2023 IEEE International Conference on Image Processing (ICIP) | 978-1-7281-9835-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICIP49359.2023.10222036

LAPTRAN: TRANSFORMER EMBEDDING GRAPH LAPLACIAN FOR POINT CLOUD PART SEGMENTATION

Abiao Li[†] *Chenlei Lv*[‡] *Yuming Fang*[⋆†] *Yifan Zuo* [†]

[†] Jiangxi University of Finance and Economics, China
[‡]Nanyang Technological University, Singapore

ABSTRACT

Since the feature representations of the points located at the junction regions of various parts are ambiguous, it is still challenging to exploit the fine-grained semantic features of point clouds on part segmentation tasks. To resolve the issue, we design a modified transformer module, named Laplacian transformer, to investigate the local differences between each point and its corresponding neighbors based on graph Laplacian theory. This module constructs a more accurate local geometric representation of the point cloud. It concentrates on the points located at the junction areas of various parts while boosting the recognition effect of these points. Encapsulated with the Laplacian module, we propose a Unet-like transformer framework to perform part segmentation for point clouds. Experimental results demonstrate that the proposed framework achieves more accurate results on public benchmark datasets.

Index Terms— Point Cloud, Fine-grained Feature, Laplacian Transformer, Part Segmentation

1. INTRODUCTION

Deep learning has witnessed a technological revolution in natural language processing and computer vision tasks, designing a deep learning framework for point clouds is much desired for its wide range of applications, such as autonomous driving and 3D modeling. As the pioneer work for point cloud processing based on deep learning, PointNet [1] is proposed to extract global features via symmetric functions for point classification and segmentation tasks. The drawback of PointNet is that it extracts global features from each point in isolation, neglecting local neighborhood interactions that are vital for simplification [2] and shape reconstruction [3].

Based on PointNet, many improvement schemes [4, 5, 6] are designed to investigate local geometric features besides processing individual points with the aim of extracting representative global features. However, the neighborhood search method employed by these algorithms suffers from the inherent problem that it is performed in the coordinate space instead of the feature space, which imposes a constraint on finding points with feature similarity in a higher-dimensional space. Therefore, dynamic local neighborhood graph is considered to explore the local geometric information of point clouds, which performs convolution-like operations on the edges of linked nearby points. The representative methods include DGCNN [7] and LDGCNN [8]. However, the information between the deep features of the point cloud and the neighbors searched may be too similar to provide efficient edge vectors, resulting in limitations in modeling the global features of the point cloud.

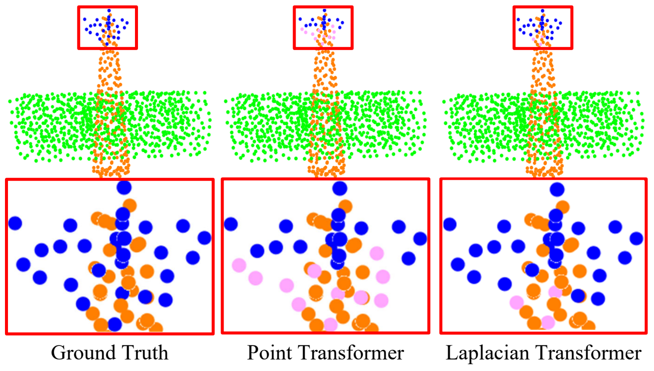

Fig. 1. Comparisons of semantic-based part segmentation. The points exhibiting incorrect segmentation results are indicated by pink markers enclosed within red boxes. Our method achieves more accurate results in the junction areas.

Currently, a efficient model is the attention mechanism-based transformer [9], which possess powerful capability in learning global features for large-scale data. By incorporating the self-attention mechanism into the point cloud network model, Point Transformer [10] achieves competitive results in point cloud processing tasks. However, unreliable results may still be obtained when transformer models are employed to perform point cloud segmentation tasks. As shown in Fig.1, there is a high probability that points located in the junction region are incorrectly segmented, which are produced by ambiguous topological relations and overlapping points.

⋆Corresponding Author-Yuming Fang(leo.fangyuming@foxmail.com). This work was supported in part by the Natural Science Foundation of China under Grants 62132006 and 62271237, and in part by the Natural Science Foundation of Jiangxi under Grant 20223AEI91002.

