Local Enhanced Transformer Networks for Land Cover Classification With Airborne Multispectral LiDAR Data

Dilong Li, *Member, IEEE*, Shenghong Zheng, *Member, IEEE*, Ziyi Chen, *Senior Member, IEEE*, Jonathon Li, *Fellow, IEEE*, Lanying Wang, *Graduate Student Member, IEEE*, and Jixiang Du

***Abstract*— Transformer networks have demonstrated remarkable performance in point cloud processing tasks. However, balancing local feature aggregation with long-range dependency modeling remains a challenging issue. In this work, we present a local enhanced Transformer network (LETNet) for land cover classification with multispectral LiDAR data. Specifically, we first rethink position encoding in 3-D Transformers and design a novel feature encoding module that embeds comprehensive geometric and semantic information, serving a similar purpose. Then, the proposed local enhanced Transformer module is used to capture the accurate global attention weights and refine the features. Finally, to effectively extract and integrate global features across various scales, an attention-based pooling module is introduced. This module extracts global features from each encoder and decoder layer and constructs a feature pyramid to fuse these multiscale global features. Both quantitative assessments and comparative analyses demonstrate the competitive capability and advanced performance of the LETNet in land cover classification task.**

***Index Terms*— Airborne multispectral LiDAR, land cover classification, Transformer.**

I. Introduction

In RECENT years, the rapid advancements in 3-D sensor technologies have significantly enhanced the attention garnered by 3-D point clouds across diverse applications, including autonomous driving, robotics, urban scene interpretation, and cartography [1]. Compared with the regular single-wavelength LiDAR data, multispectral LiDAR technology provides the more comprehensive spectral information, which is critical for land cover classification task. Pioneering researchers, such as Wichmann et al. [2] and Gong et al. [3], initially assessed the feasibility of employing multispectral LiDAR data for land cover classification. Subsequent studies [4], [5], [6], [7], [8] further validated the effectiveness and achieved decent performance.

The significant success of deep learning techniques in image processing has propelled the development of deep learning methods in the field of point cloud processing. PointNet [9] revolutionized the field of raw point cloud processing by employing pointwise MLPs for feature extraction and leveraging the permutation invariance of symmetry functions to overcome the inherent drawbacks of point clouds compared with regular grid data. As the extension of PointNet, PointNet++ [10] constructed a hierarchical network that iteratively implemented PointNet to learn the local features and combined the learned features from multiple scales and different layers to achieve better performance. The following studies [11], [12], [13], [14], [15] expanded this branch from various aspects. Nevertheless, most of them focus on the local feature learning and aggregation, but fail to learn the global context from long-range dependencies [16].

Due to the remarkable long-range context learning ability, Transformer modules have demonstrated considerable potential for point cloud processing [17]. Several studies [16], [18], [19], [20], [21], [22] make a profound explore in point cloud processing with Transformer architectures. These 3-D Transformers can be classified into two categories according to the operating scale. For local 3-D Transformers, which utilize the self-attention mechanism in the local region, such as [19], most of them are still difficult to directly capture long-range contexts because of the limited receptive field. For global 3-D Transformers, they avoid this drawback by applying the self-attention mechanism to all input points. However, most of the existing global 3-D Transformers directly feed the input features into Transformer blocks, but ignore the influence of local neighboring features.

In this letter, we propose an LETNet for land cover classification with multispectral LiDAR data. The main contributions are summarized as follows.

1) We propose a feature encoding module, which could be regarded as the position encoding in 3-D Transformers. The module consists of two operations, the geometric feature encoding and semantic feature encoding,

Manuscript received 26 June 2024; accepted 16 July 2024. Date of publication 24 July 2024; date of current version 30 August 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 42201475 and Grant 62001175, in part by the Natural Science Foundation of Fujian Province under Grant 2021J05059 and Grant 2023J01135, and in part by the Fundamental Research Funds for the Central Universities of Huaqiao University under Grant ZQN-1114. *(Corresponding author: Ziyi Chen.)*

Dilong Li, Shenghong Zheng, Ziyi Chen, and Jixiang Du are with the College of Computer Science and Technology, Fujian Key Laboratory of Big Data Intelligence and Security, Xiamen Key Laboratory of Computer Vision and Pattern Recognition, Xiamen Key Laboratory of Data Security and Blockchain Technology, Huaqiao University, Xiamen 361021, China (e-mail: scholar.dll@hqu.edu.cn; 21013083034@stu.hqu.edu.cn; chenziyihq@hqu.edu.cn; jxdu@hqu.edu.cn).

