 3.3.

The overall structure of PointSwin is shown in Figure 2.

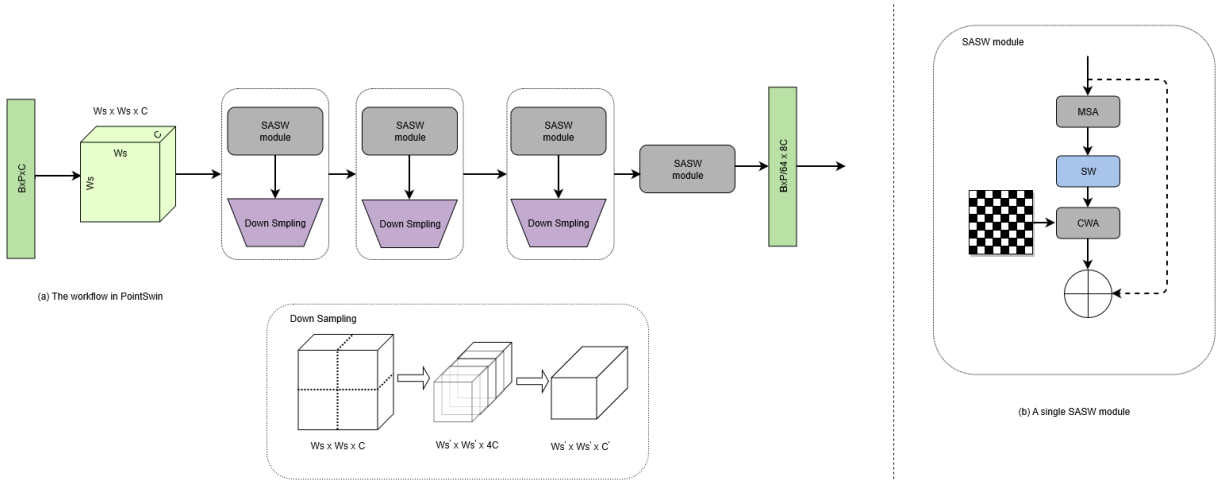

Figure 1. (**a**) The overall workflow diagram of the Self-Attention with Shifted-Window part in PointSwin; (**b**) The workflow of a single SASW block.

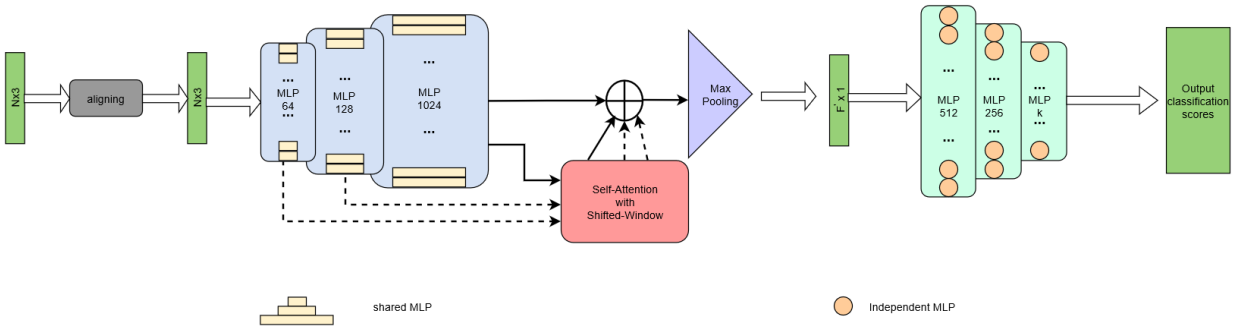

Figure 2. Illustration of the overall architecture of PointSwin and PointSwin-E. In the Self-Attention with Shifted-Window part, the part with a solid line in and out is combined with other parts as PointSwin. In the Self-Attention with Shifted-Window part, the part that enters and exits as a dotted line is combined with other parts as PointSwin-E.

3.1. Backbone Network

In order to efficiently obtain more representative global features in point cloud data, our method uses the main structure in PointNet's classification network as our backbone network. We continue to use a Joint Aligning network to align the input points for multiple spatial transformations in subsequent steps, use MultiLayer-Perceptron to extract features, and Use the MaxPooling operator to assemble the features into global features.

3.1.1. Joint Aligning Network

The point cloud is invariant to rigid transformations [47], but raw coordinate values do not. In the subsequent steps of the MLP layer to extract features of different dimensions, the model will perform various spatial transformations on the coordinate values. Therefore, the correction work should be completed on the original point cloud before the MLP layer. To directly apply the feature extract method on raw point cloud data, PointNet adapts the Spatial Transformer [48] method to align the input before extracting features. The Spatial Transformer method enables the baseline model to effectively perform spatial operations on the input data by inserting differentiable modules and has been validated on the 2D

image vision task. The Joint Aligning network takes the raw point cloud $U \in \mathbb{R}^{(B\times F\times P)}$ with batch B, coordinate F, and point P as input, and the set of subsequent shared MLP and MaxPooling layer as an affine transformation operator $\mathcal{T}_\theta()$. The output matrix will be a tiny-network, which can be written as:

$$\theta = \mathcal{T}_\theta(U) \tag{1}$$

3.1.2. MultiLayer Perceptron (MLP)

MultiLayer Perceptron is extremely advantageous in learning complex models. While extracting the features, MLP can extract features of different dimensions, levels, and granularities, and the boundary and edge features can be better obtained by mapping to high-dimensional space. As a set of classifier networks, MLP does a very good job of fitting curves. In PointNet, Charles et al. [6] proposed a new approximation theorem based on a Universal Approximate Theorem, proving that with enough nodes, the combination of MLP and MaxPooling can fit a continuous set function of arbitrary complexity as much as possible. Therefore, we still use a combination of MLP and MaxPooling to fit curves to learn global features.

In this part, we refer to the settings of PointNet and use two types of MLP. The shared MLP is used to extract different dimensional features, and the independent MLP is used to incorporate global features. The shared MLP has three hidden layers. It can be expressed as:

$$\left\{x_1^{k+1}, \ldots, x_i^{k+1}\right\} = concat\left\{F_1^k\left[x_1^k, \ldots, x_i^k\right], \ldots, F_j^k\left[x_1^k, \ldots, x_i^k\right]\right\} \tag{2}$$

where $k \in \{1,2,3\}$, $F_j^k \colon \mathbb{R}^{i\times j'} \mapsto \mathbb{R}^{i\times j}$ is a perceptron operator, i is the number of points involved in the calculation of the model, and each x_i^k has j' feature values.

