Transformers in 3D Point Clouds: A Survey

Dening Lu, Qian Xie, Mingqiang Wei, *Senior Member, IEEE*
Linlin Xu, *Member, IEEE,* Jonathan Li, *Senior Member, IEEE*

arXiv:2205.07417v1 [cs.CV] 16 May 2022

***Abstract*—In recent years, Transformer models have been proven to have the remarkable ability of long-range dependencies modeling. They have achieved satisfactory results both in Natural Language Processing (NLP) and image processing. This significant achievement sparks great interest among researchers in 3D point cloud processing to apply them to various 3D tasks. Due to the inherent permutation invariance and strong global feature learning ability, 3D Transformers are well suited for point cloud processing and analysis. They have achieved competitive or even better performance compared to the state-of-the-art non-Transformer algorithms. This survey aims to provide a comprehensive overview of 3D Transformers designed for various tasks (e.g. point cloud classification, segmentation, object detection, and so on). We start by introducing the fundamental components of the general Transformer and providing a brief description of its application in 2D and 3D fields. Then, we present three different taxonomies (i.e., Transformer implementation-based taxonomy, data representation-based taxonomy, and task-based taxonomy) for method classification, which allows us to analyze involved methods from multiple perspectives. Furthermore, we also conduct an investigation of 3D self-attention mechanism variants designed for performance improvement. To demonstrate the superiority of 3D Transformers, we compare the performance of Transformer-based algorithms in terms of point cloud classification, segmentation, and object detection. Finally, we point out three potential future research directions, expecting to provide some benefit references for the development of 3D Transformers.**

***Index Terms*—3D Transformers, Point Cloud Processing, Self-attention Mechanism, Deep Neural Networks, Survey.**

I. Introduction

Transformer models, as encoder-decoder architectures, have become the dominant algorithms in Natural Language Processing (NLP). Due to the impressive ability of long-range dependencies modeling, they have been expanded to the field of computer vision. As shown in Fig. 1, a standard Transformer encoder generally consists of six main components: 1) input (word) embedding; 2) positional encoding; 3) self-attention mechanism; 4) normalization; 5) feed-forward operation; and 6) skip connection. For the 3D point cloud processing, the Transformer decoder is designed for the dense prediction tasks such as part segmentation and semantic segmentation. And it usually adopts the PointNet++ [1] or U-Net design where Transformer blocks are also incorporated. Here, we take an input point cloud $P = \{p_1, p_2, p_3, ..., p_N\} \in R^{N \times D}$ as an example to describe each component in the Transformer encoder. D is the feature dimension of the input point. D equals three means only 3D coordinates of the point cloud are taken as the input, while D equals six means 3D coordinates and normal vectors are both taken as the input. The details of the components mentioned above are as follows:

- Firstly, for the input embedding, P is projected to a high-dimension feature space which can facilitate subsequent learning. It can be achieved by a Multi-Layer Perception (MLP) or other feature extraction backbone networks like PointNet [2]. We denote the embedded feature map as $X \in R^{N \times C}$.
- Secondly, the positional encoding is used to capture the geometrical information of the input data. Since the 3D coordinates can be taken as natural position information, the position encoding can be crafted manually by sine and cosine functions or some normalization operations [3]. Moreover, there also exist some position encoding schemes with a trainable parameter matrix B [4], [5], which is more adaptive to different input data.
- Thirdly, the core component of the Transformer encoder is the self-attention mechanism. The sum of the embedded feature map X and positional encoding result is taken as the input. And then it is projected to three different feature spaces by three learnable weight matrices $W_Q \in R^{C \times C_Q}, W_k \in R^{C \times C_K}, W_V \in R^{C \times C}$, where C_K equals C_Q. In this way, $Query$, Key, and $Value$ matrices can be expressed as:

$$\begin{cases} Query & = X \times W_Q, \\ Key & = X \times W_K, \\ Value & = X \times W_V. \end{cases} \tag{1}$$

  This operation is able to improve the expression ability of the self-attention mechanism. Given the $Query$, Key, and $Value$ matrices, an attention map can be expressed as follows:

$$Attentionmap = Softmax(\frac{Q \times K^T}{\sqrt{C_K}}), \tag{2}$$

  where Q, K, V mean the $Query$, Key, and $Value$ matrices respectively, "$\times$" means the matrix multiplication. As we can see, the attention map with the size of $N \times N$ measures the similarity of any two input points, so it is also called the similarity matrix. Then the attention map and $Value$ matrix are multiplied to generate the new feature map F, with the same size as X. Each feature vector in F is obtained by computing a weighted sum of all input features, so it is able to establish connections

Dening Lu and Qian Xie contribute equally to this work and should be considered co-first authors.

Dening Lu, Linlin Xu are with the Department of Systems Design Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: d62lu@uwaterloo.ca; linlinxu618@gmail.com).

Mingqiang Wei is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China (e-mail: mingqiang.wei@gmail.com).

Jonathan Li is with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: junli@uwaterloo.ca).

Qian Xie is with the the Department of Computer Science, University of Oxford, Oxford OX1 3QD, U.K. (e-mail: qian.xie@cs.ox.ac.uk).

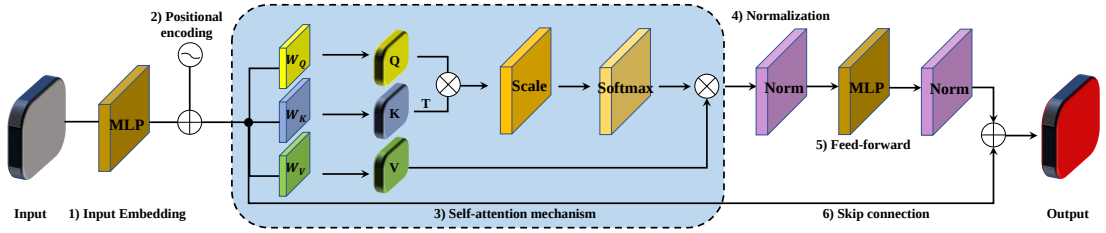

Fig. 1: Illustration of the Transformer encoder architecture.

with all input features. And this is the reason why Transformers are good at global feature learning.

- Fourthly, a normalization layer is placed before and after the feed-forward layer, performing the standardizing and normalizing operations to feature maps. There are two kinds of normalization methods used in this layer: LayerNormalization or BatchNormalization. The former is commonly used in Natural Language Processing (NLP), while the latter is commonly used in computer vision like 2D or 3D data processing.
- Fifthly, a feed-forward layer is added to enhance the representation of the attention features. Generally, it consists of two fully-connection layers with a RELU function.
- Finally, a skip connection is used to build the connection between the input and output of the self-attention module. There have been many self-attention variants using various skip connection forms [6]–[8], which we will give more details in Sec. V.

Note that there also are some 3D Transformers that do not include all these six components completely. For example, early 3D Transformer networks like Point Attention (P-A) network [6] and Attentional ShapeContextNet [7] do not have the positional encoding module. They focus on applying the self-attention mechanism to 3D point cloud processing. Point Cloud Transformer (PCT) [8] proposes a neighbor embedding mechanism achieved by EdgeConv [9]. This mechanism incorporates the positional encoding into the input embedding module. Since the self-attention mechanism is the core component of Transformers, we also incorporate the methods mainly utilizing the self-attention mechanism for point cloud processing into the 3D Transformer family.

Transformer models have been introduced to image processing widely, and achieved satisfying results for various tasks, such as image segmentation [10], object detection [11] and tracking [12]. Vision Transformer (ViT) [13] first proposes a pure Transformer network for image classification. It achieves excellent performance compared with the state-of-the-art convolutional networks. Based on ViT, there are massive Transformer variants proposed for image classification [14]–[17], segmentation [18]–[20], object detection [11], [15], [21], [22], and other vision tasks. Moreover, various Transformer architectures are proposed for performance improvement, such as convolutions+Transformers [16], multi-scale Transformers [17], and self-supervised Transformers [23]. There also have been several surveys [24]–[26] proposed to categorize all involved 2D Transformers into multiple groups. The taxonomies they use are usually algorithm architecture-based taxonomy and task-based taxonomy.

Due to the remarkable global feature learning ability and order-independent operations, Transformer architectures have also been applied to 3D point cloud processing and analysis. As shown in Fig. 2, a number of 3D Transformer backbones are proposed for point cloud classification & segmentation [4], [8], [30], [33], [66], [67], detection [31], [50], tracking [52]–[54], registration [55]–[59], [68], [69], completion [46], [62]–[65], [70] an so on. Moreover, 3D Transformer networks have also been applied to various practical application fields, such as medical data analysis [33] and autonomous driving [71], [72]. Therefore, it is necessary to conduct a systematic survey for 3D Transformer works. Recently, several 3D Transformer/Attention related reviews have been published. For instance, Khan et al. [25] review the vision Transformers according to the architecture- and task-based taxonomies. However, it mainly focuses on Transformers on 2D image analysis, and only provides a brief introduction of 3D Transformer networks. Qiu et al. [73] introduces several 3D self-attention mechanism variants, and conducts a detailed comparison and analysis for them on SUN RGBD [74] and ScanNetV2 datasets [75]. However, a comprehensive survey of Transformer models in 3D point clouds has not been conducted so far. This survey aims to provide a comprehensive investigation of 3D Transformers, based on the existing review works above. As shown in Fig. 3, we design three different taxonomies: 1) Transformer implementation-based taxonomy; 2) data representation-based taxonomy; and 3) task-based taxonomy. In this way, we are able to analyze Transformer networks from multiple perspectives. There could be an intersection between different categories. Taking Point Transformer (PT) [4] as an example: 1) in terms of the Transformer implementation, it belongs to the local Transformer category, operated in the local neighborhood of the target point cloud; 2) in terms of the data representation, it belongs to the multi-scale point-based Transformer category, extracting the geometrical and semantic features hierarchically; 3) in terms of the 3D task, it is designed for point cloud classification and segmentation. Additionally, we also conduct an investigation of different self-attention variants in 3D point cloud processing. We expect to provide some benefit references for the development of Transformer-based networks.

The major contributions of this survey can be summarised as follows:

- This is the first survey paper, to the best of our knowledge, that focuses on comprehensively covering Transformers in 3D point cloud processing and analysis.

- This paper investigates a series of self-attention variants in 3D point cloud processing. It introduces novel self-attention mechanisms that aim at improving the performance and efficiency of 3D Transformers.
- This paper conducts brief comparisons and analysis of Transformer-based methods on several 3D tasks, including 3D shape classification and 3D shape/semantic segmentation, and 3D object detection on several public benchmarks.
- This paper can provide the readers with the SOTA methods, since the most recent and advanced progress of Transformers on 3D point clouds is provided.

The paper is organized into six sections after the Introduction. Sec. II, III, and IV design three different taxonomies for 3D Transformer classification. Sec. V reviews different self-attention variants which are proposed to improve the performance of Transformers. Sec. VI makes a brief comparison and analysis of the involved 3D Transformer networks. Lastly, Sec. VII summarizes our survey work, and points out three potential future directions for 3D Transformers.

II. Transformer Implementation

In this section, we broadly categorize 3D point cloud Transformers from multiple perspectives. Firstly, in terms of the operating scale, 3D Transformers can be divided into two parts: Global Transformers and Local Transformers (Sec. II-A). The operating scale represents the scope of the algorithm in the point cloud, such as the global domain or the local domain. Secondly, in terms of the operating space, 3D Transformers can be divided into Point-wise Transformers and Channel-wise Transformers (Sec. II-B). The operating scale represents the dimension in which the algorithm is operated, such as the spatial dimension or the channel dimension. Lastly, we review efficient Transformer networks which are designed to reduce the computational cost (Sec.II-C).

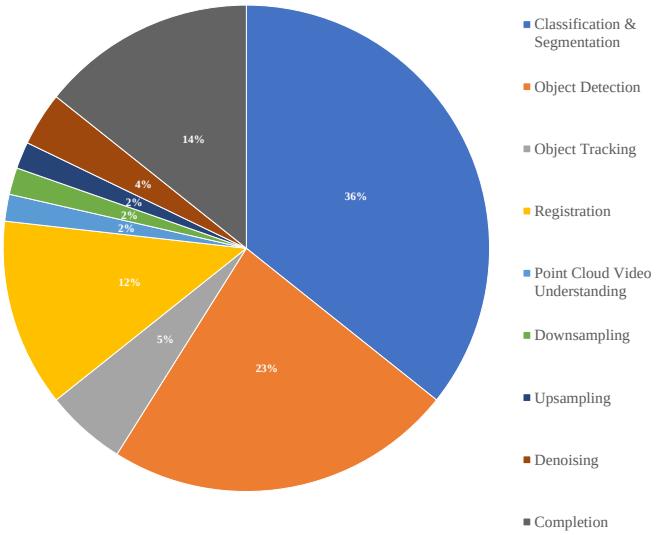

Fig. 2: Applications of Transformers in 3D point cloud processing.

A. Operating Scale

According to the operating scale, 3D Transformers can be divided into two parts: Global Transformers and Local Transformers. The former means that Transformer blocks are applied to all the input points for global feature extraction, while the latter means that Transformer blocks are applied in a local patch for local feature extraction.

1) Global Transformers: There are many existing works [5]–[8], [27], [29], [33], [34], [48], [76] focusing on global Transformer studying. For a global Transformer block, each new output feature in F can establish connections with all input features X, making it permutation-invariant and capable of learning the global context features [8].

Similar to the PointNet [2] structure, Guo et al. [8] propose Point Cloud Transformer (PCT), a pure global Transformer network. Taking the 3D coordinates as the input P, PCT first proposes a neighbor-embedding architecture to map the point cloud into a high dimensional feature space. This operation can also incorporate local information into the embedded features. And then these features are fed into four stacked global Transformer blocks to learn semantic information. The global features are finally extracted by a global Max and Average (MA) pooling for classification and segmentation. Moreover, PCT designs an improved self-attention module, named Offset-Attention (OA), inspired by the Laplacian matrix in Graph convolution networks [77]. We detail the structure of the OA module in Sec. V-A. It is able to sharpen the attention weights and reduce the influence of noise. The state-of-the-art performance of PCT on various tasks proves that Transformers are suitable for 3D point cloud processing.

