Aberystwyth University

**LCPFormer**
Huang, Zhuoxu; Zhao, Zhiyou; Li, Banghuai; Han, Jungong

*Published in:*
IEEE Transactions on Circuits and Systems for Video Technology

*DOI:*
10.1109/TCSVT.2023.3247506

*Publication date:*
2023

*Citation for published version (APA):*
Huang, Z., Zhao, Z., Li, B., & Han, J. (2023). LCPFormer: Towards Effective 3D Point Cloud Analysis via Local Context Propagation in Transformers. *IEEE Transactions on Circuits and Systems for Video Technology*. https://doi.org/10.1109/TCSVT.2023.3247506
LCPFormer: Towards Effective 3D Point Cloud Analysis via Local Context Propagation in Transformers

Zhuoxu Huang*, Zhiyou Zhao*, Banghuai Li †, and Jungong Han, Member, IEEE.

Abstract—Transformer with its underlying attention mechanism and the ability to capture long-range dependencies makes it become a natural choice for unordered point cloud data. However, local regions separated from the general sampling architecture corrupt the structural information of the instances, and the inherent relationships between adjacent local regions lack exploration. In other words, the transformer only focuses on the long-range dependence, while local structural information is still crucial in a transformer-based 3D point cloud model. To enable transformers to incorporate local structural information, we proposed a straightforward solution based on the natural structure of the point clouds to exploit the message passing between neighboring local regions, thus making their representations more comprehensive and discriminative. Concretely, the proposed module, named Local Context Propagation (LCP), is inserted between two transformer layers. It takes advantage of the overlapping points of adjacent local regions (statistically shown to be prevalent) as intermediaries, then re-weighs the features of these shared points from different local regions before passing them to the next layers. Finally, we design a flexible LCPFormer architecture equipped with the LCP module, which is applicable to several different tasks. Experimental results demonstrate that our proposed LCPFormer outperforms various transformer-based methods in benchmarks including 3D shape classification and dense prediction tasks such as 3D object detection and semantic segmentation. Code will be released for reproduction.

Index Terms—3D vision, Point cloud learning, Transformer, Context propagation

I. INTRODUCTION

INCREASING real-world applications, including robotics, autonomous driving, and augmented reality, enhanced the use of 3D point cloud data due to its precise distance perception and information capacity. However, unlike 2D images in the form of regular pixel grids, it is challenging to effectively ingest features from 3D point clouds due to their sparsity and unstructured nature.

Based on how point clouds are represented, existing feature extraction methods can roughly be divided into three categories: projection-based, voxel-based and point-based methods. Projection-based methods [1]–[5] first project 3D point clouds into a 2D plane and then treat them as pixel representations for processing. Voxel-based methods [6]–[13] usually voxelize 3D point clouds into uniform grids and then apply 3D convolutional operations on the resulting volumetric representations. These methods provide a fair solution for applying convolution on unstructured 3D point clouds. However, voxel-based methods cannot scale well to sparse 3D data because of the cubical computation cost with the resolution and both projection/voxel-based approaches suffer from inevitable structural information loss resulting from the quantization process. In contrast, point-based methods [14]–[19] directly operate on raw point clouds to learn 3D representations. They do not introduce explicit information loss resulting from the voxelization or projection process and become increasingly popular. But using simple multi-layer perceptions (MLP) as the basic operator limits the learning capability of the network.

Transformers [20]–[25] are also adopted recently as point-based methods in 3D point cloud processing after achieving

Fig. 1. (a) Separate local regions that lack whole structure information lead to inaccurate detection boxes (blue boxes). (b) Using overlapping points as intermediaries, our LCP module can pass information between neighboring local regions (the left part) and produce more precise detection results (red boxes).
a great success in solving various 2D vision tasks [26]–[32]. On the one hand, the transformer with its underlying multi-head self-attention mechanism owns the property of permutation invariance which is consistent with the characteristics of point clouds. On the other hand, it is highly expressive with the ability to capture long-range dependencies and effectively learn context-aware representations. They both make the transformer a natural fit for 3D point clouds. However, the absence of local information integration has been described as an inherent deficiency of transformer networks [33]. While several methods have implemented different transformer-based models for 3D point clouds, the local structural information is often disregarded. PCT [20] sends point clouds through an embedding layer and directly applies attention to the entire set of points to extract features. However, this implies the demands on heavy memory and computation overheads. To reduce the complexity, Point Transformer [22] first groups point clouds into several local regions, followed by attention operations in each local region independently. Consequently, each point can receive information from every other point in the same local region, thus capturing local structural patterns. However, it still faces the problem arising from the separate local regions sampling, which leads to the destruction of the instance structure since the inherent relationship between neighboring local regions lacks exploration. Although [21] goes a step further to apply max-pooling on each local region and perform the self-attention operation again on those pooled features (named global transformer layer [21]), it still encounters the same problem because the extra self-attention operation only focuses on modeling the scene-level dependencies.

The aforementioned problems have been studied in 2D scenes, and different solutions were proposed to transmit information amongst local windows [31], [33]. However, unlike 2D images, 3D point clouds are irregular, unordered, and unstructured representations. The interactions between local regions are also irregular in unstructured positions instead of regular grid locations. Therefore, it is much more difficult in 3D scenes to model the interactions between local regions on the fine-grained point level. To solve this problem, we take an in-depth analysis of transformers in 3D point clouds and propose to promote message passing among those neighboring local regions that share the same points. In this way, each local region can obtain more informative and discriminative features to boost the ability of transformers. Based on this thought, we propose an effective yet lightweight module, named Local Context Propagation (LCP).