The independent MLP also has three hidden layers. It can be expressed as:

$$Z^k = \left\{f_1^k, \ldots, f_j^k\right\} \tag{3}$$

$$Z^{k+1} = \left\{f_1^k, \ldots, f_{j'}^k\right\} = G\left\{Z^k\right\} \tag{4}$$

where $k \in \{1,2,3\}$, f_j^k represents the different dimensional features to be integrated, j represents the number of feature dimensions to be merged, and $G \colon \mathbb{R}^{1\times j'} \mapsto \mathbb{R}^{1\times j}$ is one layer perceptron network.

3.1.3. MaxPooling Operator

To ensure that the model can accept the arrangement-invariant nature of the point cloud data, PointNet proposes a method to integrate all point features using a symmetric function, and we followed this method. As a symmetric function, MaxPooling makes it easy to perform quick operations on the data as a whole. For acquiring global features, it is a very simple but brilliant step while working with the MLP layer.

In PointSwin, what the MaxPooling operator actually does is to compress the features $F \in \mathbb{R}^{(B\times J\times P)}$ with batch B, feature dimensions J and points P to the features $F' \in \mathbb{R}^{(B\times 1\times P)}$.

3.2. Self-Attention with Shifted-Window Module (SASW)

The Self-Attention with Shifted-Window module is inspired by the Shifted Window based Self-Attention module [12] in Swin Transformer, which achieves the SOTA performance in an image object detection field.

In PointSwin, as shown in Figure 1a, the input $F \in \mathbb{R}^{B\times P\times C}$ with Batch B, Points P and input Channel C, is transformed into $F' \in \mathbb{R}^{B\times W_s\times W_s\times C'}$ with window size W_s and the channel after transformation C', with fixed window division mode. Entering into the hierarchical structure, the SASW module is combined with downsampling to improve the influence of the attention mechanism in Cross-Window computations. In the computation

window, by superimposing the Shifted-Window method and the MultiHead Self-Attention mechanism, the interaction of local features is enhanced while ensuring that local features dynamically participate in the construction of global features. At the same time, PointSwin uses the Mask mechanism in the computation window to reduce redundant calculations in Cross-Window computaions. In addition, we also add Relative position bias to the window to help the Self-Attention mechanism output more meaningful attention scores.

3.2.1. MultiHead Self-Attention (MSA)

The Self-Attention mechanism enhances the feature expression by calculating the correlation to adapt to the natural location information characteristics of the point cloud data, which is of great benefit to improving the effectiveness of the model fitting network. However, the attention mechanism invariably needs huge calculations for support, particularly for processing the massive point cloud data.

To reduce direct calculations and increase computational efficiency, we choose the MultiHead Self-Attention method for relevant calculation, which is a method that is better than the Self-Attention method for lower needs of GPU Memory-Usage and richer correlation expression. Furthermore, the enhanced feature representation of the MSA module can make up for the lack of local feature learning in the backbone network to a certain extent, especially under the combined effect of the window mechanism. This module can be expressed as:

$$H_i = Softmax\left(\frac{Q_i K_i^T}{\sqrt{s_{qk-mh}}} + F\right)V \tag{5}$$

$$M_{SA} = concat(H_1, \ldots, H_i) \tag{6}$$

where i is the num of heads, F is the set of some other computing blocks such as the mask and relative position bias, s_{qk-mh} is the downscaling factor, $Q = concat\{Q_1, \ldots, Q_i\} \in \mathbb{R}^{i \times N_q \times \frac{C_q}{i}}$, $K^T = concat\begin{Bmatrix} K_1^T \\ \ldots \\ K_i^T \end{Bmatrix} \in \mathbb{R}^{i \times N_k \times \frac{C_k}{i}}$, $V \in \mathbb{R}^{i \times N_v \times \frac{C_v}{i}}$ are the query matrix, transpose of the key matrix, and the value matrix, respectively.

For more details, please refer to Section 4.3.

3.2.2. Shifted-Window (SW)

The window mechanism has obvious advantages in reducing computational overhead and enhancing local feature learning. We hope that by applying the Self-Attention mechanism limited in the window, we can enhance the local feature learning of point cloud and reduce the high computational cost caused by the Self-Attention mechanism. However, if the windows are independent of each other and do not overlap, there will be a lack of communication between the windows, and it will not be possible to ensure that the local features of the point cloud can dynamically participate in the construction of the global features.

In order to ensure close communication between windows to transfer feature correlation between windows, PointSwin adopts the Shifted-Window method. The Shifted-Window approach has been proven effective on MLP architectures, which means adapting to the backbone network [49]. The Shifted-Window method means that the window where any feature is located during a feedforward network operation is not fixed. The feature that has passed through the Shifted-Window method will be in two different windows at least before and after the operator operation, but the window size will not change. This means that any feature can perform the correlation calculation of the Self-Attention mechanism with different features in two different windows before and after the Shifted-Window method operation.

In order for the subsequent mask method to work properly, we set the shifting value to a fixed value, which is half of the original window size. In order to maximize the enhancement effect of the interaction between features on the feature representation, we

use the Shifted-Window method in PointSwin multiple times. As shown in Figure 1, each SASW module performs a Shifted-Window operation between the MSA and CWA operators to alternately use the original window and the shifted window for feature interaction with each other. That step can be expressed as:

$$Z^{l'} = G\left(Win_f\left(Z^l\right)\right) \tag{7}$$

$$Z^{l+1} = G'\left(Win_s\left(Z^{l'}\right)\right) \tag{8}$$

where $Z^l \in \mathbb{R}^{P\times F}$ and $Z^{l'} \in \mathbb{R}^{P\times F}$ denote the input feature map of fixed windows and shifted windows in l layer, Win_f and Win_s are the function to split the feature map into fixed or shifted windows, G is the fixed-window layer processing function which contains MSA, FFN, and another procedure step function, and G' is the shifted-window layer processing function which contains CWA, FFN, and another procedure step function.