Unlike the single scale of PCT, Han et al. [27] present a Cross-Level Cross-Scale Cross-Attention Transformer network, named 3CROSSNet. Firstly, it performs Farthest Point Sampling (FPS) algorithm [1] on the raw input point cloud to obtain three point subsets with different resolutions. Secondly, it utilizes stacked multiple shared Multi-Layer Perception (MLP) modules to extract local features for each sampling point. Thirdly, it applies Transformer blocks to each point subset for global feature extraction. Finally, the Cross-Level Cross-Attention (CLCA) module and Cross-Scale Cross-Attention (CSCA) module are proposed to build connections between different-resolution point subsets and different-level features for long-range inter- and intra-level dependencies modeling.

Yu et al. [29] propose a BERT-style pre-training strategy for 3D global Transformers, which generalizes the concept of BERT [78] to 3D point cloud processing. Taking local patches as the input, they first utilize the mini-PointNet [2] for the input embedding, following ViT [13]. And then they design a point cloud Tokenizer with a discrete Variational AutoEncoder (dVAE) [79] to convert the embedded points into discrete point tokens for pre-training. The Tokenizer network is achieved by DGCNN [9] for meaningful local information aggregating, and is learned through dVAE-based point cloud reconstruction. During pre-training, the point embeddings with some masked tokens are fed into the Transformer encoder. Supervised by the point tokens generated by the Tokenizer, the encoder

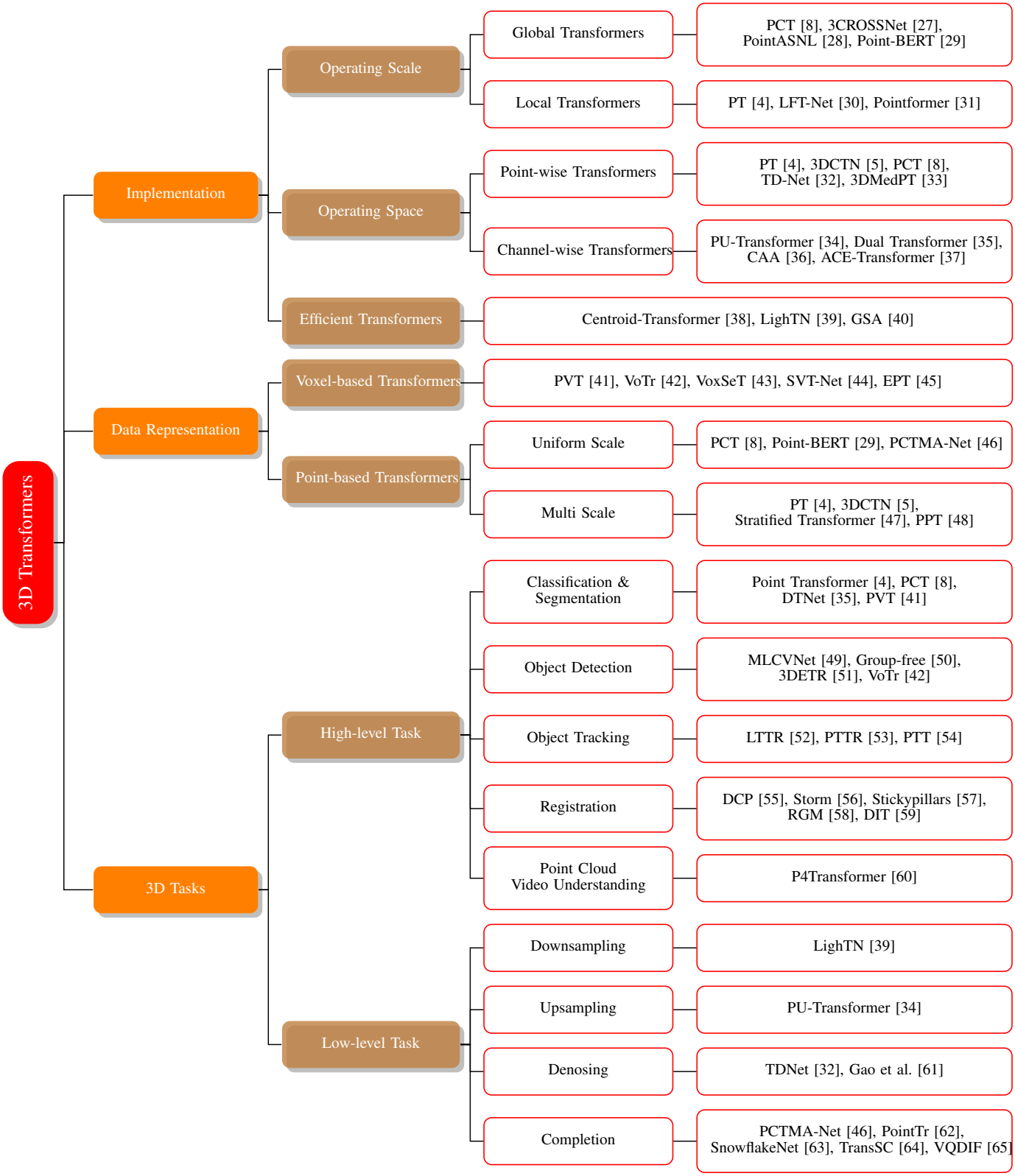

Fig. 3: Taxonomies of 3D Transformers.

can be trained to recover the corresponding tokens of the masked locations. The authors have conducted comprehensive experiments to show that the BERT-style pre-training strategy is able to improve the performance of the pure Transformer in point cloud classification and segmentation.

2) Local Transformers: Unlike global Transformers, local Transformers [4], [30], [31], [38] aim to achieve feature aggregation in the local patch instead of the entire point cloud.

Point Transformer (PT) [4] adapts the PointNet++ [1] hierarchical architecture for point cloud classification and segmentation. It focuses on local patch processing, and replaces the shared MLP modules in PointNet++ with local Transformer blocks. PT has five local Transformer blocks operated on progressively downsampled point sets. Each block is applied within KNN local neighborhoods around sampled points. Especially, the self-attention operator that PT uses is the vector attention [80] instead of the scalar attention. The former has been proven to be more effective for point cloud processing, since it supports channel-wise attention weight assignment, not just assigning one weight to a whole feature vector. Please refer to Sec. VII for the specific expression of the vector attention.

Similarly, Local Feature Transformer Network (LFT-Net) [30] also proposes to increase the expression ability of local fine-grained features. It consists of four stacked local Transformer and Trans-pool blocks, so the local features can be continuously aggregated into the global features. There are two main differences between LFT-Net and PT. One is that LFT-Net proposes a trans-pooling model, instead of commonly-used symmetry functions like max/mean/sum pooling. This model is able to alleviate the feature discarding. Another is that LFT-Net applies multi focal-loss instead of the standard cross-entropy loss function. Such loss can address the problems of class imbalance and weak learning ability for complex regions in the semantic segmentation task. It reshapes the standard cross-entropy loss by adding a class-based weight and a decay coefficient, which are able to balance the data distribution, and enhance the impact of the low-accuracy class on the loss.

Pan et al. [31] propose Pointformer to combine the local and global features both extracted by Transformer blocks for 3D object detection. It has three kinds of main blocks: a Local Transformer (LT) block, a Global Transformer (GT) block and a Local-Global Transformer (LGT) block. Firstly, the LT block applies the dense self-attention operation in the neighborhood of each centroid point generated by FPS [1]. Secondly, taking the whole point cloud as the input, the GT block aims to learn global context-aware features via the self-attention mechanism. Lastly, the LGT block adopts a multi-scale cross-attention module, to build connections between local features from the LT and global features from the GT. Specifically, the LGT block takes the output of the LT as $query$, and the output of the GT as key and $value$ to conduct the self-attention operation. In this way, all centroid points can be utilized to integrate global information, which leads to effective global feature learning.

Inspired by Swin Transformer [17], Lai et al. [47] propose Stratified Transformer for 3D point cloud segmentation. It splits the point cloud into a group of non-overlapping cubic windows via 3D voxelization and performing the local Transformer operation in each window. Stratified Transformer also follows the encoder-decoder architecture. The encoder is a hierarchical structure consisting of multiple stages, where each stage has two successive Transformer blocks. The former block utilizes a Stratified Self-Attention (SSA) to capture the long- and short-range dependencies. And the latter block utilizes a Shifted SSA to further strengthen the connections between different independent windows, following Swin Transformer [17]. Specifically, to solve the issue that the local Transformer is weak in capturing global information, SSA generates the dense local key points and sparse distant key points for each $query$ point. The former is generated in the window that the $query$ point belongs to, while the latter is generated in a larger window by downsampling the entire input point cloud. In this way, the receptive field of the $query$ point is not limited in the local window, allowing SSA to build the long-range contextual dependencies. Additionally, Stratified Transformer performs a KPConv [81] embedding in the first stage to extract the local geometric information of the input point cloud. This operation has been proven to be effective in their ablation experiments.

B. Operating Space

According to the operating space, 3D Transformers can be divided into two categories: Point-wise Transformers and Channel-wise Transformers. The former is to measure the similarity among input points, while the latter is to distribute attention weights along channels without being concerned by the unorderedness of point clouds [36]. Generally, according to Eq. 2, the attention maps of these two kinds of Transformers can be expressed as:

$$\begin{aligned} Point-wise\ Attn &= Softmax(\frac{Q \times K^T}{\sqrt{C_K}}), \\ Channel-wise\ Attn &= Softmax(\frac{Q^T \times K}{\sqrt{C_K}}), \end{aligned} \tag{3}$$

where the size of $Point-wise\ Attn$ is $N \times N$, while the size of $Channel-wise\ Attn$ is $C_K \times C_K$.

1) Point-wise Transformers: Point-wise Transformers aim to investigate the spatial correlation among points and learn the long-range context-dependent representation. The output feature map of point-wise Transformers can be formulated as a weighted sum of all input features. It spatially describes the long-range contextual dependencies [35]. Since global Transformers and local Transformers in Sec. II-A are distinguished by the spatial operating scale, i.e., the whole point cloud or a local patch, all involved methods [4]–[8], [27], [29]–[31], [33], [34], [38], [47], [48], [76] in Sec. II-A can be considered as point-wise Transformers. Apart from these methods, point-wise Transformers are also widely applied to other tasks.

Xu et al. [32] propose an encoder-decoder Transformer network (TD-Net) for point cloud denoising. The encoder consists of a coordinate-based input embedding module, an adaptive sampling module and four stacked point-wise self-attention modules. The outputs of the four self-attention modules are concatenated together as the input of the decoder. Additionally, the adaptive sampling approach can automatically learn the

offset of each sampling point generated by FPS [1]. This operation allows the sampling points closer to the underlying surface. The decoder is applied to construct the underlying manifold according to the extracted high-level features. And finally, a clean point cloud can be reconstructed by the manifold sampling.

Yu et al. [33] propose 3D Medical Point Transformer (3DMedPT) for medical point cloud analysis. Specifically, it presents a hierarchical point-wise Transformer for classification and a uniform-scale point-wise Transformer for segmentation. Especially, 3DMedPT introduces the convolution operation to the point-wise Transformer block. It adds a local feature extraction module achieved by DGCNN [9] before each Transformer block. Point Attention Network (P-A) [6] and Pyramid Point Cloud Transformer (PPT) [48] also have similar structures. Considering insufficient training sample processing in the medical domain, 3DMedPT proposes a special module named Multi-Graph Reasoning (MGR), to enrich the feature representations.

2) Channel-wise Transformers: Different from point-wise Transformers, the channel-wise Transformers [34]–[37], [66] focus on measuring the similarity of different feature channels. They are able to improve the context information modeling by highlighting the role of interaction across different channels [35].

Han et al. [35] propose Dual Transformer Network (DT-Net) for point cloud analysis, which applies the point-wise Transformer and channel-wise Transformer operations simultaneously. After the point-wise and channel-wise features are extracted respectively, they are concatenated via element-wise sum operation to improve feature representation. Ablation studies in [35] demonstrate that DT-Net with both point-wise and channel-wise Transformer modules achieves the best results, compared to that with only point-wise or only channel-wise Transformers.

Qiu et al. [36] propose a back-projection CNN module for local feature capturing, leveraging an idea of error-correcting feedback structure. It designs a Channel-wise Affinity Attention (CAA) module for better feature representations Specifically, the CAA module consists of two blocks: a Compact Channel-wise Comparator (CCC) block and a Channel Affinity Estimator (CAE) block. The CCC block can generate the similarity matrix in the channel space, without being concerned by the unorderedness of point clouds. The CAE block further calculates an affinity matrix, in which an element with a higher attention value represents the lower similarity of the corresponding two channels. This operation can sharpen the attention weights and avoid aggregating similar/redundant information. In this way, each channel of the output feature has sufficient intersection with other distinct ones, which has been proven to be beneficial to the final results. We also detail the CAA structure in Sec. V.

Instead of only using the feature channels, Transformer-Conv proposed in [37] combines both coordinate and feature channels to design a novel channel-wise Transformer. Specifically, the $Query$ matrix is generated directly by the coordinate information without any linear transformation, while the Key matrix is generated by feature channels with an MLP. And then the attention matrix is calculated by element-wise multiplication rather than dot product. In this way, the attention matrix is able to represent the relationship between the coordinate channels and feature channels of each point. Since all features come from the coordinate space, an element with a higher value in the attention matrix tends to represent that the corresponding feature channel is more faithful to the coordinate space. After that, the $Value$ matrix is expanded from the Key matrix by MLP. And then the new feature map, called the response matrix, can be obtained by element-wise multiplication between the $Value$ matrix and attention matrix. The response matrix consists of the weighted feature channels of all input points. Finally, the output features are generated by applying a channel max-pooling operation to the response matrix. The max-pooling plays a screening role and it is able to select the most important channels, i.e., the channels which fit the coordinate space best. Such channels have been proven to be effective for point cloud analysis by the ablation experiments.

