Draft (GRSL-01946-2024.R1)

# A dual-branch deep learning framework at the grid scale for individual tree segmentation

Ze Ding, Huaiqing Zhang, *Senior Member, IEEE*, RuiSheng Wang, *Senior Member, IEEE*, Li Zhang, Hanxiao Jiang and Ting Yun, *Member, IEEE*

***Abstract*—Individual tree segmentation from point clouds is essential for diverse forest applications. A dual-branch segmentation deep learning network operating at the grid scale was proposed, which includes the semantic segmentation branch for partitioning point clouds of tree trunks and the instance segmentation branch for individual trunk extraction. Meanwhile, the network analyzes input forest points at the grid scale instead of pointwise processing to preserve local geometric information of the forest points while reducing computational load. After extraction of each tree trunk in the understory layer using our network, a hierarchical k-nearest neighbors algorithm based on the extracted trunk parts was employed to accomplish individual tree segmentation. For the forest plots, our proposed approach achieves precision, recall, F1-score, and MIoU of 89.66 %, 89.13 %, 89.40 %, and 90.84 %, respectively. These results represent a significant improvement in accuracy and rapid execution capability compared to prior methods.**

***Index Terms*—Forest, deep learning network, individual tree segmentation, LiDAR**

## I. INTRODUCTION

AS integral parts of urban parks, landscape forests are vital in enhancing air quality, local climates, and urban environments [1]. With the progression of technology, light detection and ranging (LiDAR) scanning systems have become increasingly prevalent in urban parks. LiDAR can effectively penetrate forests and acquire more accurate 3D information than manual surveying and satellite remote sensing. However, the forest point clouds collected by LiDAR are a whole block of data and cannot automatically exact individual tree objects. Hence, single-tree segmentation from the collected point clouds is a prerequisite for gaining growth parameters at the individual tree scale.

Segmenting individual trees via terrestrial laser scanning (TLS) can be meaningful and challenging. The relevant studies can be divided into two categories. The first type of approach is mainly based on traditional machine learning methods. Relevant studies have shown that clustering-based methods can segment urban trees more accurately [2, 3]. The learning of trunk and canopy structure features through nonlinear least squares algorithms [4, 5], the learning of geometric features(e.g., normal vector and curvature) in dense point clouds via improved 3-dimensional characterization of local domain points [6, 7], and the learning of rod-like features of trunks via optimized min-cut algorithms [8] have been frequently used in individual tree segmentation works.

Second, many deep learning methods have also been applied to tree segmentation in forestry scenarios. Some studies have performed semantic segmentation by using point features to separate tree points from other points [9]. The features of points can be learned through point convolution, and combined with deep learning frameworks and clustering techniques to further improve the accuracy of the segmentation results [10, 11].

Despite extensive research, developing an automated framework to extract individual trees from urban point clouds remains challenging. There are several reasons: (1) There are complex structures in forests, such as mutual occlusion, misalignment, and the absence of certain areas. Additionally, underlying vegetation produces some noise which makes accurate tree segmentation more difficult. (2) Many methods incorporate deep learning, such as frameworks for 3D object detection or 3D semantic segmentation. However, the deep learning network that fuses two purposes and enables them to work together synergistically is not very prevalent.

The contributions in the work are summarized as follows: A novel dual-branch deep learning network at the grid scale is devised for parallelly semantic and instance segmentation of trunks in sub-layer forests from point clouds. A hierarchical k-nearest neighbors algorithm is introduced to segment single trees based on the trunk instances extracted from our network and guided by the morphology of tree skeletons.

## II. METHODOLOGY

### *A. Data Acquisition and Preprocessing for the Scanned Points*

The study area was the Shanghai Botanical Garden located in Shanghai, China, as shown in Fig. 1. We used a mobile laser scanning (MLS) system named LIBACKPACK DGC50 to conduct scanning work. Point clouds were obtained for part of the park with a resolution of 500 pts/$m^3$. The park features diverse tree species and multi-layered vegetation. The intricacies of this park enabled us to test the efficacy of our approach and compare the performance with other methods.