The LCP module is inserted between two transformer layers such that the overall framework looks like a sandwich structure. Specifically, when following the standard settings in [15], [21] to adopt Farthest Point Sampling (FPS) and k-Nearest Neighbors (kNN) to divide the whole point cloud into different local regions, we find statistically that most local regions tend to overlap with others. Based on this natural structure of the point clouds, as shown in Fig.1(b), those shared points in overlapping areas of the different local regions can serve as the carrier to perform message passing among neighboring local regions exactly. After getting the point feature from the last transformer layer, we collect all local regions that each shared point belongs to, then adopt a lightweight network to generate an adaptive weight acting on those regions. Finally, we update the feature of this point by aggregating its corresponding features from different local regions using the obtained weight. These features updated by the LCP module will then pass to all the other points in a local region with the self-attention mechanism in the next transformer layer. This way allows us to propagate the contexts between neighboring local regions successfully. It is worth noting that the proposed LCP module also differs from the message aggregation that happens between local regions via indirect progressive processing of the larger region with the bigger radius. Our LCP module enables direct message passing between local regions that gives the network a direct learning target and thus enhances the point-level features. In more detail, the main difference between the direct/indirect approach is that our method is applied before max-pooling which is used during every radius change. Thus the message between local regions is interacted before the max-pooling aggregation to minimize information loss when using LCP, which is critical due to the significant global nature of the transformers.

Finally, based on the proposed LCP module, we design a flexible architecture named LCPFormer. Equipped with different numbers of the sandwich structures we mentioned above, which we call LCPFormer Block (shown in Fig.3), the LCPFormer is suitable for different 3D point cloud tasks. In summary, the contributions of this work are three-fold:

• We delve deep into the transformer-based methods in 3D point cloud analysis and propose a simple yet effective module named LCP to promote message passing between neighboring local regions.
• Based on our proposed LCP module, a transformer-based backbone architecture is designed and can be applied for various 3D perception tasks, including object detection, semantic segmentation, and shape classification.
• Our proposed LCPFormer consistently outperforms different transformer-based methods in various 3D tasks including 3D shape classification (93.6% overall accuracy on ModelNet40) and 3D object detection (63.2% mAP@0.25 and 46.2% mAP@0.5 on SUN RGB-D) and semantic segmentation (63.4% mIoU on SatSatUrban; 70.2% mIoU in S3DIS Area 5).

II. RELATED WORK

A. Representation Learning for 3D Point Clouds

Different from 2D images, 3D point clouds with their disorder and sparsity make them extremely difficult to be processed. Previous works mainly include feature extraction by projection, voxelization, directly processing point clouds, and the fusion of those.

Projection-based methods implement different projections to view the 3D point clouds into regular representations on 2D planes. Then apply the 2D convolution network for the features extraction. Methods including projecting points to the perspective view, [1], the bird-eye view [3]–[5], the tangent plane [34], and the use of multi-view fusion [2], [35]–[37]. For
example, [1] hallucinates the 3D representation by taking full advantage of complementary information in the perspective view and the bird-eye view. [2] adopts a multi-view projection strategy to effectively extract features for point cloud quality assessment. Inevitably, lots of points collapsed thus causing the geometric information loss during the projection. In addition, network performance can be severely affected by the fact that the 2D representation differs due to the occlusion in 3D point clouds and different projection methods.

Voxel-based methods [13], [38], [39] is another solution that converts 3D points into voxel grids and applies 3D convolution to generate features. However, the 3D convolution suffers from the cubical computation cost with the resolution when applied directly on the voxel. Although some approaches make good use of the sparsity of the point cloud with sparse convolutions [8], [9], they can not combat the loss of structural information during the quantification.

Point-based methods are proposed in order to prevent the loss of geometric information caused by projection or voxelization. PointNet [14] processes the 3D point clouds directly with point-wise multi-layer perceptions (MLP) and pooling, while PointNet++ [15] further designs a hierarchical structure by dividing the whole point cloud into separate local regions using querying and grouping. Various structures are proposed based on such models [16], [17], [40]–[42]. For example, [43] produces produce more accurate voting centers for object detection by modeling the relationship from neighboring clusters. Alternatively, [44] PSNet proposes a fast data structuring method to tackle the data structuring issue in point-based methods. There are also other methods that combine both point-wise and voxel-wise features in a local field [10], [45] or a global field [46]–[48]. In addition, some of these approaches propose using continuous convolution on the 3D point clouds by the designed kernel structure. For example, PointConv [40] design the convolution kernels as nonlinear functions related to the distance between point coordinates. KPConv [16] further proposes a deformable convolution whose weights are placed in the Euclidean space by the kernel. In order to ingest the in-depth geometric information of the point cloud, some methods [49]–[52] propose creating graphs based on the coordinates and geometry of the point clouds and then adapting the graph convolutions on these point graphs. DGCNN [53] propose an EdgeConv that can be dynamically updated in layers leveraging the properties of the diagram. PointWeb [54] explores the interaction between points by dense connection. Our work focuses on processing point clouds directly via transformers.