C. Efficient Transformers

Despite the great success in point cloud processing, standard Transformers tend to cause high computation costs and memory consumption because of massive linear operations. Given an input point cloud with N points, the computation and memory complexities of the standard self-attention module are quadratic on N, i.e., $O(N_2)$. This is the key drawback when applying Transformers on large-scale point cloud datasets.

Recently, there are several 3D Transformers researching on improving the self-attention module for higher computational efficiency. For instance, Centroid Transformer [38] takes N point features as the input while outputs a smaller number M of point features. In this way, the key information in the input point cloud can be summarized in a smaller number of the outputs (called centroids). Specifically, it first constructs M centroids from N input points, by optimizing a general "soft K-means" objective function. And then it uses the M centroids and N input points to generate the $Query$ and Key matrices respectively. The size of the attention map is reduced from $N \times N$ to $M \times N$, so the computational cost of the self-attention is reduced from $O(N_2)$ to $O(NM)$. To further save the computational cost, the authors apply a K-Nearest Neighbor (KNN) approximation. This operation essentially converts the global Transformer to the local Transformer. In this case, the similarity matrix is generated by measuring the relationships among each $query$ feature vector and its K neighbor key vectors, instead of N vectors. So the computational cost can be further reduced to $O(NK)$. Similarly, Cheng et al. [82] also propose PatchFormer to reduce the size of the attention map. It first splits the raw point cloud into M patches, and then aggregates the local feature in each patch. The significant difference is that PatchFormer uses M aggregated local features to generate Key matrix, while Centroid Transformer uses M centroids to generate $Query$ matrix. In this way, the computational cost of the self-attention in PatchFormer can also be reduced to $O(NM)$.

Wang et al. [39] propose Light-weight Transformer Network (LighTN) to reduce the computational cost in a different way.

LighTN aims to simplify the main components in the standard Transformer, maintaining the superior performance of Transformers while increasing the efficiency. Firstly, it removes the positional encoding block because the input 3D coordinates can be considered as a substitute for the positional encoding. This operation eliminates the overhead of positional encoding itself. Secondly, it utilizes a small-size shared linear layer as the input embedding layer. And the dimensions of embedded features are reduced by half compared to the computationally-saving neighbor embedding setting in [8]. In this way, the computational costs of the input embedding can be reduced. Thirdly, it presents a single head self-correlation layer as the self-attention module. The projection matrices of W_Q, W_K, and W_V are removed, to reduce learnable parameters for high efficiency. Since the attention map is generated only by the input self-correlation parameters, the self-attention module is also named the self-correlation module, which can be formulated as:

$$\begin{aligned} SA(X) &= FC_{out}(C(X)), \\ C(X) &= softmax(\frac{X \times X^T}{\sqrt{C}}) \times X, \end{aligned} \tag{4}$$

where $SA(*)$ represents the self-attention block, FC_{out} represents the linear transformation, $softmax(*)$ is the activation function, and C is the input feature dimension declared in Eq. 2. Lastly, the authors build three linear layers (a standard FFN block generally has two linear layers) in the Feed-Forward Network (FFN) and use the expand-reduce strategy [83] in the middle layer. So the negative impact caused by the decreasing learnable parameters in the self-correlation layer can be mitigated. By this means, LighTN is able to guarantee the performance and the FFN only causes a slight increase in the computational costs. Similarly, Group Shuffle Attention (GSA) proposed in [40] also simplifies the self-attention mechanism in its Transformer network. It integrates the shared projection weight matrix and non-linearity function into the self-attention mechanism (please see Sec. V-A for detailed description of GSA).

III. Data Representation

There are several forms of 3D data representation, such as points and voxels, both of which can be used as the input of 3D Transformers. Since points can be represented by voxels, several voxel-based approaches can also be performed on point clouds, so as to 3D Transformers. According to different input formats, we divide 3D Transformers into Voxel-based Transformers and Point-based Transformers.

A. Voxel-based Transformers

Unlike images, 3D point clouds are generally unstructured, and cannot be directly processed by traditional convolution operators. However, 3D point clouds can be easily converted into 3D voxels, which are structured like images. Thus, some Transformer-based works [41]–[43], [47] explore to transform 3D point clouds into voxel-based representation. The most general voxelization approach can be described as follows [84]: The bounding box of a point cloud is first regularly divided into 3D cuboids via rasterization. Voxels containing points are retained, generating the voxel representation of point clouds.

Inspired by the ability of sparse convolution to process voxel data fast [85], [86], Mao et al. [42] first propose Voxel Transformer (VoTr) backbone for 3D object detection. They present the submanifold voxel module and the sparse voxel module to extract features from non-empty and empty voxels respectively. In both two modules, the Local Attention and Dilated Attention operation are designed, on the basis of the Multi-head Self-Attention mechanism (MSA), to maintain low computational consumption for numerous voxels. As stated, the proposed VoTr can be integrated into most voxel-based 3D detectors. To tackle the computation issue of Transformers in voxel-based outdoor 3D detectors, Voxel Set Transformer (VoxSeT) is proposed in [43] to detect outdoor objects in a set-to-set fashion. Based on the low-rank characteristic of the self-attention matrix, a Voxel-based Self-Attention (VSA) module is designed by assigning a set of trainable "latent codes" to each voxel, which is inspired by the induced set attention blocks in Set Transformer [87].

Inspired by the effectiveness of voxel-based representation on large-scale point clouds, voxel-based Transformers can also be applied to large-scale point cloud processing. For instance, Fan et al. [44] present Super light-weight Sparse Voxel Transformer (SVT-Net) for large scale place recognition. They design an Atom-based Sparse Voxel Transformer (ASVT) and a Cluster-based Sparse Voxel Transformer (CSVT). The former is used to encode short-range local relations, while the latter is used to learn long-range contextual relations. Park et al. [45] propose Efficient Point Transformer (EPT) for large-scale 3D scene understanding from point clouds. To relieve the problem of geometric information loss during voxelization, they introduce the center-aware voxelization and devoxelization operations. On this basis, Efficient Self-Attention (ESA) layers are employed to extract voxel features. Their center-aware voxelization can preserve positional information of points in voxels.

B. Point-based Transformers

Although voxels are of regular format, the transformation to voxels would lead to geometric information loss to some extent and cause issues [1], [2]. The format of the point cloud is the original representation, which preserves complete geometric information of the input data. Thus, most Transformer-based point cloud processing frameworks fall into this category. And their algorithm architectures are usually designed in two main groups: uniform-scale architecture [8], [29], [46], [59], [76] and multi-scale architecture [4], [5], [33], [35], [47], [48].

1) Uniform Scale: Uniform-scale architectures usually keep the scale of the point features constant during data processing. The number of output features of each module is consistent with the number of input features. The most representative work is PCT [8], which has been discussed in Sec. II-A. After the input embedding stage, four global Transformer blocks of PCT are directly stacked together to refine point features. There is no hierarchical feature aggregation operation, which

facilitates the decoder designing for dense prediction tasks like point cloud segmentation. And feeding all input points into the Transformer block is beneficial to global feature learning. However, uniform-scale Transformers tend to be weak in extracting the local features due to the lack of the local neighborhoods. And processing the whole point cloud directly would lead to a high computation cost and memory consumption.

2) Multi Scale: Multi-scale Transformers refer to those with progressive point sampling strategies during feature extraction, also called hierarchical Transformers. Point Transformer (PT) [4] is the pioneering one that introduces the multi-scale structure to a pure Transformer network. Transformer layers in PT are employed on progressively sampling point sets. On one hand, sampling operations can accelerate the computation of the whole network by reducing the parameters of the Transformer. On the other hand, these hierarchical structures usually come with KNN-based local feature aggregation operations. This local feature aggregation is beneficial to the tasks that need fine semantic perception, such as segmentation and completion. And the highly aggregated local features at the last layer of the network can be taken as the global features, which can be used for point cloud classification. Additionally, there also exist many multi-scale Transformer networks [5], [33], [47], [48] that utilize EdgeConv [9] or KPconv [81] for local feature extraction and utilize Transformers for global feature extraction. In this way, they are able to combine the strong local modeling ability of convolutions and the remarkable global feature learning ability of Transformers for better semantic feature representation.

IV. 3D Tasks

Similar to image processing [25], 3D point cloud related-tasks can also be divided into two main groups: high-level and low-level tasks. High-level tasks involve semantic analysis, which focuses on translating 3D point clouds to information that people can understand. Low-level tasks, such as denoising and completion, focus on exploring fundamental geometric information. They are not directly related to human semantic understanding but can indirectly contribute to high-level tasks.

A. High-level Task.

In the field of 3D point cloud processing, high-level tasks usually include: classification & segmentation [4], [7], [8], [28]–[30], [33], [35], [36], [38], [40], [41], [47], [82], [88], [89], object detection [31], [42], [43], [49]–[51], [66], [90], [91] and tracking [52]–[54], registration [55]–[59], [68], [69] and so on. Here, we start by introducing classification & segmentation tasks, which are very common and fundamental research topics in the field of 3D computer vision.

1) Classification & Segmentation: Similar to image classification [92]–[95], 3D point cloud classification methods aim at classifying the given 3D shapes into specific categories, such as chair, bed and sofa for indoor scenes, and pedestrian, cyclist and car for outdoor scenes. In the field of 3D point cloud processing, since the encoders of segmentation networks are usually developed from classification networks, we integrate these two tasks as one for introduction here.

Xie et al [7], for the first time, propose to introduce the self-attention mechanism into the task of point cloud recognition. Inspired by the success of shape context [96] in shape matching and object recognition, the authors first transform the input point cloud into a form of shape context representation. It is comprised of a set of concentric shell bins. Based on the proposed novel representation, they then present the ShapeContextNet (SCN) to perform point feature extraction. To automatically capture the rich local and global information, a sot-product self-attention module is further adopted on the shape context representation, resulting in the Attentional ShapeContextNet (A-SCN).

Inspired by self-attention networks in image analysis [80], [97] and Natural Language Processing (NLP) [78], Zhao et al. [4] design a vector attention-based Point Transformer layer. A Point Transformer block is constructed on the basis of the Point Transformer layer in a residual fashion. The encoder of Point Transformer is constructed by only Point Transformer blocks, pointwise transformations and pooling operation for point cloud classification. Moreover, Point Transformer also uses a U-Net structure for point cloud segmentation, where the decoder is designed to be symmetrical with the encoder. It presents a Transition Up module to recover the original point cloud with semantic features from the downsampled point set. Such module consists of a linear layer, batch normalization, ReLU, and trilinear interpolation for feature mapping. Additionally, a skip connection between the encoder block and the corresponding decoder block is introduced to enhance the interpolated features. With these carefully designed modules, Point Transformer becomes the first model that reaches over 70% mIoU (70.4%) for semantic segmentation on Area 5 of S3DIS dataset [98]. As for the task of shape classification on ModelNet40 dataset, Point Transformer also achieves 93.7% overall accuracy.

Point Cloud Transformer (PCT) [8] proposes to capture long-range relationships among input points (i.e., global context) for point cloud classification. Since PCT applies the Transformer operation to all input points without downsampling, it is easy to design a decoder for the segmentation task. As discussed in Sec. II-A, the authors first use two cascaded feed-forward networks to embed input points into the high-dimension feature space. And then they employ four Transformer blocks to improve the feature representation. Moreover, inspired by the Laplacian matrix in Graph convolution networks [77], they design a novel Offset-Attention (OA) mechanism to replace the standard self-attention for better performance. Finally, PCT achieves 93.2% overall accuracy on the ModelNet40 classification dataset, and 86.4% part-average Intersection-over-Union on the ShapeNet part segmentation dataset. The effectiveness of the proposed OA module is also verified by the 1.0% overall accuracy improvement in point cloud classification.

PointASNL [28] combines local information and global information in parallel for point cloud understanding, inspired by the non-local operation in images [99]. Specifically, it first proposes an attention-based Adaptive Sampling (AS) module

to replace the FPS algorithm [1]. And then it designes a Local-Nonlocal (LNL) module to extract local and global features for each sampling point. The local features are aggregated by employing appealing methods (e.g., PointNet++ [1], PointConv [100]) in a local neighborhood, and the global features are extracted by applying the self-attention mechanism to all sampling points. Finally, a channel-wise summation with a nonlinear convolution is used to fuse extracted local and global features. After all, PointASNL reaches 93.2% overall classification accuracy on ModelNet40 [101].

Masked Point Modeling (MPM) [29] is proposed to help pre-train pure Transformer-based models for point cloud classification. It is inspired by the concept of BERT [78] and masked autoencoder [102]. Specifically, a point cloud is first divided into several local point patches. And then a mini-PointNet is utilized to get the embedded feature (which can be regarded as tokens) for each patch. Like [29], some tokens are randomly discarded (masked) and the rest are fed to the Transformer network, to recover the masked point tokens. The training procedure is totally self-supervised since the masked point tokens have ground truth. With 8192 points as input, Point-BERT can achieve 93.8% overall accuracy on ModelNet40.

Dual Transformer Network (DTNet) [35] is proposed to encode contextual dependencies between input points from the perspectives of both position and channel for point cloud classification and segmentation. Based on this idea, two parallel feature refinement branches are constructed. The first one is a standard point-wise Transformer, which can capture long-range spatial context dependencies among features. The second branch is a channel-wise Transformer, measuring the similarity of different channels, with the same architecture as the first one. The final refined feature map is obtained via element-wise sum operation on the outputs of the above two branches.

In contrast to the standard self-attention mechanism, a centroid attention mechanism [38] takes N inputs and outputs M elements (M is smaller than N). The output elements can be regarded as centroids of the input sequence. Please see Sec. II-C for the detailed description. The authors argue that the standard self-attention can be seen as the special case of their centroid attention when M equals N. For point cloud processing, they propose Centroid Transformer by stacking multiple self-attention layers and centroid attention layers. Centroid Transformer reaches 93.2% overall accuracy on ModelNet40. For the task of point cloud reconstruction, the centroid attention block is demonstrated to be capable of achieving better reconstruction results on ShapeNet part [103] and ShapeNet Core13 dataset [104], with less network parameters than the Dynamic Routing module in 3D Capsule Network [105].