The acquired point clouds used Gaussian filtering to remove some of the noisy points. Then, a ground interpolation

This study was financially supported by the National Natural Science Foundation of China (grant numbers 32371876 and 32271877), the Natural Science Foundation of Jiangsu Province (BK20221337), China, the Jiangsu Provincial Agricultural Science and Technology Independent Innovation Fund Project (CX(22)3048). *(Corresponding author: Ting Yun)*

Ze Ding, Li Zhang, and Ting Yun are with the College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, CN. (e-mail: minorele@njfu.edu.cn; lizhang@njfu.edu.cn; yunting@njfu.edu.cn).

Huaiqing Zhang is with the Research Institute of Forest Resources Information Techniques, Chinese Academy of Forestry, Beijing 10009l, CN.

Ruisheng Wang is with the Department of Geomatics Engineering, University of Calgary, AlbertaT2N 1N4, Canada.

Hanxiao Jiang is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, US.

algorithm was used to convert the digital surface model to the canopy height model. The point clouds were then classified into aboveground and ground points via the hybrid regression technique [12]. Finally, as shown in Fig. 2, we extracted the point clouds below 2 m and split it into voxels, as shown in Fig. 2(a)-(b). These voxels were subdivided into multiple grids labeled with information and served as our dataset I, as shown in Fig. 2(c)-(d). The dataset I includes 1862 trees on 0.12 km$^2$. Meanwhile, WHU-Urban3D is used as our dataset II [13]. 70% data were used for the training set, and 30% for the testing set. The training dataset was composed of trunk instances (colorful grids) and miscellaneous (gray girds).

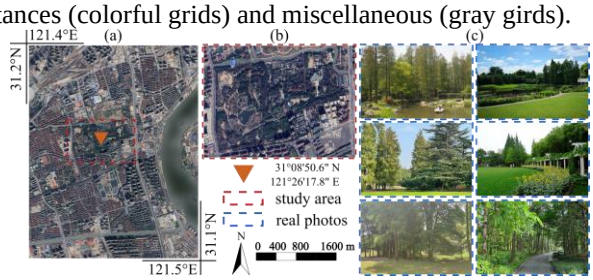


**Fig. 1.** Information about the study area. (a) Location of the study area. (b) Remote sensing images with latitude, longitude, and scale. (c) Real photos.

*B. Specific Grid Parameter*

Given a voxel containing point clouds, denoted as $\mathcal{P} \in \mathbb{R}^{N_p \times 3}$, each point $p_i(x_i, y_i, z_i) \in \mathcal{P}$ contains coordinate information, where $N_p$ is the total number of points and $i = 1, 2, ..., N_p$. Then, we partition the voxel into grids $\mathcal{G} \in \mathbb{R}^{D \times H \times W}$. For each voxel, comprising $\mathrm{M}_1 = D \times H \times W$ grids with an edge length of 2 cm, forms a 3-dimensional tensor $\mathcal{T} \in \mathbb{R}^{D \times H \times W}$, and each element in the tensor denotes the number of points contained in the corresponding grid, as shown in Fig. 2(c)-(d). Consequently, the tensor $\mathcal{T}$ serves as input to the constructed deep learning network.

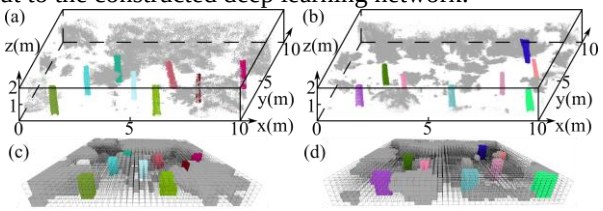


**Fig. 2.** Illustration of the training sets. (a)-(b) Point clouds in voxel labeled with trunk instances (colorful points are trunks, gray points are miscellaneous). (c)-(d) Each voxel is subdivided into multiple grids, and each grid is labeled as training samples (colorful grids are trunks, gray grids are miscellaneous, and subdivision using a larger grid size is employed here for better visual effects).

*C. Dual-branch Deep Learning Network at the Grid Scale*

We build a dual-branch deep learning network at the grid scale comprising two branches and the architecture of our model is shown in Fig. 3. The semantic segmentation branch is shown in the red in Fig. 3(b), which is designed to segment each trunk grid and non-trunk grid, and the instance segmentation branch is designed to segment trunk instances, which is shown in the blue in Fig. 3(c). Semantic segmentation is mature and efficient. So, the network extracts partial features using semantic segmentation, and as a heuristic information to assist the instance segmentation.