B. Transformers in 2D Vision.

Inspired by the great success of transformers in the natural language processing field, people start to consider applying transformers to vision tasks. [28] divides the image into independent patches, and then reshapes each patch into a vector as the input element of the transformer. Since then, transformer structure has been successfully applied to computer vision, and many improvements [26], [27], [29], [30], [32], [33], [55]–[60] have been proposed. Some methods [31], [33], [61], [62] further group images with non-overlapping windows and apply transformers within each window separately, but lack feature aggregation between each window. To deal with this problem, [33] proposes shifted windows to dynamically adjust the window position and can be used for multiple tasks including semantic segmentation, object detection, and image classification. [31] combines transformers and the HRNet [63], borrowing the natural advantage of a 3 × 3 depth-wise convolution to improve transformers. [31] is also compatible with dense prediction tasks such as human pose estimation.

C. Transformers in 3D Point Clouds.

As a sequence-to-sequence structure, transformers completely ignore the input order and treat the position information as a set of sequences [64], [65], revealing the possibility of the application of transformers in the 3D field. Currently, many transformer-based methods for 3D point clouds have been proposed to improve the performance of PointNet++ [15]. Point Transformer [22] proposes a U-Net [66] structure with self-attention layers constructed from vector attention [65]. This model can be used in both shape classification and object part segmentation tasks in 3D point clouds. Similarly, [23] also proposes a similar idea that uses the attention mechanism and designs a SortNet to refine local features from different sub-spaces. PCT [20] proposes to use an offset-attention with implicit Laplace operators and normalization refinement for transformer layers.

Pu-Transformer [67] is introduced for point cloud upsampling that uses transformer structure to enhance point-wise and channel-wise relations of the point feature.

For the detection task, Pointformer [21] designs an attention-based feature extractor to serve as the backbone for different object detection frameworks. [24] adopts PointNet++ as the backbone and proposes a Group-Free detector with multi-head self-attention.

In this paper, we take an in-depth analysis of the drawbacks of transformer-based approaches in 3D point clouds. We find that separate local regions damage the structure of instances, which restricts the ability of transformers to enhance the feature representations. Thus, we design a simple yet effective method to exploit the inherent relationship between local regions.

III. METHOD

In this section, we first briefly revisit the general formulation of the transformer and self-attention. Then we conduct an in-depth analysis of transformers in 3D point clouds and propose a lightweight yet effective module, named Local Context Propagation Module (LCP). Lastly, based on this module, we present our LCPFormer architecture for various 3D tasks including shape classification, object detection, and semantic segmentation.

A. Preliminary

We start by reviewing the commonly used multi-head self-attention (MHSA) in transformers. Given a set of input features \( F = \{ f_i \} \) and corresponding positions \( X = \{ x_i \} \), the
features from last layer

updated features

In the Local Context Propagation (LCP) module, given a point shared by several local regions, an adaptive weight is assigned to each region according to the computed attention weights. This weight is then used to update the feature of the point by aggregating its corresponding features from different local regions. The mean and max operations are performed on the aggregated features to generate the updated feature. This process ensures that context information is propagated between neighboring local regions effectively.

The main difference between our method and other methods lies in that the interaction occurs at the point level, where each point feature is updated instead of treating each local region independently. Moreover, our method enhances the feature learning from a more fine-grained perspective, which makes it more effective for transformers to achieve discriminative representations.

As discussed in Sec.III-B, when directly applying the transformer to each local region independently, it lacks sufficient information exchange among regions. There are several attempts to address this issue in the image field. HRFormer [31] uses a $3 \times 3$ depth-wise convolution to merge features from separate local windows and Swin Transformer [33] utilizes a shifted window mechanism to implicitly achieve a similar goal. However, these methods make their features inferior for transformers to achieve discriminative representations.

Conclusion

Our method is based on a simple observation that when dividing the whole point cloud into different local regions, naturally there is overlap among them. We propose a new module named Local Context Propagation (LCP), which can propagate context information between neighboring local regions on the fine-grained level. Extensive experiments in Sec.IV can well prove the superiority of this fine-grained design in transformers.
in a 3D point cloud, we select \( N' = N/2 \) center points via FPS algorithm and group neighbor points via kNN algorithm \((k = 16 \) in our experiments), thus we obtain \( k \times N' = 8N \) points in total. According to our statistics, most of the \( N \) points are grouped multiple times repeatedly. Consequently, those shared points can serve as suitable carriers to conduct message passing between local regions to make feature representations more informative and discriminative. Fig.1 (b) can well illustrate our motivation.

Briefly speaking, LCP works by updating point features in overlapping areas of different regions. As discussed above, most of the points are shared by several local regions, thus the whole point cloud features can be effectively updated. Given a point \( x_i \), we denote the local regions it belongs to as \( \{G_1, \ldots, G_m\} \). After the transformer is independently applied to those regions, for each local region \( G_j \), the point \( x_i \) should have a corresponding feature in it, and we denote it as \( f_i^j \). We update the feature of \( x_i \) by using the weighted sum of those corresponding features:

\[
f_i^j = \sum_j w_j f_i^j \quad j = 0, \ldots, m
\]

where \( w_j \) denotes the normalized weight for region \( G_j \). In this way, we successfully achieve the purpose of context information propagation between local regions.