Given the fact that the Multi-Head Self-Attention (MSA) operation is costly in point cloud analysis, Yang et al. [40] design a lightweight yet high-performance Group Shuffle Attention (GSA) module. The GSA is comprised of Group Attention and channel shuffle [106]. In detail, the point feature map is first divided into a small number of groups. And then multi-head self-attention operations are performed within each group, finally followed by the channel shuffle operation. Note that since GSA utilizes a shared projection weight matrix to generate the $Query$ and Key matrices and uses an ELU activation to generate the $Value$ matrix, it is more parameter-efficient than the standard MSA. As one of the self-attention variants, the architecture of GSA is also detailed in Sec. V. Equipped with GSA, Point Attention Transformers (PATs) for point cloud reasoning are developed and show promising results on tasks of shape classification, indoor scene segmentation and gesture recognition.

Zhang et al. [41] propose a pure Transformer-based point cloud learning backbone, taking 3D voxels as the input, termed Point-Voxel Transformer (PVT). Inspired by the recent Swin Transformer [17], a Sparse Window Attention (SWA) operation is designed to perform the self-attention within non-overlapping and shifted 3D voxel windows respectively. A relative-attention (RA) operation is also introduced to compute fine-grained features of points. With the above two designed modules, PVT can take advantage of both point-based and voxel-based networks in one pure Transformer architecture. Similarly, Lai et al. [47] propose Stratified Transformer to explicitly encode long-range contexts. It also extends Swin Transformer [17] to point cloud processing by 3D voxelization. The main difference from the PVT is that Stratified Transformer takes both dense local points and sparse distant points as the key vectors for each $query$ vector. This operation is beneficial to message passing among cubic windows and further global information capturing. Both PVT and Stratified Transformer achieve 86.6% pIoU for part segmentation on ShapeNet dataset. However, Stratified Transformer performs better for semantic segmentation, surpassing PVT by 4.7% mIoU on S3DIS dataset.

To relieve the problem of the expensive computation cost, Patch ATtention (PAT) [82] computes attention maps in linear complexity to the input size. The core idea is that each input point cloud (N points) would be over-segmented into several patches (M patches) via the K-Means algorithm. Compared to the number of points N, the number of these segmented patches M is much smaller. A feature map $B \in R^{M \times D}$ for these patches is then computed and replaces the Key matrix ($\in R^{N \times D}$) in the traditional self-attention formula, where D is the dimension of embedded feature space. In this way, the computational complexity can be reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(NM)$. On the basis of the PAT, an efficient point cloud processing framework, PatchFormer, is proposed. According to experiments in [82], PatchFormer can achieve competitive performance in point cloud classification, and is over $9\times$ faster than previous 3D Transformers.

Aiming at enhancing feature interactions between multiple levels and scales, Han et al. [88] propose a Multi-level Multi-scale Point Transformer (MLMSPT) for efficient point representation learning. They combine the idea of feature pyramid networks [107] and the self-attention mechanism. The input of the MLMSPT is a group of feature maps, which are produced by an MLP-based point feature embedding network on three point sets with different sampling resolutions. A Point Pyramid Transformer block is then designed to extract multi-scale representations. For each resolution branch, a Multi-

level Transformer (MLT) block is employed to encode rich relationships among input points, by taking in a concatenated feature map from multiple Point Pyramid Transformer layers. After MLT, feature maps from three branches are then concatenated together and sent into a Multi-scale Transformer block to extract contextual information from multiple scales, following PF-Net [108]. Cross-Level Cross-Scale Cross-Attention Network (3CROSSNet) [27] also has the similar architecture, as shown in Sec. II-A.

2) Object Detection: Thanks to the rapid development of 3D point cloud scanners, 3D object detection is becoming a more and more popular research topic. Similar to the 2D object detection task, 3D object detectors aim to output 3D bounding boxes with point clouds as input data. Recently, Carion et al. [11] introduces the first Transformer-based 2D object detector, DETR. It proposes to combine Transformers and convolutional neural networks (CNN) to eliminate non-maximum suppression (NMS). Since then, Transformer-related works have also shown a flourishing growth in the field of point cloud-based 3D object detection.

On the basis of VoteNet [109], Xie et al. [49], [110], for the first time, introduce the self-attention mechanism of Transformers into the task of 3D object detection in indoor scenes. They propose the Multi-Level Context VoteNet (MLCVNet) to improve detection performance by encoding contextual information. In their papers, each point patch and vote cluster are regarded as tokens in Transformers. And then the self-attention mechanism is utilized to strengthen the corresponding feature representations via capturing relations within point patches and vote clusters, respectively. Due to the integration of the self-attention modules, MLCVNet achieves better detection results than its baseline model on both ScanNet [75] and SUN RGB-D datasets [74]. Chen et al. [90] propose PQ-Transformer to detect 3D objects and predict room layouts simultaneously. The whole framework is also based on VoteNet, and a Transformer decoder is utilized to enhance proposal features. With the assistance of room layout estimation and refined features by the Transformer decoder, PQ-Transformer attains the mAP@0.25 of 67.2% on ScanNet.

To achieve effective feature learning, Pan et al. [31] propose a pure Transformer-based backbone, Pointformer, whose architecture follows the U-Net fashion. As shown in Sec. II-A, three Transformer-based blocks are introduced in Pointformer: Local Transformer (LT), Local-Global Transformer (LGT) and Global Transformer (GT). Similar to MLCVNet, these blocks are designed to enhance feature representative with the aid of encoding long-range dependencies of Transformers. The proposed Pointformer improves detection performance on both indoor datasets (SUN RGB-D [74] and ScanNet V2 [75]) and outdoor datasets (nuScenes [111] and KITTI [112]).

The above methods employ the hand-crafted grouping scheme to get features for object candidates by learning from points merely within the corresponding local regions. However, Liu et al. [50] argue that the point grouping operation within limited regions tends to hinder the performance of 3D object detection. Thus, they present a group-free framework with the aid of the attention mechanism in Transformers. The core idea is that the features of an object candidate should come from all the points in the given scene, instead of a subset of the point cloud. After obtaining object candidates, their method first leverages a self-attention module to capture contextual information between the object candidates. They then design a cross-attention module to refine the object features with the information of all the points. With the improved attention stacking scheme, their detector achieves the mAP@0.25 of 69.1% on the ScanNet dataset.

Inspired by DETR [11] in 2D object detection, an end-to-end 3D DEtection Transformer network, termed 3DETR [51], is first proposed to formulate 3D object detection as a set-to-set problem. Borrowing ideas from both DETR [11] and VoteNet [109], 3DETR also follows the general encoder-decoder fashion. In the encoder part, sampled points and the corresponding features extracted by MLP are directly fed into a Transformer block to refine the features. In the decoder part, these features go through a parallel Transformer-fashion decoder and are turned into a set of object candidate features. These object candidate features are finally used to predict 3D bounding boxes. After all, 3DETR improves VoteNet by 9.5% AP_{50} and 4.6% AP_{25} on ScanNetV2 and SUN RGB-D respectively.

As known, images can provide complementing information for object detection from 3D point clouds [113]. Wang et al. [114] focus on exploring the multi-modal fusion strategy. They propose an end-to-end Transformer architecture to fuse point clouds and images for 3D object detection in indoor scenes, termed Bridged Transformer (BrT). Considering the heterogeneous geometrics of point clouds and images, they do not directly interact with each other by simply applying attentions on them. Instead, point and image patch tokens are both fed into the Bridged Transformer layers. And object queries are utilized to bridge information communication between points and images. Benefiting from this bridged design, BrT reaches 71.3% mAP@0.25 on ScanNetV2 validation set.

Apart from the above methods focusing on indoor scenes, Sheng et al. [66] propose a Channel-wise Transformer based two-stage framework (CT3D) to improve 3D object detection performance in outdoor LiDAR point clouds. The input of the channel-wise Transformer comes from a Region Proposal Network (RPN). Moreover, the Transformer network consists of two sub-modules: the proposal-to-point encoding module and the channel-wise decoding module. The encoding module first takes the proposals and their corresponding 3D points as the input. And then it extracts the refined point features through a self-attention-based block. The channel-wise decoding module transforms the extracted features from the encoder module into a global representation through a channel-wise re-weighting scheme. And finally, Feed-Forward Networks (FFNs) are performed for detection predictions. In this way, CT3D achieves 81.77% AP in the moderate car category on the KITTI test set.

In a similar paradigm to DETR [11], Bai et al. [115] propose a LiDAR and Camera fusion based 3D object detector based on Transformers, called TransFusion. In TransFusion, the attention mechanism is employed to adaptively fuse features from images. It aims to relieve the problem of bad association between LiDAR points and image pixels established by

calibration matrices. CAT-Det [91] is also proposed to fuse LiDAR point clouds and RGB images more efficiently for 3D object detection performance boosting. A Pointformer and an Imageformer are first introduced in the branches of the point cloud and image respectively to extract multi-modal features. A Cross-Modal Transformer (CMT) module is then designed to combine the features from the above two streams. With the performance of 67.05% mAP on the KITTI test split, CAT-Det becomes the first multi-modal solution that significantly surpasses LiDAR-only ones.

Temporal-Channel TRansformer (TCTR) [72] is proposed to process 3D Lidar-based video for effective object detection in autonomous driving. The key idea is based on the observation that adjacent frames can provide contextual information to the current frame. Instead of merely taking the current frame t point cloud as input, it proposes to include the former T frames to assist in object detection for frame t. Specifically, the input raw point clouds are first transformed into images. And then TCTR is designed to extract and aggregate features from multiple frames, by encoding the temporal-channel domain and spatial-wise relationships along with the continuous frames.

3) Object Tracking: 3D object tracking takes two point clouds (i.e., a template point cloud and a search point cloud) as input. It outputs 3D bounding boxes of the target (template) in the search point cloud. It involves feature extraction of point clouds and feature fusion between template and search point clouds.

Cui et al. [52] argue that most existing tracking approaches do not consider the attention changes of object regions during tracking. That is, different regions in the search point cloud should contribute different importance to the feature fusion process. Based on this observation, they present a LiDAR-based 3D Object Tracking with a TRansformer network (LTTR). This method is able to improve the feature fusion of template and search point clouds by capturing attention changes over tracking time. Specifically, they first build a Transformer encoder to improve the feature representation of template and search point clouds separately. And then they employ the cross-attention mechanism to build a Transformer decoder. It can fuse features from the template and search point clouds by capturing relations between the given two point clouds. Benefiting from the Transformer-based feature fusion between template and search point clouds, LTTR reaches 65.8% mea Precision on KITTI tracking dataset. Zhou et al. [53] also propose a Point Relation Transformer (PRT) module to improve feature fusion in their coarse-to-fine Point Tracking TRansformer (PTTR) framework. Similar to LTTR, PRT employs self-attention and cross-attention to encode relations within and between point clouds respectively. The difference is that PRT utilizes the Offset-Attention [8] to relieve the impact of noise data. After all, PTTR surpasses LTTR by 8.4% and 10.4% in terms of average Success and Precision, becoming a new SOTA on the KITTI tracking benchmark.

Unlike the above two approaches which focus on the feature fusion step, Shan et al. [54] introduce a Point-Track-Transformer (PTT) module to enhance the feature representation after the feature fusion step. Features from the fusion step and the corresponding point coordinates are mapped into the embedding space. A position encoding block is also designed to capture positional features by the KNN algorithm and a MLP layer. With the above two embedded semantic and positional features as input, a self-attention block is finally applied to get more representative features. To verify the effectiveness of the proposed PTT, authors integrate it into the seeds voting and proposal generation stages of the P2B [116] model and get the PTT-Net. As demonstrated by experiments, PTT-Net improves P2B by 9.0% in terms of Precision on KITTI for the car category.

4) Registration: Given two point clouds as input, the aim of point cloud registration is to find a transformation matrix to align them.

Wang et al. [55] introduce the Transformer encoder into the task of point cloud registration by designing their Deep Closest Point (DCP) model. As normal, the input unaligned point clouds are first sent to a feature embedding module, such as PointNet [2] and DGCNN [9], to transfer 3D coordinates into a feature space. A standard Transformer encoder is then applied to perform context aggregation between two embedded features. Finally, authors utilize a differentiable Singular Value Decomposition (SVD) layer to compute the rigid transformation matrix. DCP is the first work that employs the Transformer model to improve the feature extraction of point clouds in registration. With the same paradigm, Wang et al. [56] also deploy Transformer layers to refine the point-wise features extracted by EdgeConv [9] layers, capturing the long-term relationship between point clouds. The resulted network, termed STORM, achieves better performance than DCP for partial registration on ModelNet40 dataset. Similarly, Fischer et al. [57] also leverage multi-head self- and cross-attention mechanism to learn contextual information between target and source point clouds. But their method focuses on processing outdoor scenes, e.g., the KITTI dataset [112].

To find more robust correspondences between two point clouds, Fu et al. [58] present the first deep graph matching-based framework (RGM) to perform robust point cloud registration, which is less sensitive to outliers. During the graph establishment, they employ Transformer encoders to get the soft edges of two nodes within a graph. With the generated soft graph edges, better correspondences can be obtained for the overlapping parts when registering partial-to-partial point clouds. The effectiveness of the proposed Transformer-based edge generator is demonstrated by performance drop on ModelNet40 when replacing it with full connection edges and sparse connection edges.

To address the problem of indistinct feature extraction caused by the shallow-wide Transformer architecture, Chen et al. [59] propose Deep Interaction Transformer (DIT) to improve feature discrimination. They carefully design three novel modules to perform feature extraction and correspondence confidence evaluation. To get good representations of each input point cloud, a Point Cloud Structure Extractor (PSE) is presented. It employs the Transformer encoder to model global relations, and proposes a Local Feature Integrator (LFI) to encode structural information. The extracted features (F_X, F_Y) of two input point clouds are then fed into a

deep-narrow Point Feature Transformer (PFT), to establish comprehensive associations. Moreover, they insert a positional encoding network to encode relative position information between points. In such a way, feature representations (Ψ_X, Ψ_Y) with richer information can be obtained. Given two features and the established correspondences, a Geometric Matching-based Correspondence Confidence Evaluation (GMCCE) is designed to filter out bad correspondence with low confidence values. With more representative features extracted by the full Transformer network, DIT outperforms previous methods, achieving $1.1e-8$ in terms of t_{MAE} on clean point clouds of ModelNet40 [101].