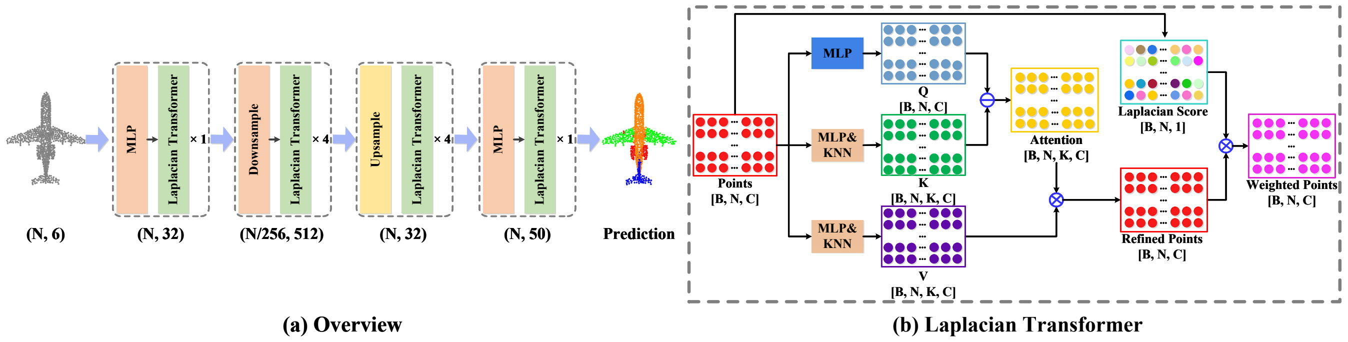

Fig. 2. (a) Overview of the proposed framework. (b) Laplacian Transformer module. For simplicity, position encoding δ is neglected in Fig.2 (b). MLP: Multilayer Perceptron; KNN: K-Nearest Neighbor algorithm; N: number of points; B: Batch size; C: number of channels of each point; K: number of neighborhood points searched.

To address this issue, we develop a Laplacian transformer module to generate a Laplacian score for each point cloud based on graph Laplacian theory. Combining the attention mechanism, the Laplacian transformer module is encapsulated in a UNet-like transformer framework to reveal point-based attention with Laplacian score. In general, regions with a higher Laplacian score are clustered along the edge contours of the object component, while intermediate flat regions receive a lower Laplacian score. Therefore, the framework is instructed to concentrate on the junction regions of various parts through the Laplacian score of each region to achieve the improved segmentation effect for these points. The contributions are summarized as:

- We design a Laplacian transformer module based on graph Laplacian theory. The module concentrates on the edges of different parts to improve the performance for contour point learning.
- Based on the Laplacian transformer module, we propose a UNet-like transformer framework for point clouds on part segmentation tasks. It improves the accuracy of local geometric features in segmentation tasks.
- We validate the effectiveness of the proposed method on benchmark datasets and perform a series of ablation studies for better performance.

2. PROPOSED METHOD

2.1. Overview

The overall architecture of the proposed model is shown in Fig.2 (a). A hierarchical network is utilized to extract multi-scale features from the input point cloud, which is similar to U-Net [11]. The input point cloud has N points with three coordinates and RGB values. In the network, an encoder-decoder structure is used, where the encoder and decoder are composed of five stages. Within the first encoder stage, MLP and Laplacian Transformer layers are sequentially performed to aggregate local structural information for the input point cloud. Then, to extract semantic information from local neighborhoods in a high-dimensional space, the features are progressively downsampled with channel expanding in the following four stages, which consists of a downsample layer and a Laplacian transformer layer. The similar structure is designed in decoder stage. The only difference is that the features are progressively upsampled by four layers. The cross-entropy function is employed to calculate the gap between the prediction and the ground truth element-wise. The details of the Laplacian transformer module are given in the following parts.

2.2. Laplacian Transformer Module

The transformer layer in Point Transformer [10] is designed based on the vector self-attention mechanism. Firstly, given point cloud features, three MLP operations φ, ψ and α are performed to obtain query Q, key K and value V vectors, respectively, which maps the point cloud to a high-dimensional space. Then, KNN is used to search neighbors set $\boldsymbol{\chi(i)}$ for point clouds $x = \{\boldsymbol{x_i} | \boldsymbol{i} = 1, 2, \dots N\}$. With the addition operation between the position encoding δ and the result $Q - K$, the attention between each point and its corresponding neighborhood is computed to represent the similarity. The refined point features are obtained by multiplying the attention with the result $V + \delta$. The computational formulation can be expressed as

$$y_i = \sum_{\boldsymbol{x_j} \in \boldsymbol{\chi(i)}} \rho(\varphi(\boldsymbol{x_i}) - \psi(\boldsymbol{x_j}) + \delta) \odot (\alpha(\boldsymbol{x_j}) + \delta) \quad (1)$$

A drawback of Eq.(1) is that it assigns equal weight to all points, which ignores the fact that the importance attached to points located in the junction region is more essential than points located in the flat region. For this reason, the Laplacian transformer module is designed, as shown in details in Fig.2

(b), which generates a Laplacian score for each point cloud to achieve differentiated weight allocation.