The detailed structure of LCP can be found in Fig.2. Assuming that the whole point cloud contains \( N \) points grouped into \( M \) local regions, we can denote the input of LCP as \( F_{in} \in \mathbb{R}^{M \times K \times C} \), where \( K \) is the number of points in each local region and \( C \) is the feature dimension of each point. To extract an effective representation for each local region, we choose to concatenate the outputs from two pooling operators: max-pooling and average-pooling on the learned point-wise features following [69]. The motivation behind this design is that max-pooling tends to extract the specific important features while average-pooling can involve the surrounding context. Both are important for effective point cloud analysis. Representations \( A \in \mathbb{R}^{M \times 2C} \) for \( M \) local regions are obtained and sent through a \( 1 \times 1 \) convolution to generate an adaptive weight matrix \( W \in \mathbb{R}^{M \times C} \), which is normalized using Softmax. This process can be formulated as:

\[
A = \text{MaxPool}(F_{in}) \oplus \text{AvgPool}(F_{in}) \quad (8)
\]

\[
W = \text{Softmax}(\text{Conv}(A)) \quad (9)
\]

The weight matrix \( W \) is applied in Eq.7 to update the feature of each point and we thus obtain the final output \( F_{out} \in \mathbb{R}^{N \times C} \).

The parameter number and computational complexity of the proposed LCP module are negligible as shown in the ablation study of module efficiency in Sec.IV-E

\[\text{D. LCPFormer}\]

Based on the LCP module, we propose our LCPFormer to effectively extract features from point clouds. As shown in Fig.3, the LCPFormer is composed of multiple transformer blocks (colored in blue) and upsampling layers (colored in green). We adopt the upsampling layer in PointNet++ [15] and skip connections are used to enhance the upsampled features.

---

**Fig. 3.** Overall architecture of our transformer feature extractor. The LCPFormer Block consists of a grouping layer, two self-attention layers (MHSA), a local context propagation (LCP) module, and a max-pooling layer. The backbone network is constructed by multiple LCPFormer blocks and upsampling layers. Our LCPFormer can be applied to different tasks including object detection, classification, and segmentation, according to the use of different heads.
Each LCPFormer Block comprises a grouping layer, two self-attention layers, and an LCP module in the middle.

The grouping layer groups the whole point cloud \( \{f_i, x_i\} \) into different local regions \( \{G_1, G_2, \ldots, G_M\} \), where \( M \) is the number of regions and \( f_i, x_i \) are of size \( C \) and \( d \) respectively. It first samples center points via FPS and for each center \( j \), a group \( G_j = \{f_{ij}, x_{ij}\} \) of size \( K \times (d + C) \) is constructed by gathering \( K \) points within a local neighborhood using kNN or BallQuery [15].

The following self-attention layer performs self-attention within each local region independently to capture local features. Then, our LCP module is introduced to enhance feature interaction between neighboring local regions. Another self-attention layer is adopted for further feature optimization based on the LCP module outputs. Lastly, a common max-pooling operation is used to abstract a single \( C' \)-dimension representation for each local region, and the final output is in the shape of \( M \times C' \).

**Network Architecture**. It is worth noting that the number of local regions, the number of self-attention layers in each block, and the number of LCPFormer Blocks may vary in different tasks. It is reasonable as different tasks share different characteristics. We construct our LCPFormer architecture utilizing the proposed LCP module and applying it to various 3D point cloud applications. For each task, we choose different hyper-parameters and slightly modify the basic architecture for better adaptation.

1) **3D Object Detection**: The backbone network for 3D object detection contains 4 LCPFormer blocks and 2 upsampling layers. The whole input point cloud is downsampled to 2048, 1024, 512, and 256 points with an increasing receptive radius of 0.2, 0.4, 0.8, and 1.2 respectively by the LCPFormer blocks, and then upsampled to 1024 points by the next two upsampling layers. In the grouping layer of each LCPFormer Block, \( k = 16 \) points are grouped for each local region to achieve a balance between computational complexity and performance. Without loss of generality, we adopt the LCPFormer to replace the PointNet++ backbone in the original VoteNet [18] model for a fair comparison.

2) **3D Classification**: The backbone network for shape classification is composed of 4 consecutive LCPFormer Blocks, each with one self-attention layer and one LCP module. The number of points is downsampled from 1024 and kept to 256 in all 4 blocks thus all points are selected during FPS operation. The \( k \) of kNN is set to 16, 12, 8, and 8 in four blocks for multi-scale feature generation.

For better comparison, we build our backbone with similar capacity and adopt the same classification head as PCT [20]. Comparison results can be found in the ablation study of module efficiency in Sec.IV-E.

3) **3D Semantic Segmentation**: We build a UNet-like network [66] using 4 LCPFormer blocks and 4 upsampling layers in the backbone for the semantic segmentation task since it requires point-wise features for dense prediction. A simple softmax segmentation head is adopted to output class probabilities from the extracted features. The input point number is set to 4096 and further downsampled to 1024, 512, and 256 points, respectively. We then upsample it gradually to 4096 points with four upsampling layers. The \( k \) of kNN is set as 16.

### IV. EXPERIMENTS

We apply our LCPFormer to various 3D tasks including 3D shape classification, 3D object detection, and semantic segmentation on the most widely-used benchmarks. Please see Appendix for further benchmarks details.

#### A. 3D Object Detection

**SUN RGB-D** [70]. We evaluate our method for 3D object detection on the commonly used SUN RGB-D dataset [70]. SUN RGB-D [70] is an indoor point cloud dataset for multiple scene understanding tasks. It contains \( \sim10K \) RGB-D images densely annotated with over 58K 3D bounding boxes in 37 different categories.