Recently, Yew et al. [69] argue that explicit feature matching and outlier filtering via RANSAC in point cloud registration can be replaced with attention mechanisms. They thus design an end-to-end Transformer framework, termed REGTR, to directly find point cloud correspondences. In REGTR, point features from a KPconv [81] backbone are fed into several multi-head self- and cross- attention layers for relationship capturing. With the above simple design, REGTR becomes the current state-of-the-art point cloud registration method on the ModelNet40 [101] and 3DMatch [117] datasets. Similarly, Qin et al. [68] also utilize the self- and cross-attention to find robust superpoint correspondences in their GeoTransformer. In terms of Registration Recall, both REGTR and GeoTransformer achieve 92.0% on 3DMatch dataset. However, GeoTransformer surpasses REGTR by 10.2% on 3DLoMatch [118] dataset.

5) Point Cloud Video Understanding: The 3D world around us is consistent and dynamic in time, which cannot be fully represented by traditional single-frame and fixed point clouds. In contrast, point cloud videos, a set of point clouds captured in a fixed frame rate, could be a promising data representation of our real physical world. It is much more important for intelligent systems to understand point cloud videos to better interact with the world. Point cloud video understanding involves processing a time sequence of 3D point clouds which has a long-range relationship with each other. Thus, Transformers could be a promising choice to process point cloud videos, since they are good at dealing with global long-range interactions.

Based on the above observation, Fan et al. [60] propose Point 4D Transformer network, termed P4Transformer, to process point cloud videos for action recognition. To extract the spatial-temporal local features of a point cloud video, the input data are first represented by a set of spatial-temporal local areas. And then a point 4D convolution is used to encode features for each local area. After that, authors design a Transformer encoder to receive and integrate the features of local areas via capturing long-range relationships across the entire video. P4Transformer is successfully applied into the task of 3D action recognition and 4D semantic segmentation from point clouds. It achieves higher results than PointNet++-based methods on many benchmarks (e.g., the MSR-Action3D [119], the NTU RGB+D 60 [120] and 120 [121] datasets for 3D action recognition, and the Synthia 4D [86] dataset for 4D semantic segmentation). P4Transformer demonstrates the effectiveness of Transformers on point cloud video understanding.

B. Low-level Task.

The input data of low-level tasks is usually the raw scanned point cloud with occlusion, noise, and uneven densities. Thus, the final goal of low-level tasks is to get a high-quality point cloud, which could contribute to high-level tasks. Some typical low-level tasks include point cloud downsampling [39], upsampling [34], denoising [32], [61], completion [46], [62]–[65], [70] and so on.

1) Downsampling: Given a point cloud with N points, downsampling methods aim at outputting a smaller size of point cloud with M points, while remaining the geometric information of the given point cloud. Leveraging the powerful learning ability of Transformers, wang et al. [39] propose Light-weight Transformer Network (LighTN), downsampling point clouds in a task-oriented manner. As shown in Sec. II-C, it first removes the position encoding, and then uses a small-size shared linear layer as the embedding layer. Moreover, the MSA module is replaced with a single head self-correlation layer. Experimental results demonstrate the above strategies significantly reduce the computational cost while preserving the capability of feature learning. 86.18% classification accuracy can still be attained while only 32 points are sampled. Moreover, the designed lightweight Transformer network is a plug-and-play module, which can be easily inserted into other related networks.

2) Upampling: Contrary to downsampling, upsampling methods aim to output a point cloud with a bigger size than the input point cloud. The upsampled points are expected to lie on the underlying surfaces of the objects represented by the given sparse point clouds. PU-Transformer [34] is the first work to apply the Transformer-based model to the task of point cloud upsampling. The core idea is to activate the strong capability of Transformer encoders in point cloud feature representation by designing two novel blocks in the Transformer encoders. The first block is the Positional Fusion block (PosFus), which aims at capturing local position-related information of point cloud data. The second one is the Shifted Channel Multi-head Self-Attention (SC-MSA) block. It is designed to address the problem of the lack of connection between the outputs of different heads in conventional MSA. Please see more details about the SC-MSA in Sec. V. PU-Transformer shows the promising potential of Transformer-based models in point cloud upsampling.

3) Denoising: Denoising approaches take point clouds corrupted by noise as the input, and output clean point clouds by utilizing the local geometry information. Xu et al. [32] propose Transformer-based Denoising Network (TDNet) for point cloud in the encoder-decoder fashion. Taking each point as a word, they improve NLP Transformer [3] to make it suitable for point cloud feature extraction. The Transformer-based encoder can map the input point cloud into a high-dimensional feature space and further learn the semantic relationship among points. With the extracted feature from the encoder, the latent manifold of the input noise point cloud can be obtained. Finally, a clean point cloud can be generated by sampling each patch manifold.

Another category of point cloud denoising is to filter out noise points directly from the input point clouds. For instance, some Lidar point clouds could contain a huge number of virtual (noise) points. These points are produced by the specular reflections of glass or other kinds of reflective materials. To detect these reflective noise points, Gao et al. [61] first project the input 3D LiDAR point cloud into a 2D range image. And then a Transformer-based auto-encoder network is employed to predict a noise mask to indicate the points coming from reflection.

4) Completion: In most 3D practical applications, it is usually difficult to get complete point clouds of objects or scenes due to occlusion from other objects or self-occlusion. This issue makes point cloud completion an important low-level task in the field of 3D vision. The complete point cloud contains more geometrical information about objects, which can be used to help computers understand the physical world better.

PoinTr proposed in [62], for the first time, converts point cloud completion to a set-to-set translation task. Specifically, authors claim that the input point cloud can be represented by a set of groups of local points, termed "point proxies". Taking a sequence of point proxies as the input, a geometry-aware Transformer block is carefully designed to generate the point proxies of the missing parts. In a coarse-to-fine fashion, FoldingNet [122] is finally employed to produce points based on the predicted point proxies. The geometry-aware Transformer block is a plug-and-play module, which can capture both the semantic and geometric relationship among points. PoinTr attains 8.38 Average L_1 Chamfer Distance (CD) on the PCN dataset [123].

Different from PointTr, Xiang et al. [63] propose to formulate the task of point cloud completion as the growth of 3D points in a snowflake-like fashion. Based on this insight, SnowflakeNet is presented to focus on recovering fine geometric details, such as corners, sharp edges and smooth regions, of the complete point cloud. The core idea is to combine Snowflake Point Deconvolution (SPD) layer with the skip-Transformer to better guide the point splitting process. SPD can generate multiple points from a given point. Skip-Transformer is capable of capturing both contexts and spatial information from the given point and the generated points. With the skip-Transformer integrated, the SPD layers are capable of modeling structure characteristics, thus producing more compact and structured point clouds. Benefiting from the snowflake-like idea and the skip-Transformer, SnowflakeNet surpasses PoinTr by 1.17 Average L_1 Chamfer Distance (CD) on the PCN dataset.

Due to the partial scanned data, robotic grasping methods often suffer from wrong grasping estimation. To solve this issue, Chen et al. [64] present a robotic grasping-oriented shape completion model, termed TransSC. A Transformer-based Multi-Layer Perception (TMLP) module is designed to extract better point-wise feature representations. And then a manifold-based decoder is employed to produce the complete point clouds by decoding the point features. Lin et al. [46] also leverage a Transformer encoder to improve feature representation in their point cloud completion network, termed PCTMA-Net. Similar to TransSC, they claim that the Transformer-based embedding network can extract more discriminate features for each point than MLP-based networks. Liu et al. [70] also integrate self-attention and cross-attention to enhance feature extraction in their proposed dynamic Transformer-based point cloud completion framework.

Instead of working directly on the point cloud, Vector Quantized Deep Implicit Functions (*VQDIF*) proposed in [65] introduces a novel 3D sparse representation. It converts the 3D point cloud to a set of discrete 2-tuples. Accordingly, they design a VQDIF encoder and decoder to perform transformation between the 3D point cloud and the proposed 2-tuples. The sequences of 2-tuples features from partial observations can then be fed into a Transformer-based autoregressive model, i.e., ShapeFormer, to generate complete feature sequences. Next, these sequences are then projected to a feature grid by the VQDIF decoder. Finally, a 3D-Unet [124] is employed to generate local deep implicit functions of objects' whole shapes.

V. 3D Self-attention Variants

Based on the standard self-attention module, there are many variants proposed to improve the performance of Transformers in 3D point cloud processing, as shown in Fig. 4 and 5. According to Sec. II-B, we categorize the relevant variants into two parts: *Point-wise Variants* and *Channel-wise Variants*.

A. Point-wise Variants

Point Attention (P-A) network [6] (Fig. 4(a)) and Attentional ShapeContextNet (A-SCN) network [7] (Fig. 4(b)) design different residual structures in their Transformer encoders. The former strengthens the connection between the output and input of the module, while the later establishes the relationship between the output and the $Value$ matrix of the module. And relevant experiments have demonstrated that the residual connection is necessary in order to learn a good model [7].

Inspired by the Laplacian matrix $L = D - E$ in Graph convolution networks [77], Point Cloud Transformer (PCT) [8] further proposes an Offset-Attention (OA) module (Fig. 4(c)). It calculates the offset (difference) between the Self-Attention (SA) features and the input features X by matrix subtraction, which is analogous to a discrete Laplacian operation. Additionally, it refines the normalization of the similarity matrix by replacing *Scale + Softmax* (SS) with *Softmax + L_1 Norm* (SL) operation. It is able to sharpen the attention weights and reduce the influence of noise. Based on the Offset-Attention, Zhou et al. [53] propose a Relation Attention Module (RAM) which has a similar structure as the OA module. The difference is that it first projects $Query$, Key and $Value$ matrices into latent feature spaces by linear layers. Then, instead of generating the $Attentionmap$ by multiplying the $Query$ and Key matrices directly, it applies the L_2 normalization to the $Query$ and Key matrices. This operation can prevent the dominance of a few feature channels with extremely large magnitudes. Ablation experiments in [53] demonstrate that the L_2 normalization is able to improve the model performance.

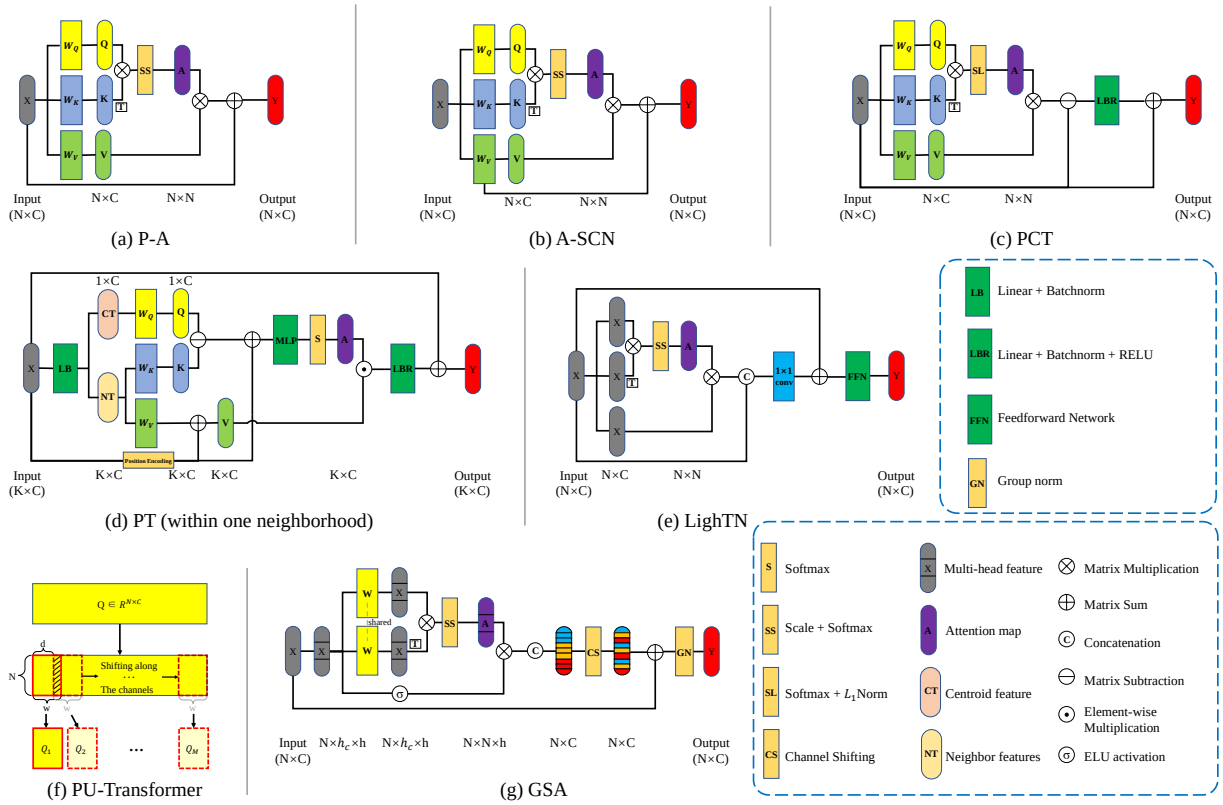

Fig. 4: Architectures of Point-wise Variants.

Point Transformer (PT) [4] (Fig. 4(d)) introduces the vector subtraction attention operator to its Transformer network, replacing the commonly-used scalar dot-product attention. Compared with the scalar attention, vector attention is more expressive since it supports adaptive modulation of individual feature channels, not just whole feature vectors. This kind of expression appears to be very beneficial in 3D data processing [4]. Point Transformer utilizes the subtraction-form vector attention to achieve the local feature aggregation. The attention map is generated by merely building the connections between the centroid feature and its neighbor feature, instead of measuring the similarity between any two point features within a neighborhood. Additionally, 3D Convolution-Transformer Network (3DCTN) [5] conducts a detailed investigation on self-attention operators in 3D Transformers, including the scalar attention and different forms of vector attention.