Specifically, the Laplacian score is multiplied with the refined point features in order to obtain the desired recognition effect on the edge regions of the point cloud.

The entire structure represented by the formula is

$$y_i = \sum_{\boldsymbol{x_j} \in \boldsymbol{\chi(i)}} \rho(\varphi(\boldsymbol{x_i}) - \psi(\boldsymbol{x_j}) + \delta) \odot (\alpha(\boldsymbol{x_j}) + \delta) \odot (\boldsymbol{S_i} + 1) \tag{2}$$

where $\boldsymbol{S_i}$ denotes the Laplacian score of point cloud $\boldsymbol{x_i}$. We set the parameter as $\boldsymbol{S_i} + 1$ to avoid zero values of $\boldsymbol{S_i}$ in the central flat regions. Details on how to generate Laplacian scores for each point cloud are given in the next section.

2.3. Laplacian Score

We generate Laplacian score for point clouds based on graph Laplacian theory [12]. We regard single point cloud as the undirected graph $G = \{V, E, W\}$ that contains a vertex set V of cardinality $|V| = n$, the edge set E connects vertices and the adjacency W represents the weight of each point to other points. W is a real symmetric matrix, where W_{ij} denotes the weight between vertex V_i and vertex V_j. As shown in Fig.3, we find k nearest neighbors for each vertex in the graph. If vertex V_j belongs to the neighbor set of vertex V_i, the corresponding weight $W_{i,j}$ between vertex V_i and vertex V_j is computed. Otherwise, W_{ij} is zero.

We use the exponential function of the Euclidean distance to compute the weight between vertex V_i and vertex V_j:

$$W_{ij} = \begin{cases} exp(-\frac{\left\|V_i^c - V_j^c\right\|_2^2}{\sigma^2}), \left\|V_i^c - V_j^c\right\|_2^2 \leq \varepsilon \\ 0, \ otherwise \end{cases} , \tag{3}$$

where $V_i^c \in R^3$ is the 3D coordinate of vertex V_i. σ and ε are parameters where ε is a hyper parameter controlling the range of neighbor points of vertex V_i. Eq.(3) shows that the weight between two vertices is inversely proportional to their distance. With a closer distance, a greater weight is computed. We normalize the weight W_{ij} between vertex V_i and corresponding neighbor V_j as $\widetilde{W_{ij}} = W_{ij} / \sum_j W_{ij}$. Finally, the formula of Laplacian score for vertex V_i is

$$S_i = V_i - \sum_j \widetilde{W}_{ij} V_j, \tag{4}$$

where S_i means the difference between vertex V_i and the result weighted by vertex V_j belonging to the neighbor set of vertex V_i. The difference between a point and its neighbors is proportional to the Laplacian score. As shown in Fig.3, the regions of the aircraft with non-zero Laplacian score are concentrated in the junction regions of various parts. This means that larger weights are assigned to the points in this region, boosting the model to learn the features of these points with similar geometric information.

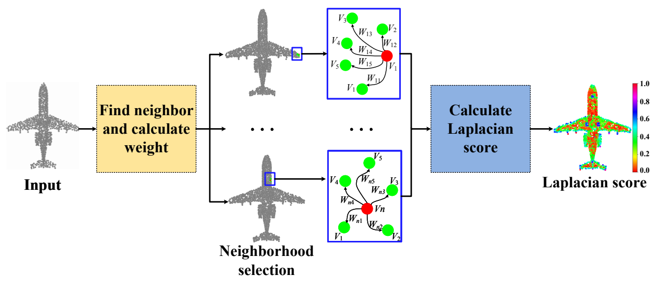
Input
Find neighbor and calculate weight
Neighborhood selection
V_3 V_2 V_4 V_5 V_1 W_{13} W_{12} W_{14} W_{15} W_{11}
V_5 V_4 V_3 Vn V_1 V_2 W_{n5} W_{n3} W_{n4} W_{n1} W_{n2}
Calculate Laplacian score
Laplacian score
1.0
0.8
0.6
0.4
0.2
0.0

Fig. 3. The overall pipeline of generating Laplacian score for point clouds. The green vertices represent the collection of neighbor points searched for the red vertex during the process of neighborhood selection.