Jonathon Li and Lanying Wang are with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: junli@uwaterloo.ca; lanying.wang@uwaterloo.ca).

Digital Object Identifier 10.1109/LGRS.2024.3432870

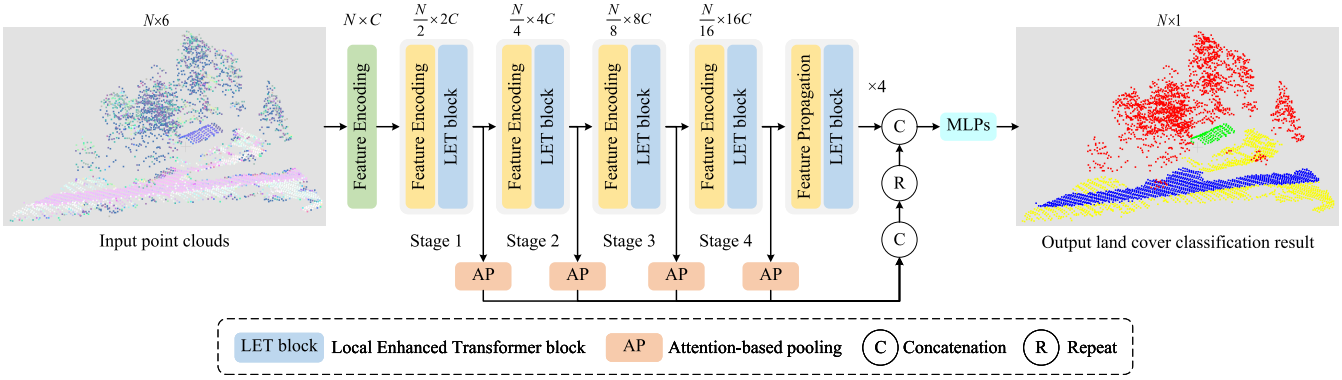

Fig. 1. Overall architecture of LETNet.

which contributes to the module learning the latent geometric representations and comprehensive semantic features.

2) We propose a novel local enhanced Transformer module to obtain more accurate global attention via considering local neighboring features.
3) We propose an attention-based pooling operator that pools global features from each layer of the encoder and decoder, constructing a feature pyramid with these global features to enhance the fusion of multiscale features effectively.

II. Methodology

A. Network Architecture

The overall architecture of LETNet is shown in Fig. 1. (N, D) represent the number of input points and their dimension, respectively. Before the feature encoder stages, we first apply the feature encoding module (without sampling) to convert the feature dimension from 6 to C. C is set to be 64 here. The feature encoder of LETNet is divided into four stages. At the beginning of each stage, the farthest point sampling (FPS) method [23] is utilized to downsample the input point cloud. The ratio of downsampling is set to be 2, resulting in the numbers of points in the encoder stages being [N/2, N/4, N/8, N/16]. With the points and features output from previous layer and the downsampled points and features, the feature encoding module groups and aggregates the local geometric and semantic features and doubles the dimension at the same time. Then, the local enhanced Transformer module is implemented to refine the features, resulting in output feature dimensions of [2C, 4C, 8C, 16C] for the encoder stages. For decoder (or feature upsampling) stages, we also stack the local enhanced Transformer module after the feature propagation operation. To better fuse the multiple scale features, the attention-based pooling module is implemented to obtain the global feature representations of each encoder and decoder layer. The pooled features are concatenated as the global feature. The global feature is repeated by the number of points and then concatenated with the output of the last decoder layer. Finally, an MLP is applied to map the feature to the final logits.

B. Feature Encoding

Since PointNet++ [10] introduced the hierarchical structure and set abstraction operation, numerous following studies focus on the enhancement of local feature learning and aggregation. Specifically, some studies make efforts to encode the local geometric feature by the revamp of points' geometric relation, such as RSCNN [12] and DGCNN [11], and some other studies pay attention on the enhancement of feature encoding operation, such as PointMLP [24]. Unlike the most of previous models process the coordinate and feature together, the proposed feature encoding module encodes the geometric feature and semantic feature separately. As shown in Fig. 2, the input points and features are processed by the geometric feature encoding module and semantic feature encoding module, respectively; then, the outputs of are concatenated into an MLP.

1) Geometric Feature Encoding: For Transformer models, position encoding plays an important role. But, unlike the regular position encoding in NLP or 2-D computer vision, the position encoding in 3-D computer vision considers more complicated spatial geometry relationship than the literal "position"; for example, the PT [19] introduced trainable position encoding. Here, we propose a geometric feature encoding module to encode geometric information efficiently, which plays the same role as position encoding.