Our training settings mostly follow [24]. We employ an AdamW optimizer with an initial learning rate of 0.0003 and a weight decay of 0.0005. We train the network for 600 epochs.

| Methods                  | Reference | Backbone       | mAP@IoU0.25 | mAP@IoU0.5 |
|--------------------------|-----------|----------------|-------------|------------|
| H3DNet [71]              | ECCV 2020 | 4 x PointNet++  | 60.1        | 39.0       |
| MLCVNet [72]             | CVPR 2020 | PointNet++     | 59.8        | -          |
| HGNNet [73]              | CVPR 2020 | GU-Net         | 61.6        | -          |
| 3DETR [74]               | ICCV 2021 |                | 58.0        | 30.3       |
| 3DETR-m [74]             | ICCV 2021 | PointNet++     | 59.1        | 32.7       |
| T3D [43]                 | TCSVT 2021| PointNet++     | 60.1        | -          |
| EQ-Net [75]              | CVPR 2022 | EQ-PointNet++  | 60.5        | 38.5       |
| VoteNet [18]             | ICCV 2019 | PointNet++     | 59.1        | 35.8       |
| Pointformer + VoteNet [21]| CVPR 2021 | Pointformer    | 61.1        | 36.9       |
| LCPFormer + VoteNet      | N/A       | w/ LCP         | 61.4        | 39.6       |
| Group-Free [24]          | ICCV 2021 | PointNet++     | 62.8        | 44.4       |
| LCPFormer                | N/A       | w/ LCP         | 62.7        | 43.3       |
|                          |           |                | **63.2**    | **46.2**   |
TABLE II

PERFORMANCE COMPARISONS WITH PREVIOUS METHODS ON MODELNET40 DATASET. Note that we do not use normal vectors as the extra inputs or other special tricks as well. * MEANS WE REPORT THE BEST RESULTS AMONG PUBLIC CODEBASES AND OUR OWN REPRODUCTION. † WE BUILD OUR LCPFFormer ON THE BASIS OF THE PCT WITH THE EXTRA LCP MODULE.

| Methods          | Reference    | mAcc(%) | OA(%) |
|------------------|--------------|---------|-------|
| 3DShapeNets [76] | CVPR 2015    | 77.3    | 84.7  |
| VoxSet [38]      | IROS 2015    | 83.0    | 85.9  |
| Subvolume [36]   | CVPR 2016    | 86.0    | 89.2  |
| PointNet [14]    | CVPR 2017    | 86.2    | 89.2  |
| A-SCN [77]       | CVPR 2018    | -       | 89.8  |
| MVCNN [35]       | ICCV 2015    | -       | 90.1  |
| SO-Net [78]      | CVPR 2018    | -       | 90.9  |
| Point Transformer | ICCV 2021    | -       | 91.7  |
| Kd-Net [79]      | ICCV 2017    | -       | 91.8  |
| PointNet++ [15]  | NeurIPS 2017 | -       | 91.9  |
| PointGrid [48]   | CVPR 2018    | -       | 92.0  |
| SpecGCN [80]     | ECCV 2018    | -       | 92.1  |
| Point-PlaneNet [81] | DSP 2020 | 90.5 | 92.1 |
| PCNN [82]        | TOG 2018     | -       | 92.3  |
| PointWeb [54]    | CVPR 2019    | 89.4    | 92.3  |
| SpiderCNN [83]   | ECCV 2018    | -       | 92.4  |
| PointCNN [84]    | NeurIPS 2018 | 88.1    | 92.5  |
| PointConv [40]   | CVPR 2019    | -       | 92.5  |
| A-CNN [85]       | CVPR 2019    | -       | 92.6  |
| P2Sequence [86]  | AAAI 2019    | -       | 92.6  |
| Point Transformer | Access 2021  | -       | 92.8  |
| KPConv [16]      | ICCV 2019    | -       | 92.9  |
| DGCNN [53]       | TOG 2019     | 90.2    | 92.9  |
| RS-CNN [87]      | CVPR 2019    | -       | 92.9  |
| PointANSL[88]    | CVPR 2020    | -       | 92.9  |
| InterpCNN [89]   | ICCV 2019    | -       | 93.0  |
| DRNet [90]       | WACV 2021    | -       | 93.1  |
| EQ-Net [75]      | CVPR 2022    | -       | 93.2  |
| PSE [44]         | T-CSVT 2022  | -       | 93.3  |
| RSMix [91]       | CVPR 2021    | -       | 93.5  |
| PatchFormer [92] | CVPR 2022    | -       | 93.5  |
| PCT (w/o LCP)†   | CVM 2021     | 90.0    | 93.2  |

| LCPFFormer       | -            | 90.7    | 93.6  |

B. Point Cloud Shape Classification

ModelNet40 [76]. We benchmark our 3D shape classification network on the widely used point cloud shape classification dataset ModelNet40 [76]. It contains over 10,000 CAD models in 40 categories.

We follow the basic data split as well as the data augmentation in [20]. During training, a random anisotropic scaling between [0.67, 1.5] and a random translation between [−0.2, 0.2] are used successively. We set the input point number to 1024, and train the network for 250 epochs with the SGD optimizer. The initial learning rate is set to 0.0008 and decays after every epoch using the CosineAnnealing strategy. Following most of the previous works, we report two common metrics i.e. OA (Overall Accuracy) and mAcc (mean of class-wise accuracy) for evaluation.