3. EXPERIMENTAL RESULTS

3.1. Evaluation Methodology

We demonstrate the efficiency of the proposed model on the ShapeNet benchmark for part segmentation. The dataset contains 16,880 3D models categorized into 16 categories with 50 different parts across the dataset. We separate 14,006 models for training and 2874 models for testing. We uniformly sample 2048 points across the model surface. The settings of other hyperparameters remain consistent with [10]. We use Intersection-over-Union (IoU) to evaluate the proposed model and compare it with other algorithms. The IoU of a category is computed from the average IoU of all objects belonging to that category. Cls. avg. is the mean IoU averaged across all object categories, and ins. avg. is the mean IoU averaged across all test objects.

3.2. Part Segmentation Results on ShapeNet

Table 1 shows the accuracy comparison with respect to different algorithms performed on the ShapeNet dataset. It can be seen that KPConv method achieves the best average accuracy compared to other counterpart methods. After replacing the transformer layer employed in Point Transformer[10] with the Laplacian transformer module, the proposed framework achieves the best accuracy, outperforming the model KPConv by 0.4% in terms of ins.avg. While performing poorly on 8 out of 16 objects, such as airplane, cap, and car, the proposed model achieves the best accuracy on bag, chair, earphone, and lamp. Since the points distributed in the junction regions contained in these categories are easier to identify. As shown in Fig. 4, the Laplacian score calculated at the edge areas of different parts take higher values, which have an effect on rectifying the outcomes for the point clouds in these parts predicted by the model. At the same time, comparing the segmentation results of our method with the point transformer algorithm, our method has minor errors in the junction regions of various parts.

Table 1. Part segmentation results on ShapeNet. air.: airplane. cha.: chair. ear.: ear-phone. gui.: guitar. kni.: knife. lam.: lamp. lap.: laptop. mot.: motorbike. pis.: pistol. roc.: rocket. ska.: skateboard. tab.: table

Method	ins. avg.	air.	bag	cap	car	cha.	ear.	gui.	kni.	lam.	lap.	mot.	mug	pis.	roc.	ska.	tab.
PointMLP[13]	86.1	83.5	83.4	87.5	80.5	90.3	78.2	92.2	88.1	82.6	96.2	77.5	95.8	**85.4**	64.6	83.3	**84.3**
DGCNN[7]	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.8	63.5	74.5	82.6
PointNet++[4]	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
GAPNet[14]	84.7	84.2	84.1	**88.8**	78.1	90.7	70.1	91.0	87.3	83.1	96.2	65.9	95.0	81.7	60.7	74.9	80.8
PointCNN[15]	86.1	84.1	86.5	86	80.8	90.6	79.7	92.3	88.4	85.3	96.1	77.2	95.3	84.2	64.2	80.0	82.3
KPConv[16]	86.4	84.6	86.3	87.2	**81.1**	91.1	76.5	92.6	88.4	82.7	96.2	**78.1**	95.8	**85.4**	**69.0**	**82.0**	83.6
PointTransformer [10]	86.3	**84.9**	86.4	85.2	80.2	91.3	79.5	92.2	88.3	85.2	96.2	78.0	95.6	84.8	64.7	81.2	83.6
Ours	**86.8**	84.3	**86.6**	85.2	79.1	**91.5**	**81.2**	**92.9**	**88.5**	**85.9**	**96.3**	77.8	**95.9**	84.5	64.4	81.3	83.9

Table 2. Effectiveness of different number settings of ε in the definition of Eq.(3) and K in the definition of KNN.

ε	K	cls. avg.	ins. avg.
0.02	15	84.2	86.4
0.03	15	84.4	86.5
0.03	20	**84.9**	**86.8**
0.04	20	84.6	86.7
0.04	25	84.5	86.6

Table 3. Investigation of different operators.

Operators	cls. avg.	ins. avg.
Concatenation	81.7	84.5
Summation	83.4	85.7
Hadamard product	**84.9**	**86.8**