A given point cloud is represented as a set of points $\{P_i | i = 1, 2, \ldots, n\}$, where each point P_i is given by its coordinates in R^3. Then, the point cloud is downsampled by the FPS method. For downsampled point P_i', we construct the local directed graph by K-nearest neighbors (KNN) algorithm, which is formulated as follows:

$$P_i'^k = \left(P_i^k - P_i'\right) \tag{1}$$

where k represents the number of neighborhoods.

To better explore the latent geometry information, we adopt the geometric moments representation of point clouds proposed in [8] for local geometric feature encoding. The $p + q + r$ orders geometric moments representation of point clouds is defined as the set of $x^p y^q z^r$. Here, we use the first- and second-order geometric moments of point clouds, which

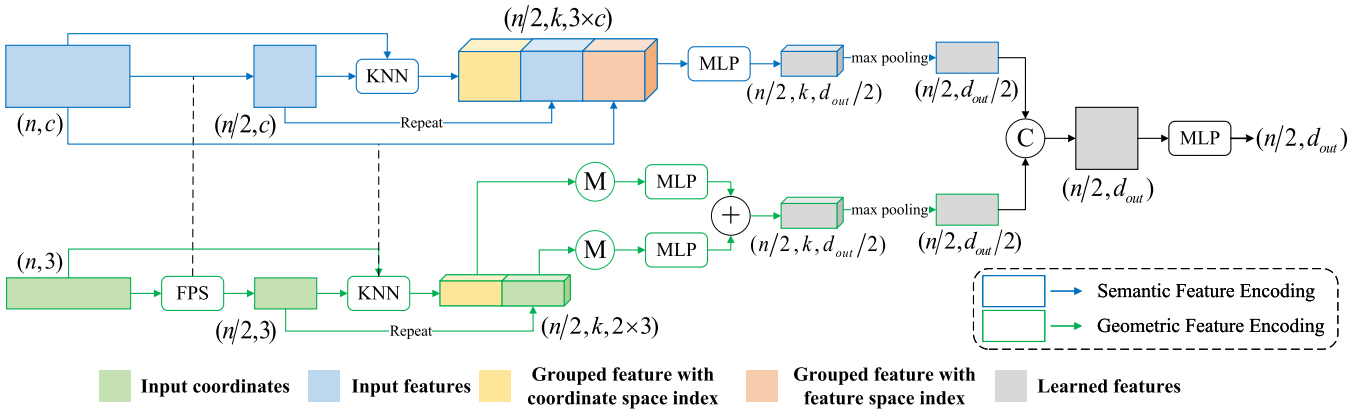

Fig. 2. Feature encoding module. M indicates the geometric moments representation, $+$ indicates the addition operation, and C indicates the concatenation operation.

is represented as follows:

$$M_1 = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad M_2 = \begin{bmatrix} xy \\ xz \\ yz \\ x^2 \\ y^2 \\ z^2 \end{bmatrix}. \tag{2}$$

Similarly, the geometric moments representation of directed edges can be calculated. Given the geometric moments representation of the downsampled points and corresponding directed edges, two MLPs are implemented to learn the high-level geometric features, respectively. Then, the learned features are fused by the channelwise addition operation, which is defined as follows:

$$F_g = \text{add}\big(\text{MLP}\big(M\big(P_i'^k\big)\big), \text{MLP}\big(M\big(k \cdot P_i'\big)\big)\big) \tag{3}$$

where M represents the geometric moments representation and F_g is the fused geometric feature. Finally, the max-pooling operation is applied to aggregate the local geometric features.

2) Semantic Feature Encoding: With the point cloud downsampling results, the given semantic features $\{F_i | i = 1, 2, \ldots, n\}$ are also downsampled as F_i'. To better represent the local semantic features, we not only group the k nearest neighbors in spatial space but also group in the feature space. Then, the grouped features and the downsampled features are concatenated to feed into an MLP, which is formulated as follows:

$$F_s = \text{MLP}\big(\text{concat}\big(F_i'^k, \widetilde{F}_i'^k, k \cdot F_i'\big)\big) \tag{4}$$

where $F_i'^k$ and $\widetilde{F}_i'^k$ represent the features grouped in spatial space and feature space, respectively, and F_s is the learned semantic feature. Finally, we also utilize the max-pooling operation to aggregate the local semantic features.