**1) The semantic segmentation branch**: This branch aims to extract semantic information at the grid scale; i.e., each grid contains semantic information about the trunk or others.

We place tensor $\mathcal{T}$ in the semantic segmentation branch of the network. The structure comprises a downsampling section (cyan part) and an upsampling section (purple part), which are connected by skip connections. The downsampling section follows a typical convolutional network architecture, increasing the network depth through repeated convolution operations to more effectively capture complex features. At each downsampling step, we increase feature channels. For each step of the upsampling section, transposed convolutions are performed to upsample and appropriately reduce the number of feature channels. Two sections all pass through the rectified linear unit (ReLU) functions and batch normalization processes. Finally, we obtain feature output with different scales and dimensions in the upsampling section, which are denoted as $\mathbf{F}_5 \in \mathbb{R}^{256 \times M_5}$ to $\mathbf{F}_1 \in \mathbb{R}^{96 \times M_1}$ from coarse to fine scales, respectively. In addition, $\mathbf{F}_5$ to $\mathbf{F}_1$ are used as part of the input for the instance segmentation branch.

**2) The instance segmentation branch**: This branch further learns instance information at the grid scale based on the outputs of the semantic segmentation branch; i.e., grids with the same trunk semantics are divided into different trunk instances. The branch contains two modules introduced in detail below.

The semantics & instance module (SIM) shown in Fig. 3(d), has two inputs and its main role is to predict instance heatmap and semantic label probabilities. One of its inputs includes the features from $\mathbf{F}_5$ to $\mathbf{F}_1$, which are obtained by the semantic segmentation branch. The other input is denoted $\mathbf{X}_{in} \in \mathbb{R}^{\hat{K} \times 3}, \hat{K} = 50$, obtained by the farthest point sampling (FPS) technique on the elements of the original tensor $\mathcal{T}$ whose values are not zero, where 3 is the index of selected elements. $\hat{K}$ is the predicted number of trunk instances and larger than the number of actual trunk instances. The positional encoding module then takes $\mathbf{X}_{in}$ as its input, and $\mathbf{X}_{pe} \in \mathbb{R}^{\hat{K} \times 256}$ is the resulting output. The module contains a linear function that expands the dimension to 256 and a positional encoding function [14] as follows:

$$\mathbf{X}_{pe} = PosEnc(linear(\mathbf{X}_{in})) \tag{1}$$

where $linear(\cdot)$ maps the input $\mathbf{X}_{in}$ to 256 dimensions as $\mathbf{X}_{256} \in \mathbb{R}^{\hat{K} \times 256}$, and the $PosEnc(\cdot)$ is shown below:

$$PosEnc(\mathbf{X}^i_{256}) = \begin{cases} \sin(\mathbf{X}^i_{256} / 10000^{2k/256}), & \text{if } i = 2k \\ \cos(\mathbf{X}^i_{256} / 10000^{2k/256}), & \text{if } i = 2k+1 \end{cases} \tag{2}$$

where $i$ is the position of the column for $\mathbf{X}^i_{256}$; 256 is the size of feature dimension; and $k$ is a sequence of natural numbers from $[0, 256/2)$. Finally, $\mathbf{X}_{pe}$ passes through the multilayer perceptron (MLP), which outputs the query $\mathbf{Q} \in \mathbb{R}^{\hat{K} \times 256}$.

On the one hand, to predict instance masks, the highest-resolution feature map $\mathbf{F}_1$ is used in conjunction with $\mathbf{Q}$. An