Results on ModelNet40 are detailed in Tab.II. LCPFFormer outperforms the PCT with 93.6% in OA and 90.7% in mAcc. Note that we do not use normal vectors as extra inputs in our experiment.

C. 3D Semantic Segmentation

SensatUrban [94] & S3DIS [93]. We then evaluate our method for 3D semantic segmentation on both indoor and outdoor scenes with two large-scale point cloud datasets. SensatUrban [94] is a photogrammetric point cloud dataset that contains over 100 million richly annotated points. It is labeled into 13 categories, including large objects e.g. buildings and ground, and extremely small objects in an urban scene e.g. bikes and paths. S3DIS [93] is an indoor 3D RGB point clouds dataset that contains six areas of three buildings. All points are labeled with their semantic ground truth from 13 categories including board, bookcase, chair, ceiling, beam, etc.

We train the network for 100 epochs with the AdamW optimizer. The initial learning rate is set to 0.001 and decays after every epoch using the CosineAnnealing strategy. We employ the mean Intersection over Union (mIoU) as the basic evaluation metric and also report the Overall Accuracy (OA) and per-class IoU scores respectively.

Results for S3DIS are shown in Tab.III. Following a common protocol [15], we evaluate the presented approach with Area 5 withheld during training and used for testing. Our method achieved 70.2% mIoU, 76.5 mAcc% and 90.1% OA. We also test our method on a challenging urban scale segmentation dataset SensatUrban [94] and achieved promising results with about 5.8% mIoU overall improvements compared with the previous best method KPConv [16]. Full results for SensatUrban are presented in Tab.IV, our proposed method also has great potential when it comes to fine-grained categories. For instance, LCPFFormer reaches 57.0% mIoU on bridge category and 46.3% mIoU on rail category while the scores of PointNet [14] remain 0.0% mIoU on both. Similar results are also reflected in the traffic category and path category, where our method makes a 29.8% mIoU and 49.2% mIoU improvement respectively compared with PointNet [14].

Our proposed LCPFFormer with the LCP module remedies the information loss of instances caused by separate local regions in common transformer-based methods [21], [22] via and decrease the learning rate by $10 \times$ at epochs 420, 480, and 540. To augment the training data, we adopt random flips in the horizontal direction, random rotations between $[-5^\circ, 5^\circ]$, and random scaling by $[0.9, 1.1]$. Following a standard evaluation protocol, we adopt the mean Average Precision (mAP) under IoU thresholds 0.25 and 0.5 as evaluation metrics, and results of the 10 most common categories are reported.

Results on SUN RGB-D are shown in Tab.I. When adopting our LCPFFormer as the backbone of VoteNet [18], results show that our method makes a 2.3% mAP@0.25 improvement and further outperform the PointFormer [21] by 0.3% mAP@0.25. Moreover, when adopting our LCPFFormer as the backbone of Group-Free [24], our LCPFFormer achieves 63.2% mAP@0.25 and 46.2% mAP@0.5, outperforming the Group-Free by 1.8% mAP@0.5 and 0.4% mAP@0.25. The comparison between the last two rows validates the effectiveness of our LCP module and is consistent with our analysis that context propagation between local regions is helpful for accurate detection results in Fig.1.
local context propagation among neighboring regions. In this way, it can achieve more informative and discriminative representations to recognize the whole structure of an instance very well, thereby reducing classification errors and contributing to its excellent performance on semantic segmentation tasks.

D. Visualization Analysis

To evaluate the impact of our proposed LCPFormer more intuitively, we randomly sample several point cloud scenes to visualize their output results. We choose both SensatUrban and SUN RGB-D datasets for a broader comparison. The visualization results are shown in Fig.4 and Fig.5.

Results shown in Fig.4 compare our detection results with the ground-truth label. Similar outcomes can also be found in Fig.5 that strongly support our motivation, which is the separated sampling in the local region causes the destruction of the instance information and our proposed LCP can effectively improve the point cloud feature with a more informative and discriminative representation. As we can see from the second and fourth rows, some instances were incorrectly segmented. Conversely, the LCPFormer successfully maintained the integrity of the instances and classify most of the points correctly.

E. Ablation Study

In this section, we conduct extensive ablation experiments to verify our method, including the effectiveness of the proposed LCP module, and the impact of different design choices on the transformer backbone. All experiments are trained and evaluated on the SUN RGB-D [70] dataset.

Effects of each component. We validate the effectiveness of the proposed LCPFormer and the LCP module on the detection task. In addition to presenting the difference between our point level LCP and the natural message aggregation from the receptive field changes during downsampling, we also implement a global transformer layer to replace our LCP module, and the impact of different design choices on the transformer backbone. All experiments are trained and evaluated on the SUN RGB-D [70] dataset.