3.2.3. Cross-Window Attention (CWA)

For the fixed window, we use the MSA module to calculate the inter-feature correlation to obtain the feature map after interacting with each other within the window [12]. While for the shifted window, we use the Cross-Window Attention module to process and calculate. The CWA module acts directly on the shifted window. Actually, its core correlation calculation has no difference from the MSA module, but its meaning and impact are quite different.

The CWA module not only allows the features in the new window to calculate the correlation between each other to enhance the learning of the local neighborhood of point cloud features, it also can correlate the attention scores of the full feature map with each other to ensure that the local features participate in the adaptive adjustment process of the global features.

In PointSwin's hierarchical structure of stacked SASW modules, the combination of CWA modules and downsampling actions further enhances the sufficiency of features interacting with each other. After the inter-layer downsampling operation, the features in the old window will be mixed into the new window for MSA and CWA operations of the new layer, making the features communicate with each other more abundantly. The CWA module makes the computation of correlations represented by self-attention mechanism computations on point clouds more meaningful.

3.2.4. Mask in Cross-Windows

The biggest advantage of the Self-Attention mechanism applied to the window in point cloud processing tasks is to allow local features to interact with each other as much as possible to learn more meaningful local features. The advantage of the mask mechanism is to ensure that the amount of redundant computation is reduced [12] under the condition that the interaction between features is still rich.

Setting the mask reasonably not only could help reduce the redundant computation but also help the model find more important and tight connections among features. In fact, we found that using the Mask method in the computation window of the shifted window brings both computational speed and prediction accuracy improvements to the model.

We place the mask method in the procedure of Self-Attention to the shifted window. After shifting the windows, we split all features into nine categories. We give each feature a token in order to facilitate the subsequent calculation to identify whether this feature needs to be included in or not.

The first step of Mask is to mask the computation on the diagonal line of the windows, as these computations have appeared in the previous module. The second step is to mask the computation among the less relevant features to put more attention on the features with more potential for correlation.

We give nine tokens from 0 to 8 for all features from left to right and top to bottom as the left of Figure 3 shows and set the shift size as half of the window size. Thus, after shifting the windows, the features with token 5 and token 3 will calculate Self-Attention scores in one window. Similarly, the features with token 7 and token 1 will be aggregated into one window, and it is also similar for cluster the features with token 8, 6, 2, and 0.

As the new window has the features with token 5 and token 3, we want the features with token 5 which from the different old windows communicate with the features with token 3, respectively, for they are different points and different feature dimensions. Furthermore, they were far away from the origin feature map, and we hope they can interact directly to uncover the potential for associations. However, the new window has the features with token 7 and 1; we let the features with token 7 interact with the feature with token 1 but limit the communication among the features with token 7 or 1 to reduce the insignificant computation. Additionally, for the new window including the features with tokens 8, 6, 2, and 0, we hope they interact with each other internally but do not calculate their own Self-Attention scores to calculate the relevance among the different old windows.

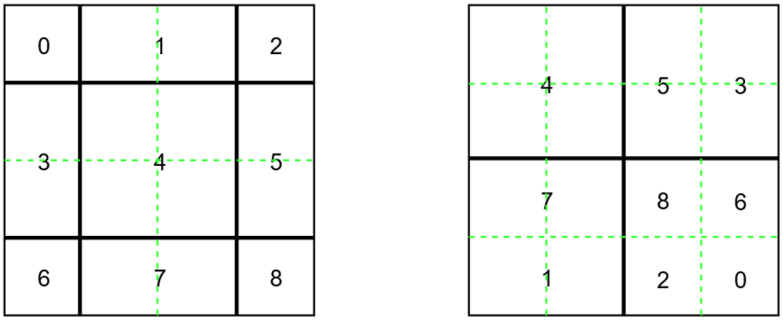

Figure 3. Illustration of window division and Mask token labeling of a small piece of data before and after window shifts. The left picture is a schematic diagram before the window shifts, and the right picture is a schematic diagram after the window shifts.

With the above design, we ensure that the repetitive computation has been masked to reduce redundant computing and avoid interfering with the Cross-Window attention score computations. More importantly, this mask mechanism selectively does calculations on parts that are more likely to be related, so that the model has a calculation selection bias on the basis of calculating the Attention score to obtain the degree of attention preference, which further enhances the effectiveness of the information interaction among features.

3.2.5. Relative Position Bias

In order to enhance the model's ability to perceive local areas, most methods usually add position information as a bias value that affects the judgment for the model to use. The absolute position bias does not apply to our model; thus, we utilize the relative position bias method to influence the model decisions.

In the procedure of obtaining a newly inspired value through using MSA for the features in the window, we follow [13,50–53] by adding the relative position biased to the step of the computing attention score. It can be expressed as:

$$\mathrm{MSA} = Softmax\left(\frac{QK^T}{\sqrt{s_{qk}}} + Bias + F\right)V \tag{9}$$

where F is the other computing module such as the Mask, and $Bias \in \mathbb{R}^{H \times w_s^2 \times w_s^2}$.

Since it is observed that computing the attention scores between features or points alone cannot achieve the desired improvement, we mix features and points as input. Therefore, in our design, we can abstract the connection between features into four cases. They are the SPDF category (same point with different feature dimension), the SPSF

category (same point with same feature dimension), the DPSF category (different point with same feature dimension), and the DPDF category (different point with different feature dimension), as shown in Figure 4.

Similarities and differences in point and feature dimensions mean different attention meanings. We give higher weight to the class DPSF and the class DPDF, for we tend to accept that feature interactions between different points are more significant to complement the local feature learning. We take less notice of the class SPSF and the class SPDF, for we believe that fewer needs for interest should be put on the self and the same point.

In addition, because of the difference in the relative relationship of each group of features and points in a single window, there is exactly one set of corresponding relationships; that is, the number of relative position deviation values at the spatial level between each value in the window is equal to the square of the window size value. To enhance the feature representation meaning of the relative position deviation value, we register this value as a learnable variable in the model.

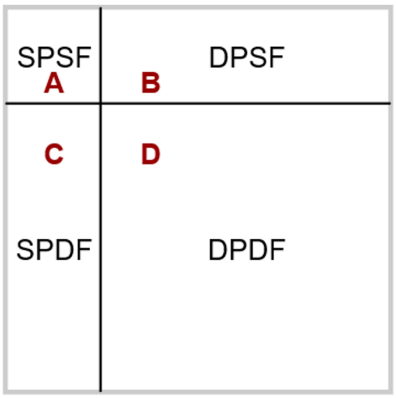

Figure 4. Illustration of the class of relative position bias. For block **A**, the self of block **A** is class SPSF, block **B** in the same row is class DPSF, block **C** in the same column is class SPDF and block **D** in the rest of window is class DPDF.