The local geometric feature F_g and local semantic feature F_s are fused by the concatenation operation. After the feature fusion, an MLP is applied to increase the robustness of the module.

C. Local enhanced Transformer

Due to the remarkable global feature learning ability, global Transformer modules are widely used for point cloud analysis, like PCT [18]. However, most of these previous works obtain the attention weights matrix from input feature map directly, and ignore the impact of nearby features. To this end, we propose the local enhanced Transformer module to measure the similarity of tokens more accurate.

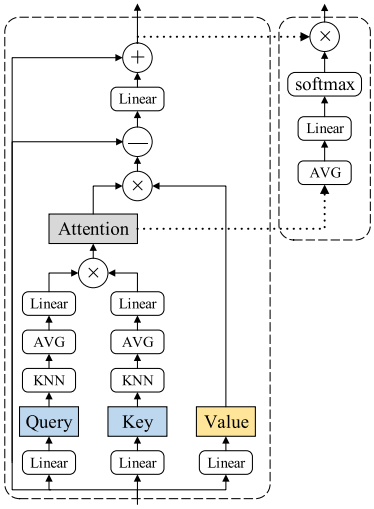

Fig. 3. (Left) Local enhanced Transformer module and (right) attention-based pooling module. AVG indicates the average pooling operation, and $-$ and $+$ indicate the channelwise minus and addition operations, respectively.

As shown in Fig. 3, given the input features $F_{in} \in R^{N \times d}$, let Q, K, V be the query, key, and value respectively, the attention weights are obtained as follows:

$$\begin{aligned} q &= \text{Linear}_q(F_{in}), k = \text{Linear}_k(F_{in}), \\ Q &= \text{Linear}_Q(\text{avg}(knn(q))), \\ K &= \text{Linear}_K(\text{avg}(knn(k))), \\ A &= \text{softmax}\big(Q \cdot K^T\big), \end{aligned} \tag{5}$$

where q and k are obtained from F_{in} by two independent linear layers, knn represents the grouping operation in spatial space, avg represents the average pooling operation, A is the attention weights matrix. The self-attention output features F_{sa} are the weighted summation of the attention weights matrix A and value.

$$\begin{aligned} V &= \text{Linear}_v(F_{in}) \\ F_{sa} &= A \cdot V \end{aligned} \tag{6}$$

To sharpen the attention weights and reduce the influence of noise, we also calculate the offset between self-attention

TABLE I
RESULTS OF COMPARISON METHODS

Model		Road	Grass	Tree	Building	OA(%)	Kappa
PointNet [9]	PA	74.2	79.4	90.7	63.8	84.3	0.741
	UA	58.0	89.3	92.1	39.6		
PointNet++ [10]	PA	74.4	86.9	94.2	66.7	88.3	0.811
	UA	77.0	91.1	93.5	51.1		
DGCNN [11]	PA	88.3	89.1	94.5	83.8	91.6	0.862
	UA	74.2	94.0	97.2	62.9		
RS-CNN [12]	PA	91.5	91.4	97.6	93.0	94.7	0.914
	UA	81.0	96.7	97.9	81.5		
RandLA-Net [15]	PA	86.0	90.5	96.1	82.7	92.5	0.878
	UA	80.9	94.0	96.2	75.0		
PCT [18]	PA	94.7	93.0	97.0	93.3	95.3	0.923
	UA	80.2	97.1	98.9	82.3		
PT [19]	PA	96.2	95.0	**99.1**	96.3	97.3	0.956
	UA	86.0	**98.1**	99.4	**94.9**		
LETNet	PA	**96.4**	**95.4**	99.0	**97.9**	**97.53**	**0.960**
	UA	**87.0**	98.0	**99.7**	**94.9**		

features and input features like PCT [18] did. Then, the offset is fed into a linear layer and added with the input features, which is formulated as

$$F_{\text{out}} = \text{Linear}_o(F_{sa} - F_{in}) + F_{in} \tag{7}$$

where F_{out} is the output features. The whole attention based pooling module can be formulated as

$$F_p = \text{softmax}\big(\text{Linear}_p(\text{avg}(A))\big) \times F_{\text{out}} \tag{8}$$

where F_p is the pooled global features.