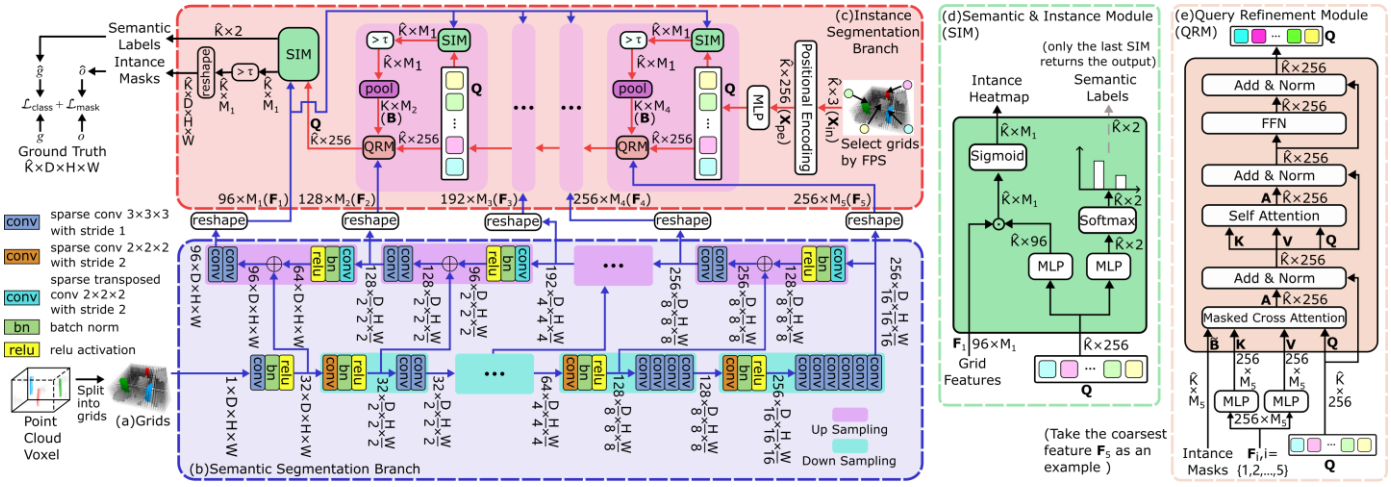


**Fig. 3.** The architecture of deep learning network. (a) Grids with labeled information. (b)-(c) The specific structure of the semantic segmentation branch and instance segmentation branch. (d)-(e) The specific structure of the semantic & instance module and query refinement module.

instance heatmap is generated from $\mathbf{Q}$ after passing through an MLP layer and a sigmoid function. Eventually, the binary instance masks $\mathbf{B} \in \mathbb{R}^{\hat{K} \times M_1}$ are calculated as follows:

$$\mathbf{B} = \{b_{i,j} = (\sigma(\mathbf{F_1} \cdot \mathrm{MLP}(\mathbf{Q}))_{i,j} > \tau)\} \tag{3}$$

where $\sigma(\cdot)$ is a sigmoid function; and $\tau$ is a thresholding scalar that is set to 0.5. Equation (3) is shown in the left subplot of Fig. 3(d) (with a green background).

On the other hand, to determine the semantic labels, $\mathbf{Q}$ passes through an MLP layer to 1+1 dimensions(one trunk label and one background). The probabilities of the semantic labels are then computed via the softmax function. Eventually, together with the predicted instance masks, we obtain the semantic and instance information of the girds.

The main role of the query refinement module (QRM), shown in Fig. 3(e), is to further refine the query. We utilize the decoder of the transformer as the core component, which is composed of four parts. Each part corresponds to the task of processing features at four different resolutions (here, we take the coarsest resolution feature $\mathbf{F}_5$ as an example). $\mathbf{F}_5$, namely, the output features of the semantic segmentation branch, is first passed through two MLP layers, resulting in $\mathbf{K}, \mathbf{V} \in \mathbb{R}^{M_5 \times 256}$ respectively. Each layer begins by applying masked cross-attention to one of the features obtained from the semantic segmentation branch. The masked cross-attention $\mathbf{A}$ is calculated as follows:

$$\mathbf{A} = \mathrm{softmax}(\mathbf{Q}\mathbf{K}^T / \sqrt{256} + \tilde{\mathbf{B}}) \cdot \mathbf{V} \tag{4}$$

$$\tilde{\mathbf{B}}(i,j) = \begin{cases} -\infty, \text{ if } \mathbf{B}(i,j) = 0 \\ 0, \quad \text{otherwise} \end{cases} \tag{5}$$

where 256 is the dimension of $\mathbf{Q}$ .The attention paid to grids that are masked as zero can be effectively reduced by adding the matrix $\tilde{\mathbf{B}}$ . After the masked cross-attention step, the obtained result is added to the $\mathbf{Q}$ of the same dimension and undergoes layer normalization (Add & Norm). Then, standard self-attention is performed again. We then pass the output through a feedforward network (FFN). Finally, we get the output of the refined query.