As mentioned in Sec. II-C, LighTN [39] presents a self-correlation module, to reduce the computational cost. As shown in Fig. 4(e), it eliminates the projection matrices, W_Q, W_K, and W_V simultaneously in the self-attention mechanism. Only the input self-correlation parameters are used to generate attention features. According to Eq. 4, we can see that the self-correlation mechanism generates a symmetry attention map $X \cdot X^T$, which satisfies the permutation invariance in point cloud processing [39]. The authors also conduct a series of ablation studies, removing different projection matrices, to demonstrate the effectiveness of the proposed self-correlation mechanism.

PU-Transformer [34] proposes a Shifted Channel Multi-head Self-Attention (SC-MSA) block to improve the MSA mechanism. Specifically, despite the rich information captured by MSA, only feature dependencies within the same head can be estimated, while lacking information propagation between different heads. To address this issue, as shown in Fig. 4(f), PU-Transformer applies a window (dashed square) shift along the channels to ensure that any two consecutive splits have a uniform overlap area. Compared with the independent splits of the regular MSA, SC-MCA is able to enhance the channel-wise relations in the output features.

Group Shuffle Attention (GSA) proposed in [40] has two improvements compared with the standard MSA. The first one is that GSA is a parameter-efficient self-attention mechanism. It uses a shared projection matrix W to generate the $Query$ and Key matrices, and uses a non-linearity σ to generate the $Value$ matrix:

$$Attn_{\sigma}(X) = softmax(\frac{Q \times K^T}{\sqrt{C}}) \times \sigma(X), \tag{5}$$

where $Q = K = X \times W$ and C is the dimension of X. In this way, GSA is able to reduce the computational costs of the self-attention operation. The second one is that GSA introduces channel shuffle to MSA, which enhances the information flow between heads. As shown in Fig. 4(g), unlike PU-Transformer [34], it re-groups the channels by rewriting each point feature.

B. Channel-wise Variants

Dual Transformer Network (DT-Net) [35] proposes a channel-wise MSA, applying the self-attention mechanism to

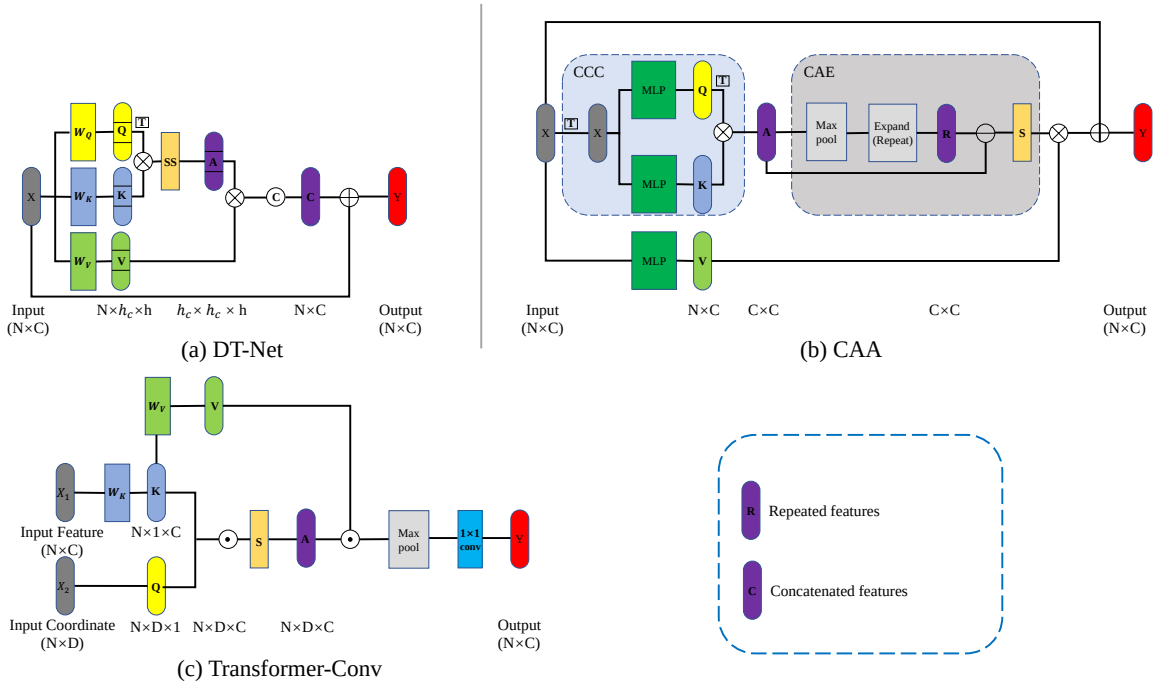

Fig. 5: Architectures of Channel-wise Variants.

the channel space. As shown in Fig. 5(a), unlike the standard self-attention mechanism, channel-wise MSA multiplies the transposed $Query$ matrix and Key matrix. By this means, the attention map can be generated to measure the similarities between different channels, as described in Eq. 3.

As shown in Fig. 5(b), the Channel-wise Affinity Attention (CAA) module [36] utilizes a similar approach, Compact Channel-wise Comparator block (CCC), to generate the similarity matrix between different channels. Moreover, it designs a Channel Affinity Estimator block (CAE) to generate the affinity matrix, strengthening the connection between distinct channels and avoiding aggregating similar/redundant information. The $Value$ matrix is generated by an MLP layer, and the final feature map is obtained by multiplying the affinity matrix and $Value$ matrix. Additionally, the CAA module uses a regular skip connection between the input and output feature map.

The Transformer-Conv module proposed in [37] learns the potential relationship between feature channels and coordinate channels. As shown in Fig. 5(c), The $Query$ matrix and Key matrix are generated by coordinates and features of the point cloud respectively. And then the similarity matrix can be produced by a relation function β (e.g., element-wise multiplication) and channel softmax operation. Different from the above methods, the $Value$ matrix in the Transformer-Conv module is generated from the Key matrix by linear projection. This operation is able to establish a response relationship between the $Value$ matrix and similarity matrix. And such a relationship can be captured by multiplying the similarity matrix and $Value$ matrix in an element-wise manner. Lastly, the final feature map can be generated by using a channel max-pooling and further 1×1 convolution.

VI. Comparison and Analysis

This section briefly gives an overall comparison and analysis of 3D Transformers on several main-stream tasks, including classification, part segmentation, semantic segmentation and object detection.

A. Classification & Segmentation

3D point cloud classification and segmentation are two fundamental yet challenging tasks, in which Transformers have played a key role. Classification can most reflect the ability of feature extraction of the networks. Thus, we first summarize these 3D Transformers according to the classification task. Table I shows the classification accuracy of different methods on the ModelNet40 [101] dataset. For fair comparisons, input data and input size are also shown. We use the Overall Accuracy (OA) as the evaluation metric, which is widely adopted.

From the table, we can see the recent proliferation of Transformer-based point cloud processing methods since 2020, when the Transformer architecture was first employed in image classification by ViT [13]. Due to the strong ability of global information aggregating, Transformers rapidly assume a leading position in this task. For the performance, most 3D Transformers achieve the classification accuracy of around 93.0%. The newest PVT [41] pushes the limit to 94.0%, which surpasses most non-Transformer algorithms of the same period. As an emerging technology, the success of the Transformer in point cloud classification demonstrates its superiority and great potential in the field of 3D point cloud processing. We also present the results of several state-of-the-art non-Transformer-based methods as reference. As can be seen, the classification accuracy of the recent non-Transformer-based

TABLE I: A comparative analysis between involved point cloud classification methods on the ModelNet40 [101] dataset. OA means overall accuracy. All results quoted are taken from the cited papers. P = points, N = normals.

Method	input	input size	OA(%)
Non-Transformer			
PointNet [2]	P	1024×3	89.2
PointNet++ [1]	P	1024×3	90.7
PointNet++ [1]	P, N	5120×6	91.9
PointWeb [125]	P	1024×3	92.3
SpiderCNN [126]	P, N	1024×6	92.4
PointCNN [127]	P	1024×3	92.5
PointConv [100]	P, N	1024×6	92.5
FPConv [128]	P, N	1024×6	92.5
Point2sequence [129]	P	1024×3	92.6
DGCNN [9]	P	1024×3	92.9
KPConv [81]	P	6800×3	92.9
InterpCNN [130]	P	1024×3	93.0
ShellNet [131]	P	1024×3	93.1
RSMix [132]	P	1024×3	93.5
PAConv [133]	P	1024×3	93.9
RPNet [134]	P, N	1024×6	94.1
CurveNet [135]	P	1024×3	94.2
PointMLP [136]	P	1024×3	94.5
Attention/Transformer			
ShapeContextNet [7]	P	1024×3	90.0
PATs [40]	P	1024×3	91.7
DTNet [35]	P	1024×3	92.9
MLMSPT [88]	P	1024×3	92.9
PointASNL [28]	P, N	1024×6	93.2
PCT [8]	P	1024×3	93.2
Centroid Transformers [38]	P	1024×3	93.2
LFT-Net [30]	P, N	2048×6	93.2
3DMedPT [33]	P	1024×3	93.4
3CROSSNet [27]	P	1024×3	93.5
PatchFormer [82]	P, N	1024×6	93.6
Point Transformer [4]	P, N	1024×6	93.7
Point-BERT [29]	P	8192×3	93.8
CAA [36]	P	1024×3	93.8
PVT [41]	P, N	1024×6	94.0

methods has exceeded 94.0%, and the highest one is 94.5%, achieved by PointMLP [136]. Therefore, it is hard to say which kind of algorithm is the best, and we believe that there will be new breakthroughs of 3D Transformers in the future.

For part segmentation, comparisons are performed on the ShapeNet part segmentation dataset [103]. The commonly used part-average Intersection-over-Union is set as the performance metric. As summarised in Table II, all the Transformer-based methods can achieve the pIOU of around 86%, except for ShapeContextNet [7], which is an early model before 2019. Note that Stratified Transformer [47] achieves the highest 86.6% pIoU among all the comparative methods. And it is also the best model in the task of semantic segmentation on the S3DIS semantic segmentation dataset [98] (Table III).

B. Object detection

3D object detection from point clouds remains to be developed by Transformers. Compared to the above three tasks, there are merely a few Transformer or Attention-based methods proposed. The reason could be that the task of object detection is more complicated than classification. Table IV and V summarise the performance of these Transformer-based networks on two public indoor scene datasets: SUN RGB-D [74] and ScanNetV2 [75]. VoteNet [109] is also reported here as a reference, which is the pioneering work in 3D object detection. In terms of $AP@25$ in the ScanNetV2 dataset, all the Transformer-based methods perform better than VoteNet. Among them, Pointformer [31] and MLCVNet [49] are based on VoteNet, and achieve similar performance. Both of them utilize the self-attention mechanism in Transformers to enhance the feature representations. Instead of leveraging the local voting strategy in the above two approaches, GroupFree3D [50] directly aggregates semantic information from all the points in the scene to extract the features of objects. Its performance of 69.1% demonstrates that aggregating features from all the elements by the self-attention mechanism is a more efficient way than the local voting strategy in VoteNet, MLCVNet, and Pointformer. 3DETR [51], as the first end-to-end Transformer-based 3D object detector, achieves the second best detection performance, 65.0%, in the ScanNetV2 dataset.

VII. Discussion and Conclusion

A. Discussion

As in the 2D field, Transformers also show its superiority in 3D point cloud processing. From the perspective of the 3D tasks, Transformer-based methods mainly focus on high-level tasks, such as classification and segmentation. And they achieve breakthrough improvements in terms of accuracy. We argue the reason is that Transformers are better at extracting global contextual information via capturing long-dependency relationships, which corresponds to the semantic information in high-level tasks. On the contrary, low-level tasks, such as denoising and sampling, focus on exploring local geometric features. From the perspective of the performance, 3D Transformers improve the accuracy of those tasks above and surpass most of the existing methods. However, as shown in Sec. VI, there is still a gap between them and the start-of-the-art non-Transformer-based methods. Therefore, despite the rapid development of 3D Transformers, as an emerging technology, they still need further exploration and improvement.

Based on the properties of Transformers and their successful applications in 2D domain, we point out several potential future directions for 3D Transformers, hoping it will ignite the further development of this technology.

1) Patch-wise Transformers: As mentioned in Sec. II-B, 3D Transformers can be divided into two groups: Point-wise Transformers and Channel-wise Transformers. Moreover, referring to the exploration of Transformers in 2D image processing [80], we are able to further divide Point-wise Transformers into Pair-wise Transformers and Patch-wise Transformers based on the operating form. The former is to calculate the attention weight for a feature vector by a corresponding pair of points, while the latter is by incorporating information from all points in a given patch. Specifically, the self-attention mechanism of pair-wise Transformers can be described as:

$$y_i = \sum_{j \in \Re_i} \alpha(x_i, x_j) \odot \beta(x_j), \tag{6}$$

where y_i is the output feature, $\Re_i$ is the operating scope of the self-attention module, β projects the feature x_j to a new feature space by linear layers, and $\alpha(x_i, x_j)$ is utilized to

TABLE II: A comparative analysis between different point cloud Transformers in terms of pIoU on the ShapeNet part segmentation dataset. pIoU means part-average Intersection-over-Union. All results quoted are taken from the cited papers.