TABLE III

| Method            | mIoU(%)  | mAcc(%)  | OA(%) | mIoU(%)  | mAcc(%)  | OA(%) |
|-------------------|----------|----------|-------|----------|----------|-------|
| PointNet [14]     | 41.1     | 23.7     | -     | 23.7     | 80.8     | -     |
| PointNet++ [15]   | -        | -        | -     | -        | 32.9     | 84.3  |
| TragenConv [34]   | 52.8     | 62.2     | 82.5  | 33.3     | 77.0     | -     |
| SGPH4 [49]        | 58.0     | 66.5     | 86.4  | 37.3     | 85.3     | -     |
| SparseConv [8]    | -        | -        | -     | 42.7     | 88.7     | -     |
| LCTransformer [95] | 64.1     | 71.9     | 87.6  | -        | -        | -     |
| KPCconv [16]      | 67.1     | 72.8     | 87.2  | 57.6     | 93.2     | -     |
| RandLA-Net [17]   | 62.4     | 71.4     | 87.2  | 52.7     | 89.8     | -     |
| PatchFormer [92]  | 68.1     | -        | -     | -        | -        | -     |
| PSNet [44]        | 62.9     | -        | 87.8  | -        | -        | -     |
| DenseKPN [96]     | 68.9     | -        | 90.8  | -        | -        | -     |
| Point Transformer* [22] | 70.0     | 76.8     | 90.4  | -        | -        | -     |
| LCPFormer w/o LCP| 69.3     | 75.2     | 90.2  | 61.7     | 93.0     | -     |
| LCPFormer w/ LCP | 70.2     | 76.8     | 90.8  | 63.4     | 93.5     | -     |

TABLE IV

| Methods            | OA(%)  | mIoU(%)  | ground | sky | buildings | walls | bridge | parking | rail | road | signs | fence | pole | traffic | car | truck | bus | train | motorcycle | bus | taxi | motorbike | water |
|-------------------|--------|----------|--------|-----|-----------|-------|--------|---------|------|------|-------|-------|------|----------|-----|-------|-----|-------|-------------|-----|------|-----------|-------|
| PointNet [14]     | 80.8   | 23.7     | 68.0   | 89.5 | 80.0      | 0.0   | 0.0    | 4.0     | 0.0  | 31.6 | 0.0   | 0.0   | 35.1 | 0.0       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| PointNet++ [15]   | 84.3   | 32.9     | 72.5   | 94.2 | 84.8      | 2.7   | 2.1    | 25.8    | 0.0  | 31.5 | 11.4  | 38.8  | 7.1  | 56.9       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| TragenConv [34]   | 77.0   | 33.3     | 71.5   | 91.4 | 75.9      | 35.2  | 0.0    | 45.3    | 0.0  | 26.7 | 19.2  | 67.6  | 0.0  | 54.8       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| SparseConv [8]    | 85.3   | 37.3     | 69.9   | 94.6 | 88.9      | 32.8  | 12.6   | 15.8    | 0.0  | 30.1 | 34.0  | 74.4  | 0.0  | 43.2       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| KPCconv [16]      | 83.7   | 42.7     | 74.1   | 97.9 | 94.2      | 63.3  | 7.5    | 24.2    | 0.0  | 30.1 | 34.0  | 74.4  | 0.0  | 43.2       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| RandLA-Net [17]   | 83.7   | 42.7     | 74.1   | 97.9 | 94.2      | 63.3  | 7.5    | 24.2    | 0.0  | 30.1 | 34.0  | 74.4  | 0.0  | 43.2       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |
| LCPFormer         | 93.5   | 63.4     | 86.5   | 98.3 | 96.0      | 55.8  | 57.0   | 50.6    | 46.3 | 61.4 | 51.5  | 85.2  | 49.2 | 86.2       | 0.0 | 0.0   | 0.0 | 0.0   | 0.0         | 0.0 | 0.0  | 0.0       | 0.0   |

TABLE V

| Backbone          | LCP | GTL | mAP@IoU0.25 | mAP@IoU0.5 |
|-------------------|-----|-----|-------------|------------|
| PointNet++        | X   | X   | 62.8        | 42.3       |
| LCPFormer         | X   | X   | 62.7        | 43.3       |
| LCPFormer w/o LCP|     |     | 63.2        | 46.2       |
| LCPFormer w/ LCP |     |     | 61.2        | 43.9       |
Fig. 4. Visualization of detection results on SUN RGB-D dataset. Left: Ground truth. Right: Our LCPFormer results.

The first row denotes the PointNet++ baseline. We can find that if we discard the LCP module from our LCPFormer, the simple transformer-based method can achieve comparable performance with the baseline. However, if our proposed LCP module is adopted, it can accomplish a significant performance improvement, especially in the mAP@0.5 metric, which has about 3.9% mAP gains. This result keeps consistent with our discussion in Sec.III-C and Sec.IV-A. The LCP module helps transformers to achieve more accurate bounding boxes. In contrast, as shown in the last row in Tab.V, the implemented global transformer layer (GTL) actually damages the performance to some extent as it omits the inherent relationship between neighboring local regions.

Ablation of the number of neighbor points. Since our method highly relies on overlapping points and the number of neighbor points $k$ naturally determines the degree of overlap to a large extent. Tab VI shows the influence of different neighbor point number $k$. Within a certain range ($k \leq 16$), increasing more points to enhance the structural information in the local area brings more gains for the model performance. But when these neighboring regions go too far ($k \geq 32$), it brings excessive noise with more irrelevant points, which weakens the accuracy of the network to a certain extent. Similar results can also be found in [22]. We adopt $k = 16$ as the default setting in our experiments for a balance between performance and computational cost.

| $k$  | mAP@IoU0.25 | mAP@IoU0.5 |
|------|-------------|-------------|
| 4    | 56.92       | 37.53       |
| 8    | 59.48       | 41.37       |
| 16   | **62.70**   | **43.34**   |
| 32   | 61.52       | 44.22       |