3.3. The Efficient Architecture (PointSwin-E)

In the course of our research, we also constructed an efficient version of the PointSwin model and named it PointSwin-E. PointSwin-E, like PointSwin, includes the backbone network and SASW modules. However, unlike PointSwin, PointSwin-E does not stack multiple SASW modules to form a hierarchical structure, and it does not combine SASW modules with downsampling operations.

PointSwin-E consists of two parallel SASW modules which, respectively, operate on the output features of the two hidden layers in the early stage of the backbone network. As shown in Figure 5, only one SW operation is performed in each SASW module. The output features of the two parallel SASW modules will be transmitted back to the intermediate stage of the backbone network for splicing with the output features of the previous MLP operator. The concatenated features are used as the global feature candidate pool to continue the MaxPooling operation and subsequent MLP operations until the feature aggregation work is completed. Therefore, it can be seen from the organization of the SASW module that PointSwin-E pays more attention to the time overhead and GPU Memory-Usage overhead.

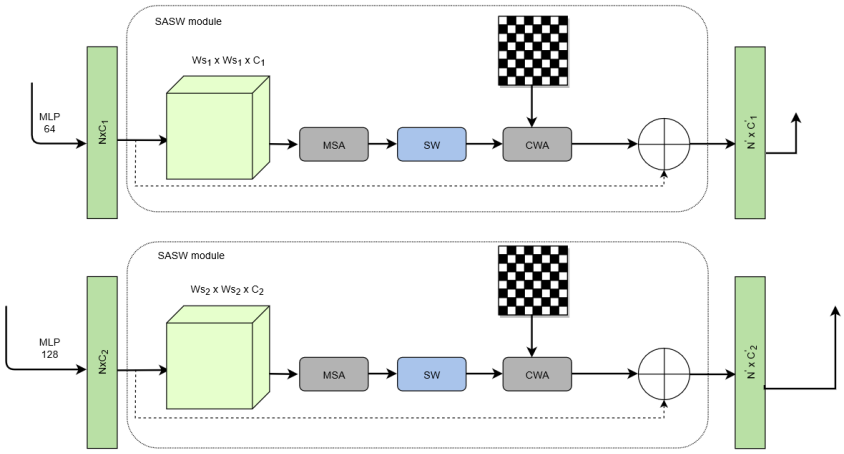

Figure 5. Illustration of the Self-Attention with Shifted-Window part of the PointSwin-E model. The upper and lower parts are two parallel SASW modules.

PointSwin-E still focuses on global features in feature extraction, and it only performs intra-feature interaction operations on a small number of feature maps. Its significance is mainly to influence the model's decision to construct global features by adding a small number of intra-feature interactions, and the influence is a positive effect. This significance proves the feasibility and effectiveness of the Sel-Attention with Shifted-Window mechanism in point cloud processing tasks. This feasibility shows that using the Self-Attention mechanism limited in the shifting window to calculate the correlation between point cloud features will not only bring higher prediction accuracy to the model but also reduce the high computational overhead brought by the Self-Attention mechanism to ensure the availability of the model.

PointSwin-E only performs feature interaction operations on the small-scale output feature maps of low-order functions of the backbone network, ensuring the efficient use of computing resources. PointSwin-E retains the concate link mode between the SASW output data and the backbone network in PointSwin, which ensures that local features can participate in the construction of global features and affect the final prediction decision of the model. Benefiting from such a delicate and compact network structure, the computational time overhead of PointSwin-E is much smaller than that of PointSwin. However, just as the old saying goes, every coin has two sides. The delicate and small network structure also means a lack of learning ability. This problem will become more apparent as the solution problem becomes more complex.

4. Experiments

In this section, we first introduce the classification results of PointSwin on two datasets of different scales and compare it to the other methods with the analysis. Then, by analyzing the extended experimental data of the PointSwin, we present the research findings that indicate the maximum utility of the Self-Attention with Shifted-Window mechanism. In addition, we conduct an ablation study for the PointSwin.

4.1. ModelNet40 Shape Classification

The ModelNet40 dataset [4] is the most commonly used public dataset for testing model performance in point cloud classification processing. It has over 10,000 CAD models for 40 object categories. We use the common version of ModelNet40 dataset which converts an off file to ply file and was sampling to 2048 points only with the coordinates of the point. We compare our model with eight classical methods.

As can be seen from Table 1, compared with PointNet, which is the baseline method, our method achieves better performance in both overall accuracy and mean accuracy on different categories by enhancing the interaction between features. Compared with the

PointNet++ model that adds farthest point sampling (FPS) and ball query (BQ) methods to the PointNet network to find local neighborhoods, our model does not use the method of obtaining the local order but just lets the features interact by themselves to determine the correlation between each other. Our model achieves better performance than PointNet++ in terms of OA value.

Table 1. Results of ModelNet40 shape classification.

Method	Input	mACC	OA	Params	Time-Consuming *
VoxNet [2]	voxel	83.0	85.9	-	-
Subvolume [54]	voxel	86.0	89.2	-	-
PointNet (vanilla)	point	-	87.2	-	-
PointNet	point	86.2	89.99	3.71 M	9 s
A-SCN [55]	point	87.6	90.0	-	-
MVCNN [56]	image	-	90.1	-	-
DeepSets [57]	point	-	90.3	-	-
PointNet++ (W/O norml)	point	-	90.7	-	-
PointSwin	point	86.9	90.88	3.85 M	120 s
PointSwin-E	point	86.39	90.8	4.21 M	21 s

* The data are the average time spent training each epoch under the NVIDIA TITAN RTX graphics card environment.

PointSwin improves the model's decision accuracy for global feature construction by enhancing the learning of local features, proving the feasibility of the correlation between features calculated by the MultiHead Self-Attention mechanism to be transferred between windows through Shifted-Window. PointSwin-E performs slightly worse than PointSwin on this dataset but also shows close performance. We believe that this is the problem of insufficient local feature learning caused by insufficient interaction features in the local neighborhood of PointSwin-E. Of course, compared to PointSwin, PointSwin-E shows its more efficient side, and the time cost of the latter is about one-sixth of the former.