Method	pIoU	air-plane	bag	cap	car	chair	ear-phone	guitar	knife	lamp	laptop	motor-bike	mug	pistol	rocket	skate-board	table
3DMedPT [33]	84.3	81.2	86.0	91.7	79.6	90.1	81.2	91.9	88.5	84.8	96.0	72.3	95.8	83.2	64.6	78.2	83.8
ShapeContextNet [7]	84.6	83.8	80.8	83.5	79.3	90.5	69.8	91.7	86.5	82.9	96.0	69.2	93.9	82.5	62.9	74.4	80.8
DTNet [35]	85.6	83.0	81.4	84.3	78.4	90.9	74.3	91.0	87.3	84.7	95.6	69.0	94.4	82.5	59.0	76.4	83.5
3CROSSNet [27]	85.9	83.8	84.9	86.1	79.8	91.2	70.3	91.1	87.0	85.0	95.9	73.2	94.9	83.2	56.2	76.7	83.0
CAA [36]	85.9	84.5	82.2	86.8	78.9	91.1	74.5	91.4	89.0	84.5	95.5	69.6	94.2	83.4	57.8	75.5	83.5
PointASNL [28]	86.1	84.1	84.7	87.9	79.7	92.2	73.7	91.0	87.2	84.2	95.8	74.4	95.2	81.0	63.0	76.3	83.2
LFT-Net [30]	86.2	83.0	83.9	90.9	79.4	93.1	71.4	92.5	88.6	85.7	95.9	69.3	94.2	85.0	65.6	74.6	85.5
PCT [8]	86.4	85.0	82.4	89.0	81.2	91.9	71.5	91.3	88.1	86.3	95.8	64.6	95.8	83.6	62.2	77.6	83.7
MLMSPT [88]	86.4	84.4	84.7	89.2	80.2	89.4	77.1	92.3	87.5	85.3	96.7	71.6	95.2	84.2	61.3	76.0	83.6
PatchFormer [82]	86.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Point Transformer [4]	86.6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PVT [41]	86.6	85.3	82.1	88.7	82.1	92.4	75.5	91.0	88.9	85.6	95.4	76.2	94.7	84.2	65.0	75.3	81.7
Stratified Transformer [47]	86.6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

TABLE III: A comparative analysis between different point cloud Transformers in terms of mIoU/mAcc/OA on the S3DIS Area 5 semantic segmentation dataset. mIoU means mean classwise intersection over union, mAcc means mean of classwise accuracy, and OA means overall pointwise accuracy. All results quoted are taken from the cited papers.

Method	OA	mIoU	mAcc	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
ShapeContextNet [7]	81.6	52.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PATs [40]	-	60.07	70.83	93.04	98.51	72.28	1.00	41.52	85.05	38.22	57.66	83.64	48.12	67.00	61.28	33.64
PCT [8]	-	61.33	67.65	92.54	98.42	80.62	0.00	19.37	61.64	48.00	76.58	85.20	46.22	67.71	67.93	52.29
PointASNL [28]	87.7	62.6	68.5	94.3	98.4	79.1	0.0	26.7	55.2	66.2	83.3	86.8	47.6	68.3	56.4	52.1
MLMST [88]	-	62.9	-	94.5	98.7	90.6	0.0	21.1	60.0	51.4	83.0	89.6	28.9	70.7	74.2	55.5
LFT-Net [30]	-	65.2	76.2	92.8	96.1	81.9	0.0	37.6	70.3	70.4	73.2	76.0	40.9	78.8	71.0	58.2
PVT [41]	-	67.30	-	91.18	98.76	86.23	0.31	34.21	49.90	61.45	81.62	89.85	48.20	79.96	76.45	54.67
EPT [45]	-	67.5	74.7	91.5	97.4	86.0	0.2	40.4	60.8	66.7	87.7	79.6	73.7	58.6	77.2	57.3
PatchFormer [82]	-	68.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Point Transformer [4]	90.8	70.4	76.5	94.0	98.5	86.3	0.0	38.0	63.4	74.3	89.1	82.4	74.3	80.2	76.0	59.3
Stratified Transformer [47]	91.5	72.0	78.1	-	-	-	-	-	-	-	-	-	-	-	-	-

TABLE IV: A comparative analysis between different point cloud Transformers in terms of AP on the ScanNetV2 and SUN RGB-D object detection datasets. AP means Average Precision. All results quoted are taken from the cited papers.

Method	**ScanNetV2**		**SUN RGB-D**	
	AP_{25}	AP_{50}	AP_{25}	AP_{50}
VoteNet [109]	58.6	33.5	57.7	-
3DETR [51]	62.7	37.5	56.8	30.1
Pointformer [31]	64.1	-	61.1	-
MLCVNet [49]	64.5	41.4	59.8	-
3DETR-m [51]	65.0	47.0	59.0	32.7
GroupFree3D [50]	69.1	52.8	63.0	45.2

measure the relationship between x_i and x_j, which can be decomposed as:

$$\alpha(x_i, x_j) = \rho(\gamma(\delta(x_i, x_j))), \tag{7}$$

where ρ is normalization function like softmax, γ is a mapping function that ensures $\delta(x_i, x_j)$ has the same size as $\beta(x_j)$, and δ is a relation function which can be expressed as:

$$\begin{aligned}
&\textit{Concatenation}: \delta(x_i, x_j) = [\varphi(x_i), \psi(x_j)]\,,\\
&\textit{Summation}: \delta(x_i, x_j) = \varphi(x_i) + \psi(x_j),\\
&\textit{Subtraction}: \delta(x_i, x_j) = \varphi(x_i) - \psi(x_j),\\
&\textit{Hadamard product}: \delta(x_i, x_j) = \varphi(x_i) \odot \psi(x_j),\\
&\textit{Dot product}: \delta(x_i, x_j) = \varphi(x_i)^T \psi(x_j),
\end{aligned} \tag{8}$$

where *Dot product* belongs to the scalar attention operator, while the other forms are vector attention operators. The subtraction-form vector attention is used in Point Transformer [4]. From the Eq. 6, we can see that the attention weight $\alpha(x_i, x_j)$ is determined by a corresponding pair of point features x_i and x_j. Pair-wise Transformers have achieved compelling performance in the 2D field and are also commonly used in 3D point cloud processing. Nearly all algorithms in Sec. II-B1 can be considered as pair-wise Transformers, where most of them use *Dot product*.

Zhao et al. [80] also explore a family of patch-wise Transformers in image processing, whose self-attention mechanism can be expressed as:

$$y_i = \sum_{j \in \Re_i} \alpha(x_{\Re_i})_j \odot \beta(x_j), \tag{9}$$

where $x_{\Re_i}$ is the patch of feature vectors in $\Re_i$, α transforms the $x_{\Re_i}$ to a new tensor with the same spatial dimensionality, and $\alpha(x_{\Re_i})_j$ is the j-th feature vector in this tensor. Similar to pair-wise Transformers, $\alpha(x_{\Re_i})$ can also be decomposed as:

$$\alpha(x_{\Re_i}) = \rho(\gamma(\delta(x_{\Re_i}))), \tag{10}$$

and δ can be expressed as three different forms [80]:

$$\begin{aligned}
&\textit{Concatenation}: \delta(x_{\Re_i}) = [\varphi(x_i), [\psi(x_j)]_{\forall j \in \Re_i}]\,,\\
&\textit{Star-Product}: \delta(x_{\Re_i}) = [\varphi(x_i)^T \psi(x_j)]_{\forall j \in \Re_i},\\
&\textit{Dot product}: \delta(x_{\Re_i}) = [\varphi(x_j)^T \psi(x_k)]_{\forall j,k \in \Re_i}.
\end{aligned} \tag{11}$$

By comparing Eq. 6 and 9, we can see that the latter is to aggregate all feature vectors in $\Re_i$ to generate the weight matrix that is applied to $\beta(x_j)$, instead of merely utilizing a pair of features. In this way, patch-wise Transformers are able to enhance the connections among different feature vectors, and extract more robust short- and long-range dependencies. However, since the feature vectors are arranged in a particular

TABLE V: Detailed Performance of mAP@0.25 for each category on the ScanNet V2 object detection dataset.

Method	mAP	cab	bed	chair	sofa	table	door	wind	bkshf	pic	cntr	desk	curt	fridg	showr	toil	sink	bath	ofurn
VoteNet [109]	58.6	36.3	87.9	88.7	89.6	58.8	47.3	38.1	44.6	7.8	56.1	71.7	47.2	45.4	57.1	94.9	54.7	92.1	37.2
3DETR [51]	62.7	50.2	87.0	86.0	87.1	61.6	46.6	40.1	54.5	9.1	62.8	69.5	48.4	50.9	68.4	97.9	67.6	85.9	45.8
Pointformer [31]	64.1	46.7	88.4	90.5	88.7	65.7	55.0	47.7	55.8	18.0	63.8	69.1	55.4	48.5	66.2	98.9	61.5	86.7	47.4
MLCVNet [49]	64.48	42.45	88.48	89.98	87.4	63.50	56.93	46.98	56.94	11.94	56.72	76.05	63.94	60.86	65.91	98.33	59.18	87.22	47.89
3DETR-m [51]	65.0	49.4	83.6	90.9	89.8	67.6	52.4	39.6	56.4	15.2	55.9	79.2	58.3	57.6	67.6	97.2	70.6	92.2	53.0
GroupFree3D [50]	69.1	52.1	92.9	93.6	88.0	70.7	60.7	53.7	62.4	16.1	58.5	80.9	67.9	47.0	76.3	99.6	72.0	95.3	56.4

order in $x_{\Re_i}$, patch-wise Transformers are not permutation-invariant, which may have some negative effects on point cloud processing.

Currently, there is little patch-wise Transformer research in the field of 3D point cloud processing. Considering the advantages of patch-wise Transformers and their outstanding performance in image processing, we believe that introducing patch-wise Transformers to point cloud processing is beneficial to performance improvement.

2) Adaptive Set Abstraction: PointNet++ [1] proposes a Set Abstraction (SA) module to extract the semantic features of the point cloud hierarchically. It mainly utilizes FPS and query ball grouping algorithms to achieve sampling point searching and local patch construction respectively. However, the sampling points generated by FPS tend to be evenly distributed in the original point cloud, while ignoring the geometric and semantic differences between different parts. For example, the tail part of the aircraft is more geometrically complex and distinct than the fuselage part, which makes the former need more sampling points to describe. Moreover, query ball grouping focuses on searching the neighbor points only based on the Euclidean distance. However, it ignores the semantic feature differences among points, which makes it easy to group points with different semantic information into the same local patch. Therefore, developing an adaptive set abstraction is beneficial to improving the performance of 3D Transformers. Recently, there have been several Transformer-based methods in the 3D field exploring adaptive sampling [39]. But nearly none of them makes full use of the rich short- and long-range dependencies generated by the self-attention mechanism. In the field of image processing, Deformable Attention Transformer (DAT) proposed in [137] generates the deformed sampling points by introducing an offset network. It achieves consistently-improved results on comprehensive benchmarks and reduces computational costs. It will be meaningful to present an adaptive sampling method based on the self-attention mechanism for the hierarchical Transformer. Additionally, inspired by the superpixel [138] in the 2D field, we argue that it is feasible to utilize the attention map in 3D Transformers to obtain the "superpoint" [139] for point cloud oversegmentation, converting point-level 3D data into district-level data. In this way, this adaptive clustering technique can be used to replace the query ball grouping method.

3) Self-supervised Transformer Pre-training: Transformers have shown impressive performance on NLP and 2D image processing tasks. However, much of their success stems not only from their excellent scalability but also from large-scale self-supervised pre-training [78]. Vision Transformer [13] performes a series of self-supervision experiments, and demonstrates the potential of the self-supervised Transformer. In the field of point cloud processing, despite the significant progress of supervised point cloud approaches, point cloud annotation is still a labor-intensive task. And the limited labeled dataset hinders the performance improvement of supervised approaches, especially in terms of the point cloud segmentation task. Recently, there have been a series of self-supervised approaches proposed to address these issues, such as Generative Adversarial Networks (GAN) [140], Auto-Encoders (AE) [141], [142], and Gaussian Mixture Models (GMM) [143]. These methods use auto-encoders and generative model to realize self-supervised point cloud representation learning [89]. Their satisfactory performances have demonstrated the effectiveness of the self-supervised point cloud approaches. However, few self-supervised Transformers are currently applied to 3D point cloud processing. With the increase of 3D point cloud datasets, especially large-scale complex point cloud datasets, it is worthwhile to explore the self-supervised 3D Transformers for point cloud representation learning.

Overall, we can see that Transformers have just started to be applied to point cloud-related tasks. And the real power of Transformers in point cloud processing still has much space for deep exploration.

B. Conclusion

Transformer models have attracted widespread attention in the field of 3D point cloud processing, and achieved impressive results in various 3D tasks. In this paper, we have comprehensively reviewed recent Transformer-based networks applied to point cloud-related tasks, such as point cloud classification, segmentation, object detection, registration, sampling, denoising, completion and other practical applications. We first introduce basic definitions of Transformers, and describe the development and applications of the 2D and 3D Transformers briefly. Then we utilize three different taxonomies to categorize the involved methods into multiple groups, analyzing them from multiple perspectives. Additionally, we also investigate a series of self-attention variants that aims to improve the performance and reduce the computational cost. In terms of point cloud classification, segmentation and object detection, a brief comparison of the involved methods is provided in this paper. Finally, we provide three potential future research directions for the development of 3D Transformers. We hope this survey gives researchers a whole view of 3D Transformers, and drives their interest to further improve the performance.

REFERENCES

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

[2] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[4] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 259–16 268.

[5] D. Lu, Q. Xie, L. Xu, and J. Li, "3dctn: 3d convolution-transformer network for point cloud classification," *arXiv preprint arXiv:2203.00828*, 2022.

[6] M. Feng, L. Zhang, X. Lin, S. Z. Gilani, and A. Mian, "Point attention network for semantic segmentation of 3d point clouds," *Pattern Recognition*, vol. 107, p. 107446, 2020.

[7] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional shapecontextnet for point cloud recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4606–4615.

[8] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021.

[9] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[10] H. Wang, Y. Zhu, H. Adam, A. Yuille, and L.-C. Chen, "Max-deeplab: End-to-end panoptic segmentation with mask transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5463–5474.

[11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[12] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8126–8135.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[14] B. Wu, C. Xu, X. Dai, A. Wan, P. Zhang, Z. Yan, M. Tomizuka, J. Gonzalez, K. Keutzer, and P. Vajda, "Visual transformers: Token-based image representation and processing for computer vision," *arXiv preprint arXiv:2006.03677*, 2020.