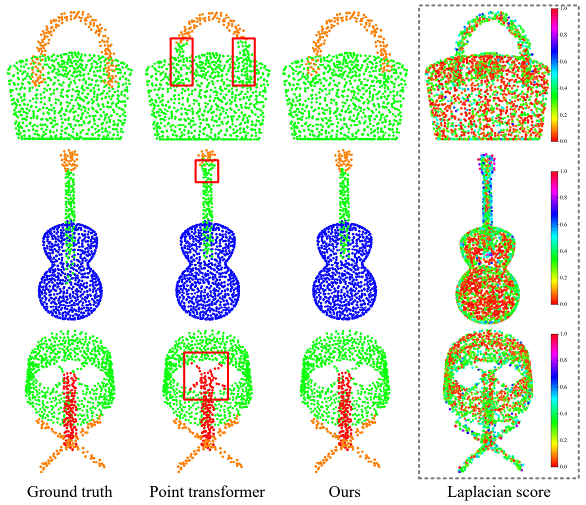

Fig. 4. Visualization of segmentation results of different methods. The Laplacian score indicates the attention of the Laplacian transformer module assigned to the whole point cloud object. The higher the Laplacian score of a region, the more attention the model pays to that region.

3.3. Ablation Study

Effects of different hyper-parameter settings. Concerning the effect of different settings of ε, as defined in Eq.(3), and K, as used in KNN, we test our model with various settings. As shown in Table 2, the performance improves as the parameters ε and K increase. The reason is that the Laplace score calculated for points located at the junction regions can more accurately reflect the disparity between the central point and its neighborhood when a larger range of neighboring points is considered. Also, a proper choice of the value of K can contain more representative neighbor points, which favors the extraction of local features. However, the performance deteriorates when the number of ε and K become larger.

Effects of different operators. We apply different operators that incorporate Laplacian score into the transformer structure to evaluate their performance. As shown in Table 3, Concatenation, Summation, and Hadamard product refer to the element-wise operations of concatenating over the channel, adding, and multiplying the Laplacian score with the refined points in the Laplacian transform module, respectively. The results presented in Table 3 demonstrate the superior performance of the Hadamard product over other operators. Unlike element-wise addition, the Hadamard product excels at greatly amplifying the weights assigned to points within the target region across the entire point cloud.

4. CONCLUSION

In this paper, in order to alleviate the ambiguity of the point cloud features embodied at the junction regions of various parts, we design a Laplacian transformer module based on graph Laplacian theory for contours point learning. This module reveals the implied spatial relationship between points and generates the corresponding Laplacian score for each point cloud, guiding the model to concentrate on points located at the junction regions. Based on the Laplacian transformer module, we propose a Unet-like transformer framework for part segmentation task. We have shown that the proposed framework is effective on public datasets and outperforms a series of mainstream approaches. In the future, we will explore a novel fusion of transformer and convolution techniques to enhance the performance of 3D segmentation in the task of semantic segmentation for real-world scenes.

5. REFERENCES

[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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.

[2] Chenlei Lv, Weisi Lin, and Baoquan Zhao, "Approximate intrinsic voxel structure for point cloud simplification," *IEEE Transactions on Image Processing*, vol. 30, pp. 7241–7255, 2021.

[3] Chenlei Lv, Weisi Lin, and Baoquan Zhao, "Intrinsic and isotropic resampling for 3d point clouds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[4] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[5] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem, "PointNeXt: Revisiting pointnet++ with improved training and scaling strategies," *arXiv preprint arXiv:2206.04670*, 2022.

[6] Chenxi Zhao, Weihao Zhou, Li Lu, and Qijun Zhao, "Pooling scores of neighboring points for improved 3d point cloud segmentation," in *Proceedings of the IEEE International Conference on Image Processing*, 2019, pp. 1475–1479.

[7] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions On Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[8] Kuangen Zhang, Ming Hao, Jing Wang, Xinxing Chen, Yuquan Leng, Clarence W de Silva, and Chenglong Fu, "Linked dynamic graph cnn: Learning through point cloud by linking hierarchical features," in *Procedings of the IEEE International Conference on Mechatronics and Machine Vision in Practice*, 2021, pp. 7–12.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[10] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun, "Point transformer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021, pp. 16259–16268.

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the IEEE Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[12] Siheng Chen, Dong Tian, Chen Feng, Anthony Vetro, and Jelena Kovačević, "Fast resampling of three-dimensional point clouds via graphs," *IEEE Transactions on Signal Processing*, vol. 66, no. 3, pp. 666–681, 2017.

[13] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu, "Rethinking network design and local geometry in point cloud: A simple residual mlp framework," *arXiv preprint arXiv:2202.07123*, 2022.

[14] Can Chen, Luca Zanotti Fragonara, and Antonios Tsourdos, "GapNet: Graph attention based point neural network for exploiting local feature of point cloud," *arXiv preprint arXiv:1905.08705*, 2019.

[15] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen, "PointCNN: Convolution on $\mathcal{X}$-transformed points," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[16] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6410–6419.