According to the experimental values of OA and mACC, PointSwin shows better performance than PointNet and PointNet++ on this dataset, which is enough to prove the superiority of the SASW mechanism. This fully demonstrates the great potential of the Self-Attention with Shifted-Window mechanism. Through various experiments in Sections 4.3 and 4.4, we confirmed that the PointSwin model has reached the upper limit of expression capability on the ModelNet40 dataset of this ply version, because the number of sampling points and the sampling model are fixed.

4.2. Classification on ShapeNetPart Dataset

The ShapeNetPart dataset is a subset of the ShapeNetCore Dataset [58,59], which is mainly used for training and testing in the part segmentation task of point clouds. It contains about 16,000 models of 16 categories of objects. The number of objects in a class varies from 2 to 6, and the total number of objects in all classes is 50. Although this dataset was originally designed to serve part segmentation, a relatively small number of publicly available datasets in the point cloud domain were available in previous years, and this dataset was also used to help validate model performance in the shape classification domain. Therefore, in order to more fully compare the performance with the baseline method PointNet and control the influencing variables as much as possible at the same time, we also performed the accuracy test of the PointSwin model on this dataset.

As shown in Table 2, the efficient model can achieve better accuracy performance than PointNet. Compared with the original model PointSwin, the accuracy performance and computational efficiency of PointSwin-E are superior.

Table 2. Comparative experimental results of PointNet [6] and the two models of PointSwin on the ShapeNetPart dataset.

Method	mACC	OA	SASW Module Flops *	Time-Consuming **
PointNet	94.99	98.12	-	55 s
PointSwin	95.06	97.94	26.6 M	150 s
PointSwin-E	96.04	98.39	1.8 M	48 s

* The data are the core calculation amount of the SASW module in each batch when the BatchSize is set to 64. ** The data are the average time spent training each epoch under the NVIDIA TITAN RTX graphics card environment.

On this dataset, we observed that PointSwin showed lower prediction accuracy results than PointNet. We believe that this is due to the imbalance of the samples in the dataset, which causes the PointSwin model to rely too much on the feature opinions represented by local features when making decisions, so that the model falls into the predicament of local overfitting. This also inspired us to think about the future research direction. We believe that when applying the PointSwin model to other point cloud processing tasks in the future, it is necessary to consider the impact of sampling density and sample imbalance on the local Self-Attention mechanism to improve the generalization performance of the model.

Currently, we verify the improved classification performance of our proposed model on this dataset. In future work, we will generalize the model to solve segmentation problems and will continue to test and verify segmentation effects on the ShapeNetPart and ShapeNetCore datasets.

4.3. Efficiency of Shifted-Window with Self-Attention Mechanism

This section is focused on exploring the best performance of the Self-Attention with Shifted-Window mechanism. We empirically verify and analyze the validity of some key modules in the SASW mechanism by the control variable method. In order to unlock the full potential of this mechanism, we choose to conduct comparative experiments on the more well-distributed ModelNet40 dataset to minimize the injustice of the experiment.

4.3.1. Group 1: The MultiHead Self-Attention Mechanism

In the Self-Attention mechanism, we use the SingleHead Self-Attention method (SSA) and the MultiHead Self-Attention method (MSA) as filter carriers, respectively, and observe the difference between the accuracy results and the time consuming of the two methods. The experimental results are shown in Table 3.

Table 3. The OA and time results of group 1.

Method	OA	Time-Consuming *
PointNet	89.22	20 s
SingleHead Self-Attention method	89.46	142 s
MultiHead Self-Attention method	89.58	120 s

* The data are the average time spent training each epoch under the NVIDIA TITAN RTX graphics card environment.

The advantage of the MultiHead Self-Attention method is not only in parallel computing but also in the simultaneous consideration of multidimensional information, which is equivalent to promoting the interaction of features in local neighborhoods. Therefore, the MultiHead Self-Attention method combined with the shifting window mechanism is equivalent to having a party for the features in the local neighborhood, which will effectively promote the learning of local features.

In the experiments, we observed that the SSA method is more computationally time consuming and obtains lower accuracy results than the MSA method. This result is consistent with its theoretical significance. The MSA method performs computations simultaneously across multiple attention heads, giving the model a greater possibility to

discover meaningful relationships between features. The SSA method is limited by the single attention head because the single learning mode leads to a decrease in accuracy.

Consequently, in the MSA method, the setting of NumHeads is most critical. A too large NumHeads value will lead to attention distraction, and a too small NumHeads value will result in insufficient computational efficiency. In our model, we found that the value of NumHeads is more appropriate when the ratio of Channel to NumHeads is about 8.

4.3.2. Group 2: The Input Data Format of the SASW Module

In the PointSwin model, the input of the SASW module is the mixture of features and points of different dimensions to perform the computation of the MSA module. In this group of comparison experiments, we set up three additional groups of control experiments, which are the method of obtaining the correlation score by only interacting between points or features and the method to fuse correlation scores after interacting between points and features, respectively. The experimental results are shown in Table 4.

Table 4. The OA results of group 2.

Method	OA
Attention among features	90.03
Attention among points	89.7
Mixed after separate attention	90.19
Mixed before attention	90.88

There are two main types of input data for the Self-Attention mechanism module currently common in academia: one is Point-wise and the other is Channel-wise [10]. We believe that in point cloud processing tasks, considering the free spatial characteristics of point clouds, restricting the computational dimension to a specific dimension may result in point clouds not being able to exert their inherent advantages of spatial characteristics. After analyzing the success of the PointSwin model and the above experiments, we believe that feeding the Self-Attention mechanism with mixed-dimensional data is feasible and worthy of further study [10,43].

Through the analysis of comparative experimental data, we believe that the SASW mechanism is potentially motivated by the need for interactions between different points in different feature dimensions. This behavior of interacting between multidimensional features at different points is in line with the spatial significance of finding neighborhoods in point cloud data.

In the PointSwin model, we do not preprocess the input point cloud data for neighborhood sorting. Therefore, the model spontaneously imitates the action of finding neighborhoods of physical significance through the linkage of the window mechanism and the Cross-Window Attention method during training to supplement the demand for neighborhood sequences. This endogenous linkage imitating action ensures that the model can obtain better correlation feature representation.