[15] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 568–578.

[16] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 22–31.

[17] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[18] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[19] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[20] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, "End-to-end video instance segmentation with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8741–8750.

[21] Z. Yao, J. Ai, B. Li, and C. Zhang, "Efficient detr: improving end-to-end object detector with dense prior," *arXiv preprint arXiv:2104.01318*, 2021.

[22] J. Yang, C. Li, P. Zhang, X. Dai, B. Xiao, L. Yuan, and J. Gao, "Focal self-attention for local-global interactions in vision transformers," *arXiv preprint arXiv:2107.00641*, 2021.

[23] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised visual transformers," *arXiv e-prints*, pp. arXiv–2104, 2021.

[24] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, "A survey of visual transformers," *arXiv preprint arXiv:2111.06091*, 2021.

[25] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM Computing Surveys (CSUR)*, 2021.

[26] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on vision transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[27] X.-F. Han, Z.-Y. He, J. Chen, and G.-Q. Xiao, "3crossnet: Cross-level cross-scale cross-attention network for point cloud representation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3718–3725, 2022.

[28] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5589–5598.

[29] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-bert: Pre-training 3d point cloud transformers with masked point modeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[30] Y. Gao, X. Liu, J. Li, Z. Fang, X. Jiang, and K. M. S. Huq, "Lft-net: Local feature transformer network for point clouds analysis," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[31] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7463–7472.

[32] X. Xu, G. Geng, X. Cao, K. Li, and M. Zhou, "Tdnet: transformer-based network for point cloud denoising," *Applied Optics*, vol. 61, no. 6, pp. C80–C88, 2022.

[33] J. Yu, C. Zhang, H. Wang, D. Zhang, Y. Song, T. Xiang, D. Liu, and W. Cai, "3d medical point transformer: Introducing convolution to attention networks for medical point cloud analysis," *arXiv preprint arXiv:2112.04863*, 2021.

[34] S. Qiu, S. Anwar, and N. Barnes, "Pu-transformer: Point cloud upsampling transformer," *arXiv preprint arXiv:2111.12242*, 2021.

[35] X.-F. Han, Y.-F. Jin, H.-X. Cheng, and G.-Q. Xiao, "Dual transformer for point cloud analysis," *arXiv preprint arXiv:2104.13044*, 2021.

[36] S. Qiu, S. Anwar, and N. Barnes, "Geometric back-projection network for point cloud classification," *IEEE Transactions on Multimedia*, vol. 24, pp. 1943–1955, 2022.

[37] G. Xu, H. Cao, J. Wan, K. Xu, Y. Ma, and C. Zhang, "Adaptive channel encoding transformer for point cloud analysis," *arXiv preprint arXiv:2112.02507*, 2021.

[38] L. Wu, X. Liu, and Q. Liu, "Centroid transformers: Learning to abstract with attention," *arXiv preprint arXiv:2102.08606*, 2021.

[39] X. Wang, Y. Jin, Y. Cen, T. Wang, B. Tang, and Y. Li, "Lightn: Light-weight transformer network for performance-overhead tradeoff in point cloud downsampling," *arXiv preprint arXiv:2202.06263*, 2022.

[40] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian, "Modeling point clouds with self-attention and gumbel subset sampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3323–3332.

[41] C. Zhang, H. Wan, S. Liu, X. Shen, and Z. Wu, "Pvt: Point-voxel transformer for 3d deep learning," *arXiv preprint arXiv:2108.06076*, 2021.

[42] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, and C. Xu, "Voxel transformer for 3d object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3164–3173.

[43] S. L. Chenhang He, Ruihuang Li and L. Zhang, "Voxel set transformer: A set-to-set approach to 3d object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[44] Z. Fan, Z. Song, H. Liu, Z. Lu, J. He, and X. Du, "Svt-net: Super light-weight sparse voxel transformer for large scale place recognition," in *AAAI*, 2022.

[45] C. Park, Y. Jeong, M. Cho, and J. Park, "Efficient point transformer for large-scale 3d scene understanding," 2022. [Online]. Available: https://openreview.net/forum?id=3SUToIxuIT3

[46] J. Lin, M. Rickert, A. Perzylo, and A. Knoll, "Pctma-net: Point cloud transformer with morphing atlas-based point generation network for dense point cloud completion," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5657–5663.

[47] L. J. L. W. H. Z. S. L. X. Q. J. J. Xin Lai, Jianhui Liu, "Stratified transformer for 3d point cloud segmentation," in *CVPR*, 2022.

[48] L. Hui, H. Yang, M. Cheng, J. Xie, and J. Yang, "Pyramid point cloud transformer for large-scale place recognition," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6098–6107.

[49] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang, "Mlcvnet: Multi-level context votenet for 3d object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 447–10 456.

[50] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong, "Group-free 3d object detection via transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2949–2958.

[51] I. Misra, R. Girdhar, and A. Joulin, "An end-to-end transformer model for 3d object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2906–2917.

[52] Y. Cui, Z. Fang, J. Shan, Z. Gu, and S. Zhou, "3d object tracking with transformer," *BMVC*, 2021.

[53] C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu, "Pttr: Relational 3d point cloud object tracking with transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[54] S. Jiayao, S. Zhou, Y. Cui, and Z. Fang, "Real-time 3d single object tracking with transformer," *IEEE Transactions on Multimedia*, 2022.

[55] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3523–3532.

[56] Y. Wang, C. Yan, Y. Feng, S. Du, Q. Dai, and Y. Gao, "Storm: Structure-based overlap matching for partial point cloud registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[57] K. Fischer, M. Simon, F. Olsner, S. Milz, H.-M. Groß, and P. Mader, "Stickypillars: Robust and efficient feature matching on point clouds using graph neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 313–323.

[58] K. Fu, S. Liu, X. Luo, and M. Wang, "Robust point cloud registration framework based on deep graph matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8893–8902.

[59] G. Chen, M. Wang, Y. Yue, Q. Zhang, and L. Yuan, "Full transformer framework for robust point cloud registration with deep information interaction," *arXiv preprint arXiv:2112.09385*, 2021.

[60] H. Fan, Y. Yang, and M. Kankanhalli, "Point 4d transformer networks for spatio-temporal modeling in point cloud videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 204–14 213.

[61] R. Gao, M. Li, S.-J. Yang, and K. Cho, "Reflective noise filtering of large-scale point cloud using transformer," *Remote Sensing*, vol. 14, no. 3, p. 577, 2022.

[62] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "Pointr: Diverse point cloud completion with geometry-aware transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 498–12 507.

[63] P. Xiang, X. Wen, Y.-S. Liu, Y.-P. Cao, P. Wan, W. Zheng, and Z. Han, "Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5499–5509.

[64] W. Chen, H. Liang, Z. Chen, F. Sun, and J. Zhang, "Transsc: Transformer-based shape completion for grasp evaluation," *arXiv preprint arXiv:2107.00511*, 2021.

[65] X. Yan, L. Lin, N. J. Mitra, D. Lischinski, D. Cohen-Or, and H. Huang, "Shapeformer: Transformer-based shape completion via sparse representation," *arXiv preprint arXiv:2201.10326*, 2022.

[66] H. Sheng, S. Cai, Y. Liu, B. Deng, J. Huang, X.-S. Hua, and M.-J. Zhao, "Improving 3d object detection with channel-wise transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2743–2752.

[67] Y. Wei, H. Liu, T. Xie, Q. Ke, and Y. Guo, "Spatial-temporal transformer for 3d point cloud sequences," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1171–1180.

[68] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *CVPR*, 2022.

[69] Z. J. Yew and G. h. Lee, "Regtr: End-to-end point cloud correspondences with transformers," in *CVPR*, 2022.

[70] X. Liu, G. Xu, K. Xu, J. Wan, and Y. Ma, "Point cloud completion by dynamic transformer with adaptive neighbourhood feature fusion," *IET Computer Vision*, 2022.

[71] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7077–7087.

[72] Z. Yuan, X. Song, L. Bai, Z. Wang, and W. Ouyang, "Temporal-channel transformer for 3d lidar-based video object detection for autonomous driving," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[73] S. Qiu, Y. Wu, S. Anwar, and C. Li, "Investigating attention mechanism in 3d point cloud object detection," in *2021 International Conference on 3D Vision (3DV)*. IEEE, 2021, pp. 403–412.

[74] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 567–576.

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

[76] X.-Y. Gao, Y.-Z. Wang, C.-X. Zhang, and J.-Q. Lu, "Multi-head self-attention for 3d point cloud classification," *IEEE Access*, vol. 9, pp. 18 137–18 147, 2021.

[77] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[78] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[79] J. T. Rolfe, "Discrete variational autoencoders," *arXiv preprint arXiv:1609.02200*, 2016.

[80] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 076–10 085.

[81] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6411–6420.

[82] Z. Cheng, H. Wan, X. Shen, and Z. Wu, "Patchformer: A versatile 3d transformer based on patch attention," *arXiv preprint arXiv:2111.00207*, 2021.

[83] S. Mehta, M. Ghazvininejad, S. Iyer, L. Zettlemoyer, and H. Hajishirzi, "Delight: Deep and light-weight transformer," *arXiv preprint arXiv:2008.00623*, 2020.

[84] Y. Xu, X. Tong, and U. Stilla, "Voxel-based representation of 3d point clouds: Methods, applications, and its potential use in the construction industry," *Automation in Construction*, vol. 126, p. 103675, 2021.

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

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

[87] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3744–3753.

[88] X.-F. Han, Y.-J. Kuang, and G.-Q. Xiao, "Point cloud learning with transformer," *arXiv preprint arXiv:2104.13636*, 2021.

[89] K. Fu, P. Gao, R. Zhang, H. Li, Y. Qiao, and M. Wang, "Distillation with contrast is all you need for self-supervised point cloud representation learning," *arXiv preprint arXiv:2202.04241*, 2022.

[90] X. Chen, H. Zhao, G. Zhou, and Y.-Q. Zhang, "Pq-transformer: Jointly parsing 3d objects and layouts from point clouds," *IEEE Robotics and Automation Letters*, 2022.

[91] Y. Zhang, J. Chen, and D. Huang, "Cat-det: Contrastively augmented transformer for multi-modal 3d object detection," *CVPR*, 2022.

[92] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[93] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[94] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[95] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[96] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[97] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[98] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1534–1543.

[99] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[100] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9621–9630.

[101] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[102] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021.

[103] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *ACM Transactions on Graphics (ToG)*, vol. 35, no. 6, pp. 1–12, 2016.

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

[105] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, "3d point capsule networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1009–1018.

[106] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[107] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[108] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "Pf-net: Point fractal network for 3d point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7662–7670.

[109] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9277–9286.

[110] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang, "Vote-based 3d object detection with context modeling and sob-3dnms," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1857–1874, 2021.

[111] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[112] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[113] C. R. Qi, X. Chen, O. Litany, and L. J. Guibas, "Imvotenet: Boosting 3d object detection in point clouds with image votes," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4404–4413.

[114] Y. Wang, T. Ye, L. Cao, W. Huang, F. Sun, F. He, and D. Tao, "Bridged transformer for vision and point cloud 3d object detection," *CVPR*, 2022.

[115] X. Z. Q. H. Y. C. H. F. Xuyang Bai, Zeyu Hu and C.-L. Tai, "TransFusion: Robust Lidar-Camera Fusion for 3d Object Detection with Transformers," *CVPR*, 2022.

[116] H. Qi, C. Feng, Z. Cao, F. Zhao, and Y. Xiao, "P2b: Point-to-box network for 3d object tracking in point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6329–6338.

[117] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1802–1811.

[118] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4267–4276.

[119] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *2010 IEEE computer society conference on computer vision and pattern recognition-workshops*. IEEE, 2010, pp. 9–14.

[120] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1010–1019.

[121] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot, "Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2684–2701, 2019.

[122] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Interpretable unsupervised learning on 3d point clouds," *arXiv preprint arXiv:1712.07262*, vol. 2, no. 3, p. 5, 2017.

[123] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 728–737.

[124] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.

[125] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "Pointweb: Enhancing local neighborhood features for point cloud processing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5565–5573.

[126] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102.

[127] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," *Advances in neural information processing systems*, vol. 31, 2018.

[128] Y. Lin, Z. Yan, H. Huang, D. Du, L. Liu, S. Cui, and X. Han, "Fpconv: Learning local flattening for point convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4293–4302.

[129] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8778–8785.

[130] J. Mao, X. Wang, and H. Li, "Interpolated convolutional networks for 3d point cloud understanding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1578–1587.

[131] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1607–1616.

[132] D. Lee, J. Lee, J. Lee, H. Lee, M. Lee, S. Woo, and S. Lee, "Regularization strategy for point cloud via rigidly mixed sample," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 900–15 909.

[133] M. Xu, R. Ding, H. Zhao, and X. Qi, "Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3173–3182.

[134] H. Ran, W. Zhuo, J. Liu, and L. Lu, "Learning inner-group relations on point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 477–15 487.

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

[136] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual mlp framework," *arXiv preprint arXiv:2202.07123*, 2022.
[137] Z. Xia, X. Pan, S. Song, L. E. Li, and G. Huang, "Vision transformer with deformable attention," *arXiv preprint arXiv:2201.00520*, 2022.
[138] L. Zhu, Q. She, B. Zhang, Y. Lu, Z. Lu, D. Li, and J. Hu, "Learning the superpixel in a non-iterative and lifelong manner," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1225–1234.
[139] L. Hui, J. Yuan, M. Cheng, J. Xie, X. Zhang, and J. Yang, "Superpoint network for point cloud oversegmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5510–5519.
[140] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.
[141] M. Gadelha, R. Wang, and S. Maji, "Multiresolution tree networks for 3d point cloud processing," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 103–118.
[142] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards k-means-friendly spaces: Simultaneous deep learning and clustering," in *international conference on machine learning*. PMLR, 2017, pp. 3861–3870.
[143] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," in *International conference on machine learning*. PMLR, 2018, pp. 40–49.