III. RESULTS AND ANALYSIS

A. Dataset

The dataset selected for this study is situated within the town of Whitchurch Stouffville in Ontario, Canada, precisely positioned at 43°58′00″ latitude and 79°15′00″ longitude. Following the method of [25], we identify 13 representative areas, selecting areas 6 and 7 for testing purposes and allocating the remainder for training. To fulfill the objectives of land cover classification, we relabel the chosen areas into four classes: tree, building, grass, and road. In order to thoroughly assess the performance of the proposed LETNet, we employed four common quantitative evaluation metrics typically utilized in land cover classification tasks. These metrics encompass overall accuracy (OA), the Kappa index, producer accuracy (PA), and user accuracy (UA).

B. Implementation Details

We utilize the PyTorch library to implement LETNet on RTX 4090 GPUs. We use the SGD optimizer with a cosine annealing scheduler without the warm restart. The initial learning rate is set to 0.01 and the minimum learning rate to 0.0001. We conduct training for a maximum of 400 epochs with a batch size of 8.

C. Performance

The experimental results are presented in Table I, and LETNet achieves impressive OA and Kappa index of 97.53% and 0.960, surpassing all comparative methods. In comparison with PT and PCT, LETNet surpasses PCT by 2.23% points on OA and PT by 0.23% points. Regarding the Kappa metric, LETNet outperforms PCT by 0.037 and PT by 0.004. In addition, on the PA and UA metrics, LETNet demonstrates superior performance across most categories. Fig. 4 shows the visualizations of classification results. The visualization results of LETNet closely match the ground truth. By magnifying local areas for comparison, we also arrive at the same phenomenon. These experimental results further demonstrate LETNet's robust geometric extraction capabilities.

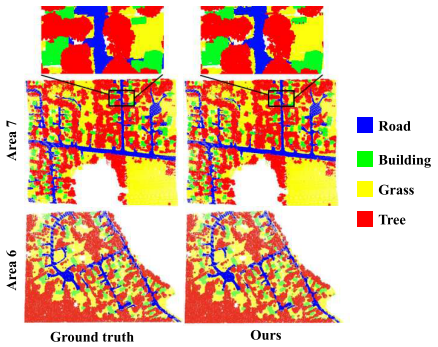

Fig. 4. Visualization of land cover classification results on areas 6 and 7.

TABLE II
ABLATION STUDY OF THE KEY COMPONENTS

Model	GFE	SFE	GT	GA	OA(%)	Kappa
A					95.60	0.928
B	✓				96.60	0.945
C	✓	✓			97.03	0.952
D	✓	✓	✓		97.47	0.959
E	✓	✓	✓	✓	**97.53**	**0.960**

TABLE III
ABLATION STUDY OF GEOMETRIC FEATURE ENCODING

Geometric feature encoding		Road	Grass	Tree	Building	OA(%)	Kappa
Coordinates	PA	**96.6**	95.4	98.9	96.8	97.44	0.959
	UA	87.4	**98.1**	99.5	94.1		
EdgeConv	PA	95.6	**95.8**	98.9	96.2	97.38	0.958
	UA	**89.1**	97.7	99.3	93.4		
Geometric Moments	PA	96.4	95.4	**99.0**	**97.9**	**97.53**	**0.960**
	UA	87.0	98.0	**99.7**	**94.9**		

D. Ablation Study

We perform ablation experiments on the dataset to assess the effectiveness of model components and the influence of parameter settings.

1) Key Components of LETNet: LETNet is comprised of four main components: geometric feature embedding (GFE), semantic feature embedding (SFE), global Transformers (GTs), and global aggregator (GA). As shown in Table II, baseline model A only achieves an OA of 95.60% and a Kappa index of 0.928. With the GFE, model B exhibits significant improvement, achieving higher accuracies with an OA of 96.60% and a Kappa index of 0.945. By introducing the SFE, model C accomplishes an OA of 97.03% and a Kappa index of 0.952. Subsequently, model D attains an OA of 97.47% and a Kappa index of 0.959 by adding the GT. Finally, with the GA module, model E, which is also LETNet, achieves the best OA and Kappa index of 97.53% and 0.960.

2) Geometric Feature Encoding: We then investigate the influence of various geometric feature encoding strategies, and the results are presented in Table III. LETNet reaches an

TABLE IV
ABLATION STUDY OF LOCAL NEIGHBORHOOD NUMBER k

k		Road	Grass	Tree	Building	OA(%)	Kappa
8	PA	96.4	**95.6**	98.8	96.6	97.41	0.958
	UA	**88.0**	97.3	99.6	95.7		
16	PA	**97.5**	94.8	99.0	**98.2**	97.46	0.959
	UA	85.3	**98.3**	99.7	95.6		
24	PA	96.7	95.0	**99.1**	97.7	97.46	0.959
	UA	86.0	98.1	99.6	**95.8**		
32	PA	96.4	95.4	99.0	97.9	**97.53**	**0.960**
	UA	87.0	98.0	**99.7**	94.9		

OA of 97.44% when only using point cloud coordinates for geometric feature encoding. The utilization of EdgeConv [11] leads to an decrease of 0.09% compared with the coordinates. The integration of our proposed geometric feature encoding module elevates the accuracy to 97.53%. The improvement demonstrates the superior of the proposed geometric feature encoding module. Meanwhile, our proposed module achieves the highest score on the Kappa index and also obtains the highest subindex scores within the tree and building subcategories.