The experimental results fully demonstrate the high demand for local neighborhoods of the SASW mechanism applied to point cloud data and also fully demonstrate the strong effectiveness of using features and points of different dimensions to be mixed and then input into the SASW module.

4.3.3. Group 3: The Hierachical Architecture

In PointSwin, the standard model adopts a combination of hierarchical structure and downsampling, while the efficient model abandons this setting and opts for paralleling two SASW modules to influence the model's decision. Here, we observe the effectiveness of hierarchical architecture by reducing the number of stacked layers or abandoning the hierarchies for the standard model and using multiple layers of stacking for an efficient model. The experimental results are shown in Table 5.

Table 5. The OA and flops results of group 3.

Method	OA	SASW Module Flops *	Time-Consuming **
PointSwin (reduce layers)	89.7	-	-
PointSwin (W/O hierarchies)	90.43	-	-
PointSwin	90.88	26.6 M	120 s
PointSwin-E (W/ hierarchies)	90.47	-	-
PointSwin-E	90.8	1.8 M	24 s

* The data are the core calculation amount of the SASW module in each batch when the BatchSize is set to 64. ** The data are the average time spent training each epoch under the NVIDIA TITAN RTX graphics card environment.

In PointSwin, the hierarchy represents stacking multiple SASW modules while also combining SASW modules with downsampling actions. Therefore, any feature will interact with other features in 2i different windows during the window processing of PointSwin with I being the number of SASW modules, and at the same time, it will be transformed into a coarse-grained feature with multiple downsampling operations. Such multiple interactions ensure sufficient interaction between features in local neighborhoods and also ensure the depth of participation of local features in building global features.

For PointSwin, the reduction in the number of layers in the hierarchical architecture will have the consequence of incomplete correlation learning between features. For the practice of directly canceling the hierarchy in the standard model, this behavior will also affect the downsampling method; that is, it will cause the downsampling method to fail completely. Therefore, this will not only affect the adequacy of the learning of associations between features but may also affect the inter-structural transfer of relevance scores and lead to the ineffective integration of associations between features.

The experimental results also show that the advantages of PointSwin-E lie in its simpler network and less computational overhead. Adding a hierarchical structure will have the opposite effect, causing not only increased computational overhead and thus efficiency but also a decrease in the accuracy of model decision making.

4.3.4. Group 4: The FFN Network

The Feed-Forward Network [9] (FFN) is located at the end of the SASW module. The model will perform a selective dropout operation on the final output feature map and add it to the input. FFN can be expressed as:

$$y = x + Drop(MSA(x)) \tag{10}$$

We observe the existence of this module by disabling and enabling FFN in comparative experiments. The results are shown in Table 6.

Table 6. The OA results of group 4.

Method	OA
PointSwin (W/O FFN)	89.26
PointSwin (W/ FFN)	90.88

The experimental results show that the FFN network plays the role of the gate effect in the SASW module. By adopting a method similar to residual connection to selectively correct the correlation of feature maps, the SASW mechanism not only ensures the sufficient learning of correlation scores between features but also limits the occurrence of meaningless correction behaviors that may lead to overfitting.

4.4. Ablation Study

In this section, we conduct ablation experiments for the three parts separately to observe their supportive or disturbing effects on the Self-Attention with Shifted-Window part.

4.4.1. Joint Align Network

Many advanced methods proposed by PointNet cancel the operation of correcting the input data. Therefore, we conduct ablation experiments on the Joint Align Network module to observe the effect of this module on PointSwin. The results are shown in Table 7.

Table 7. Results of ablation experiment about Joint Align Network.

Method	OA
PointSwin (W/O align)	90.55
PointSwin (alter align)	90.31
PointSwin	90.88

The work to align the features is necessary to ensure that PointSwin's use of mixed-dimensional features as input to the SASW module does not produce severe mismatch errors.

In our opinion, disabling the feature aligner does not give PointSwin a good effect. The model will still need to align the initial input through compensatory operations to ensure that the data remains robust to spatial transformations. So, when the feature alignment module is disabled, we believe that PointSwin will limit the interaction between features to avoid the overfitting tendency of the network.

4.4.2. MaxPooling Function

We try to extract global features by using an adaptive average Pooling function. The experimental comparison results are shown in Table 8.

Table 8. Results of ablation experiment about MaxPooling function.

Method	OA
PointSwin (AdaptivePooling)	87.68
PointSwin (MaxPooling)	90.88

One of the big advantages of MaxPooling is symmetry, which happens to be very suitable for processing point cloud data. We try to observe whether other pooling operators can work together with the SASW module to aggregate the global features of the point cloud. However, it can be seen from the experimental results that MaxPooling is still the best global feature extraction operator even for models with SASW modules.

4.4.3. The Combination Way of the Modules

How to combine the backbone network and the SASW module is an issue worth paying attention to. We conduct three experiments separately: insert the result of the SASW module into the backbone network as input (Insert), add the result of the SASW module to the output of the backbone network (Add), and concatenate the result of the SASW module with the output of the backbone network (Concat). The results are shown in Table 9.

Table 9. Results of ablation experiments about the modules combination way.

Method	OA
PointSwin (Insert)	89.7
PointSwin (Add)	87.8
PointSwin (Concat)	90.88

According to the experimental results, the Concat method is more effective. We believe that under the influence of the non-hierarchical feature extraction structure in the backbone network and the FFN in the SASW module, the concatenation approach is more suitable for our model.

Whether in the PointSwin or the PointSwin-E model, our purpose is to enhance the interaction of local features with each other to form an influence on the global feature decision so as to achieve the effect of fine-tuning the global feature. We hope to improve the accuracy of the model's target perception based on the excellent global features obtained by the backbone network and the emphasis on local features by the SASW module. However, neither the Insert mode nor the Add mode has performed both. Therefore, the concat mode is the most suitable for PointSwin and PointSwin-E.