Ablation of the transformer backbone architecture. To deeper validate our transformer architecture, we conduct an ablation study on the number of LCPFormer blocks and the number of MHSA layers in each block. The results are summarized in Tab.VII and Tab.VIII. It is obvious that insufficient parameters will inevitably weaken the capacity of the network, but deeper models (block number = 5 or attention layer = 3) bring more parameters and unnecessary learning burdens, which makes the model extremely difficult to train.

| Blocks   | mAP@IoU0.25 | mAP@IoU0.5 |
|----------|-------------|-------------|
| 3        | 61.36       | 42.80       |
| 4        | **62.70**   | **43.34**   |
| 5        | 61.12       | 42.17       |

| Attention Layers | mAP@IoU0.25 | mAP@IoU0.5 |
|------------------|-------------|-------------|
| 1                | 61.16       | 43.66       |
| 2                | **62.70**   | **43.34**   |
| 3                | 62.08       | 43.81       |

Module efficiency analysis. Eq.7-9 summarize our proposed LCP module. To further validate the efficiency of our LCP module, we compare our method with two famous transformer-based methods for the 3D shape classification task and the results are shown in Tab.IX. Common metrics including GFLOPs, model parameters, and latency are used. All the experiments are conducted on a single NVIDIA GeForce RTX 2080Ti.

| Method     | GFLOPs | Params | Latency | OA  |
|------------|--------|--------|---------|-----|
| Point Transformer | 147.2  | 9.58M  | 18.7ms  | 91.7 |
| PCT        | 17.6   | 2.94M  | 14.6ms  | 93.2 |
| LCPFormer  | 17.6   | 3.04M  | 14.9ms  | 93.6 |

Note that our LCPFormer for the 3D shape classification task is designed on the basis of the PCT [20] structure as described in Sec.III-D. The only difference between them is...
Fig. 5. Visualization of segmentation results on SensatUrban dataset. We randomly sample several point cloud scenes to visualize their output results. A current state-of-the-art method is selected for a more intuitive comparison. The different results of our LCPFormer and RandLA-Net are highlighted in white boxes.

the extra LCP module in the LCPFormer. The comparison between PCT [20] and LCPFormer shows that our LCP module is lightweight enough yet effective. It achieves a considerable improvement of 0.4% OA on the ModelNet40 dataset with a negligible amount of parameters and inference time.

V. Conclusion

This work explores the natural fit of the transformer in 3D point cloud perception and focuses on the destruction of the instance information caused by separate local regions. We present a novel and effective message exchange module named Local Context Propagation (LCP). Unlike the previous methods, our LCPFormer is tailored for irregular point clouds and enhances the inherent relationship among the neighboring local regions via local context propagation. Finally, our proposed method achieves considerable improvement compared with various transformer-based methods in multiple 3D tasks including shape classification, and dense prediction tasks such as object detection and semantic segmentation. As a sequence-to-sequence structure, transformers show great potential for sets embedded in the geometric space like point clouds. In future work, we would like to further explore the versatility of our work and implement it to even more datasets for different 3D tasks such as 3D pose estimation, 3D point cloud matching, etc.

REFERENCES

[1] J. Deng, W. Zhou, Y. Zhang, and H. Li, “From multi-view to hollow-3d: Hallucinated hollow-3d r-cnn for 3d object detection,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 12, pp. 4722–4734, 2021.

[2] Q. Liu, H. Yuan, H. Su, H. Liu, Y. Wang, H. Yang, and J. Hou, “Pqa-net: Deep no reference point cloud quality assessment via multi-view projection,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 12, pp. 4645–4660, 2021.

[3] B. Yang, W. Luo, and R. Urtasun, “Pixor: Real-time 3d object detection from point clouds,” in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 7652–7660.

[4] M. Liang, B. Yang, S. Wang, and R. Urtasun, “Deep continuous fusion for multi-sensor 3d object detection,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 641–656.

[5] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, “Joint 3d proposal generation and object detection from view aggregation,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 1–8.
[96] Y. Li, X. Li, Z. Zhang, F. Shuang, Q. Lin, and J. Jiang, “Densekpnet: Dense kernel point convolutional neural networks for point cloud semantic segmentation,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2022.

Zhuoxu Huang received a B.E. degree in Geodesy and Geomatics Engineering from Wuhan University, Wuhan, China, in 2021. He is currently pursuing a Ph.D. degree with the Department of Computer Science, Aberystwyth University, Aberystwyth, U.K. His research interests include video understanding, 3D vision, and artificial intelligence.

Zhiyou Zhao received a B.E. degree in Computer Science and Technology from Shanghai Jiao Tong University, Shanghai, China, in 2022. He is currently pursuing a M.S. degree in Data Science in the Department of Data Science, Chinese University of Hong Kong, Shenzhen. His research interests include 3D point cloud perception, object detection and artificial intelligence.

Banghuai Li received a B.E. degree in the School of Software from Wuhan University, Wuhan, China, in 2015 and an M.S. degree in the School of Electronics Engineering and Computer Science from Peking University, Beijing, China, in 2018. He is currently a Staff RD Engineer in Momenta, a leading autonomous driving technology company in China. Before that, he was a Researcher at Megvii Company in China, where he finished this work. His research interests include 2D/3D vision, autonomous driving, and artificial intelligence.

Jungong Han is Chair Professor in Computer Vision at the Department of Computer Science, University of Sheffield, U.K. He holds the Honorary Professorship with Aberystwyth University and the University of Warwick, U.K. His research interests include computer vision, artificial intelligence, and machine learning.