3) Number of Neighbors: We also explore the parameter setting of neighbors k. Based on the results presented in Table IV, our observations indicate that the LETNet attains optimal performance when k is set to 32. This value potentially strikes a superior balance between noise and receptive field compared with alternative settings.

IV. CONCLUSION

In this letter, we propose LETNet for land cover classification with multispectral LiDAR data. The proposed method mainly contains three key modules: feature encoding module, local enhanced Transformer module, and attention-based pooling module. The feature encoding module efficiently embeds the geometric and semantic information at the beginning of each feature encoder layer. Then, the local enhanced Transformer module is applied to learn the long-range contexts and refine the feature. With the attention-based pooling module and feature pyramid construction, the proposed model can further fuse the global features extracted from each encoder and decoder layers. The extensive experimental results show that the proposed LETNet achieves promising performance on land cover classification task and validate the effectiveness and superiority of the proposed modules.

REFERENCES

[1] W. Y. Yan, A. Shaker, and N. El-Ashmawy, "Urban land cover classification using airborne LiDAR data: A review," *Remote Sens. Environ.*, vol. 158, pp. 295–310, Mar. 2015.

[2] V. Wichmann, M. Bremer, J. Lindenberger, M. Rutzinger, C. Georges, and F. Petrini-Monteferri, "Evaluating the potential of multispectral airborne LiDAR for topographic mapping and land cover classification," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. II-3, pp. 113–119, Aug. 2015.

[3] W. Gong et al., "Investigating the potential of using the spatial and spectral information of multispectral LiDAR for object classification," *Sensors*, vol. 15, no. 9, pp. 21989–22002, Sep. 2015.

[4] K. Bakuła, P. Kupidura, and Ł. Jełowicki, "Testing of land cover classification from multispectral airborne laser scanning data," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 41, pp. 161–169, Jun. 2016.

[5] S. Morsy, A. Shaker, and A. El-Rabbany, "Multispectral LiDAR data for land cover classification of urban areas," *Sensors*, vol. 17, no. 5, p. 958, Apr. 2017.

[6] T.-A. Teo and H.-M. Wu, "Analysis of land cover classification using multi-wavelength LiDAR system," *Appl. Sci.*, vol. 7, no. 7, p. 663, Jun. 2017.

[7] S. Pan et al., "Land-cover classification of multispectral LiDAR data using CNN with optimized hyper-parameters," *ISPRS J. Photogramm. Remote Sens.*, vol. 166, pp. 241–254, Aug. 2020.

[8] D. Li et al., "AGFP-net: Attentive geometric feature pyramid network for land cover classification using airborne multispectral LiDAR data," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 108, Apr. 2022, Art. no. 102723.

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.

[10] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–10.

[11] 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 Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Oct. 2019.

[12] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8895–8904.

[13] 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 *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6411–6420.

[14] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 828–838.

[15] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

[16] X. Lai et al., "Stratified transformer for 3D point cloud segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8500–8509.

[17] D. Lu, Q. Xie, M. Wei, K. Gao, L. Xu, and J. Li, "Transformers in 3D point clouds: A survey," 2022, *arXiv:2205.07417*.

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

[19] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 16259–16268.

[20] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 19313–19322.

[21] D. Lu, K. Gao, Q. Xie, L. Xu, and J. Li, "3DPCT: 3D point cloud transformer with dual self-attention," 2022, *arXiv:2209.11255*.

[22] 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 Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2158–2168, Feb. 2023.

[23] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Trans. Image Process.*, vol. 6, no. 9, pp. 1305–1315, Sep. 1997.

[24] 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," 2022, *arXiv:2202.07123*.

[25] D. Li et al., "Building extraction from airborne multi-spectral LiDAR point clouds based on graph geometric moments convolutional neural networks," *Remote Sens.*, vol. 12, no. 19, p. 3186, Sep. 2020.