5. Discussion and Future Work

The highlight of PointSwin is to enhance the local feature representation extracted by MLP through the Self-Attention with Shifted-Window mechanism. The focus of our work is to explore the feasibility of the Self-Attention with Shifted-Window mechanism in point cloud data processing and the degree of adaptation to the classic PointNet model. The experiments described above fully demonstrate that this mechanism is feasible. However, limited by the simple characteristics of the PointNet model as a baseline model, PointSwin has limited feature representation and has not achieved higher accuracy results. Because some restrictions exist in the point cloud data preprocessing steps, PointSwin still has several limitations in combating the problem of point cloud non-uniform sampling. These questions point out the direction for our future work.

Our goal is to better utilize the natural spatial location information of raw point cloud data, so in future research, we tend to optimize the model from two different aspects.

The first aspect is to search for generalized local neighborhoods adaptively without traversing the globe. Considering the pursuit of computational efficiency in the original intention of our model, we will continue to study generalized local neighborhood finding methods with less computational overhead.

The second aspect is to pay more attention to the adaptation of the model to the sampling of the input point cloud. Considering that the core mechanism of our model is to let the features shake hands with each other to determine the importance of association, we hope that the input point cloud can obtain more representative or random sampling points so as to achieve the interactive meaning of "from unfamiliar to familiar". In the future, we will try to promote the Self-Attention with Shifted-Window mechanism in larger-scale practical application scenarios. Therefore, from the perspective of improving the perception ability and computational efficiency of the model, we will continue to carry out research on the original point cloud pre-sampling algorithm to adapt to the model's higher requirements for the number of points and sampling models. Considering the time and labor cost of sampling and labeling, we hope that some object detection tasks can be accomplished during the learning process of weakly labeled data. For example, Ref. [60] uses a weakly supervised framework to learn 3D detection from weakly labeled data, and we can even use the model as a data annotation tool. This will be a very promising research direction, and we can apply this approach to projects.

In the future, we will continue to promote our model to other upstream tasks such as part segmentation, and we will continue to pay attention to the effective role that the Self-Attention with Shifted-Window mechanism can play in local feature representation. We believe that the above future works are all meaningful for research.

6. Conclusions

This paper mainly shows the PointSwin model and explores the feasibility of the Self-Attention with Shifted-Window mechanism on point cloud processing tasks. In the PointSwin model, the Self-Attention with Shifted-Window mechanism achieves the purpose of allowing arbitrary features to flow in different windows to participate in the correlation

calculation between features by stacking multiple SASW modules. Through transitive correlation computation, the model synchronously accomplishes the goal of enhancing local feature learning and assisting in building global features. Finally, the goal of improving the accuracy of the model's target perception is achieved. The good performance of PointSwin over PointNet and PointNet++ fully proves that the Self-Attention with Shifted-Window mechanism has a positive influence on promoting feature interaction.

In addition, this paper presents an efficient version of the PointSwin model, PointSwin-E. It achieves efficient and excellent performance by guiding part of the features to perform local feature-to-feature interactions. This fully demonstrates the effectiveness of the Self-Attention with Shifted-Window mechanism in the field of local feature learning of point clouds, and it lays the foundation for our follow-up research.

Author Contributions: C.J. and T.L. designed the model and implemented; C.J. and C.L. completed writing; C.J. and X.T. performed the experiments; T.L. and Y.P. guided the research. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the National Key R&D Program of China (grant number 2021ZD0140301), the National Natural Science Foundation of China (grant numbers 91948303-1, 611803375, 12002380, 62106278, 62101575, and 61906210), and the Postgraduate Scientific Research Innovation Project of Hunan Province (grant number QL20210018).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All datasets are publicly available.

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

References

1. He, P.; Ma, Z.; Fei, M.; Liu, W.; Guo, G.; Wang, M. A Multiscale Multi-Feature Deep Learning Model for Airborne Point-Cloud Semantic Segmentation. *Appl. Sci.* **2022**, *12*, 11801. [CrossRef]
2. Maturana, D.; Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 922–928.
3. Riegler, G.; Osman Ulusoy, A.; Geiger, A. Octnet: Learning deep 3d representations at high resolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3577–3586.
4. 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.
5. Wang, P.S.; Liu, Y.; Guo, Y.X.; Sun, C.Y.; Tong, X. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Trans. Graph. (TOG)* **2017**, *36*, 1–11. [CrossRef]
6. 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.
7. Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M. Deep learning for 3d point clouds: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 4338–4364. [CrossRef] [PubMed]
8. Liu, W.; Sun, J.; Li, W.; Hu, T.; Wang, P. Deep learning on point clouds and its application: A survey. *Sensors* **2019**, *19*, 4188. [CrossRef] [PubMed]
9. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
10. Lu, D.; Xie, Q.; Wei, M.; Xu, L.; Li, J. Transformers in 3D Point Clouds: A Survey. *arXiv* **2022**, arXiv:2205.07417.
11. Zhao, H.; Jia, J.; Koltun, V. Exploring self-attention for image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 10076–10085.
12. 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.
13. Hu, H.; Zhang, Z.; Xie, Z.; Lin, S. Local relation networks for image recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3464–3473.

14. Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; Shlens, J. Stand-alone self-attention in vision models. In Proceedings of the Advances in Neural Information Processing Systems 32, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates, Inc.: Red Hook, NY, USA, 2019.
15. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
16. Eldar, Y.; Lindenbaum, M.; Porat, M.; Zeevi, Y.Y. The farthest point strategy for progressive image sampling. *IEEE Trans. Image Process.* **1997**, *6*, 1305–1315. [CrossRef] [PubMed]
17. Bello, S.A.; Yu, S.; Wang, C.; Adam, J.M.; Li, J. Deep learning on 3D point clouds. *Remote Sens.* **2020**, *12*, 1729. [CrossRef]
18. Yu, X.; Tang, L.; Rao, Y.; Huang, T.; Zhou, J.; Lu, J. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 19313–19322.
19. 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.
20. Qian, G.; Li, Y.; Peng, H.; Mai, J.; Hammoud, H.A.A.K.; Elhoseiny, M.; Ghanem, B. PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies. *arXiv* **2022**, arXiv:2206.04670.
21. Mendes, N. Surface Electromyography Signal Recognition Based on Deep Learning for Human-Robot Interaction and Collaboration. *J. Intell. Robot. Syst.* **2022**, *105*, 1–21. [CrossRef]
22. Mendes, N.; Simão, M.; Neto, P. Segmentation of electromyography signals for pattern recognition. In Proceedings of the IECON 2019—45th Annual Conference of the IEEE Industrial Electronics Society, Lisbon, Portugal, 14–17 October 2019; IEEE: Piscataway, NJ, USA, 2019; Volume 1, pp. 732–737.
23. Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; Chen, B. Pointcnn: Convolution on x-transformed points. In Proceedings of the Advances in Neural Information Processing Systems 31, Montreal, QC, Canada, 3–8 December 2018; Curran Associates, Inc.: Red Hook, NY, USA, 2018.
24. 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.
25. 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, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6411–6420.
26. 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]
27. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
28. Mehta, S.; Rastegari, M. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv* **2021**, arXiv:2110.02178.
29. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 213–229.
30. Arnab, A.; Dehghani, M.; Heigold, G.; Sun, C.; Lučić, M.; Schmid, C. Vivit: A video vision transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 10–17 October 2021; pp. 6836–6846.
31. Liu, Z.; Hu, H.; Lin, Y.; Yao, Z.; Xie, Z.; Wei, Y.; Ning, J.; Cao, Y.; Zhang, Z.; Dong, L.; et al. Swin transformer v2: Scaling up capacity and resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 12009–12019.
32. Ren, P.; Li, C.; Wang, G.; Xiao, Y.; Du, Q.; Liang, X.; Chang, X. Beyond Fixation: Dynamic Window Visual Transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 11987–11997.
33. Guo, M.H.; Cai, J.X.; Liu, Z.N.; Mu, T.J.; Martin, R.R.; Hu, S.M. Pct: Point cloud transformer. *Comput. Vis. Media* **2021**, *7*, 187–199. [CrossRef]
34. Yang, J.; Zhang, Q.; Ni, B.; Li, L.; Liu, J.; Zhou, M.; Tian, Q. Modeling point clouds with self-attention and gumbel subset sampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3323–3332.
35. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 16259–16268.
36. Gao, Y.; Liu, X.; Li, J.; Fang, Z.; Jiang, X.; Huq, K.M.S. LFT-Net: Local Feature Transformer Network for Point Clouds Analysis. *IEEE Trans. Intell. Transp. Syst.* **2022**, 1–11. [CrossRef]
37. Pan, X.; Xia, Z.; Song, S.; Li, L.E.; Huang, G. 3d object detection with pointformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Montreal, QC, Canada, 11–17 October 2021; pp. 7463–7472.
38. Wu, L.; Liu, X.; Liu, Q. Centroid transformers: Learning to abstract with attention. *arXiv* **2021**, arXiv:2102.08606.

39. Lai, X.; Liu, J.; Jiang, L.; Wang, L.; Zhao, H.; Liu, S.; Qi, X.; Jia, J. Stratified Transformer for 3D Point Cloud Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 8500–8509.
40. Feng, M.; Zhang, L.; Lin, X.; Gilani, S.Z.; Mian, A. Point attention network for semantic segmentation of 3D point clouds. *Pattern Recognit.* **2020**, *107*, 107446. [CrossRef]
41. Yu, J.; Zhang, C.; Wang, H.; Zhang, D.; Song, Y.; Xiang, T.; Liu, D.; Cai, W. 3d medical point transformer: Introducing convolution to attention networks for medical point cloud analysis. *arXiv* **2021**, arXiv:2112.04863.
42. Qiu, S.; Anwar, S.; Barnes, N. Pu-transformer: Point cloud upsampling transformer. *arXiv* **2021**, arXiv:2111.12242.
43. Han, X.F.; Jin, Y.F.; Cheng, H.X.; Xiao, G.Q. Dual transformer for point cloud analysis. *IEEE Trans. Multimed.* **2022**, 1–10. [CrossRef]
44. Xu, G.; Cao, H.; Zhang, Y.; Ma, Y.; Wan, J.; Xu, K. Adaptive channel encoding transformer for point cloud analysis. In Proceedings of the International Conference on Artificial Neural Networks, Bristol, UK, 6–9 September 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 1–13.
45. Lin, L.; Huang, P.; Fu, C.W.; Xu, K.; Zhang, H.; Huang, H. One Point is All You Need: Directional Attention Point for Feature Learning. *arXiv* **2020**, arXiv:2012.06257.
46. Lin, L.; Huang, P.; Fu, C.W.; Xu, K.; Zhang, H.; Huang, H. On Learning the Right Attention Point for Feature Enhancement. *Sci. China Inf. Sci.* **2022**, 1–13.
47. Myronenko, A.; Song, X. Point set registration: Coherent point drift. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 2262–2275. [CrossRef]
48. Jaderberg, M.; Simonyan, K.; Zisserman, A. Spatial transformer networks. In Proceedings of the Advances in Neural Information Processing Systems 28, Montreal, QC, Canada, 7–12 December 2015; Curran Associates, Inc.: Red Hook, NY, USA, 2015.
49. Tolstikhin, I.O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; et al. Mlp-mixer: An all-mlp architecture for vision. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 24261–24272.
50. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.
51. Bao, H.; Dong, L.; Wei, F.; Wang, W.; Yang, N.; Liu, X.; Wang, Y.; Gao, J.; Piao, S.; Zhou, M.; et al. Unilmv2: Pseudo-masked language models for unified language model pre-training. In Proceedings of the International Conference on Machine Learning, Virtual Event, 13–18 July 2020; pp. 642–652.
52. Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3588–3597.
53. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-attention with relative position representations. *arXiv* **2018**, arXiv:1803.02155.
54. Yuan, J.; Liu, Z.; Wu, Y. Discriminative subvolume search for efficient action detection. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 2442–2449.
55. Xie, S.; Liu, S.; Chen, Z.; Tu, Z. Attentional shapecontextnet for point cloud recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4606–4615.
56. Su, H.; Maji, S.; Kalogerakis, E.; Learned-Miller, E. Multi-view convolutional neural networks for 3d shape recognition. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 945–953.
57. Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.R.; Smola, A.J. Deep sets. In Proceedings of the Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: Red Hook, NY, USA, 2017.
58. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. Shapenet: An information-rich 3d model repository. *arXiv* **2015**, arXiv:1512.03012.
59. 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]
60. Meng, Q.; Wang, W.; Zhou, T.; Shen, J.; Jia, Y.; Van Gool, L. Towards a weakly supervised framework for 3d point cloud object detection and annotation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 4454–4468. [CrossRef]