**3) Loss functions**

Since there is no inherent order between the ground truth and predicted values, we need to establish a correspondence between the two sets during training. To accomplish this, we employ bipartite graph matching, which is computed via the Hungarian algorithm. The number of ground-truth trunk instances pad with no object of the same size as $\hat{K}$. We use $\mathcal{L}_{\text{mask}}$ and $\mathcal{L}_{\text{class}}$ to supervise our deep learning network, and the specific loss functions are as follows:

$$\begin{aligned} \mathcal{L}_{\text{mask}} &= \lambda_1 \mathcal{L}_{\text{bce}}^1 + \lambda_2 \mathcal{L}_{\text{dice}}^2 \\ &= \sum_{i=1}^{\hat{K}} \left( -\lambda_1 \left[ o_i \log(\hat{o}_i) + (1-o_i)\log(1-\hat{o}_i) \right] + \lambda_2 \left(1 - 2 \times \frac{|o_i \cap \hat{o}_i|}{|o_i| + |\hat{o}_i|}\right) \right) \end{aligned} \tag{6}$$

$$\begin{aligned} \mathcal{L}_{\text{class}} &= \lambda_3 \mathcal{L}_{\text{bce}}^3 \\ &= \sum_{i=1}^{\hat{K}} \left( -\lambda_3 \left[ g_i \log(\hat{g}_i) + (1-g_i)\log(1-\hat{g}_i) \right] \right) \end{aligned} \tag{7}$$

where $\mathcal{L}_{\text{bce}}$ is the binary cross-entropy loss and $\mathcal{L}_{\text{dice}}$ is the dice loss [10]; $\hat{o}_i$ is the predicted mask of trunks and $o_i$ is the ground truth at a grid scale; $\hat{g}_i$ is the predicted semantic labels belongs to the trunk or not and $g_i$ is the ground truth at a grid scale; $\lambda_1, \lambda_2, \lambda_3$ are weighting coefficients for related items. We use $\mathcal{L}_{\text{mask}}$ to supervise trunks mask and $\mathcal{L}_{\text{class}}$ to supervise classification. The overall loss for all auxiliary predictions, together with the loss functions of the different layers, whose serial number is denoted as $l$, is defined as:

$$\mathcal{L} = \sum_{l=1}^{5} (\mathcal{L}_{\text{mask}}^l + \mathcal{L}_{\text{class}}^l) \tag{8}$$

**4) Individual tree segmentation based on the hierarchical k-nearest neighbors algorithm**

Based on the obtained trunk instances, a hierarchical k-nearest neighbors classification strategy based on Euclidean distance is adopted to allocate points to complete single tree segmentation from low to high. Firstly, the point cloud is divided into several layers based on heights, and the first layer contains every recognized trunk instance by our network. Then, the classified points in the first layer are used as the seed points for the k-nearest neighbors algorithm, and a connectivity relationship is established with the points in the

second layer (upper) via Euclidean distance with seed points. Next, the repetitive operation is performed for the point clouds in the third layer based on the classified points as seeds in the second layer. Finally, iterations from the low to high layers are conducted until all points are reassigned. This strategy effectively utilizes the skeletal structure characteristics of trees. Considering the disjoint trunks of each tree in the lower heights with easier affiliation determination, these accurate classified points can provide reliable basis support for identifying the points regarding the intersected tree crowns in the higher layers, thereby achieving an accurate delineation of tree crown boundaries.

## III. EXPERIMENTS

The experimental environment is the Ubuntu 18.04 LTS operating system with an Intel® Core™ i7-12700F processor, 32 GB of memory, and an NVIDIA RTX 3070 GPU. We use the Minkowski Engine to implement sparse convolution [15]. Moreover, we set the weighting coefficients $\lambda_1 = 5$, $\lambda_2 = \lambda_3 = 2$, and the larger value of $\lambda_1$ is beneficial to accomplish the more complicated task of trunk instance segmentation; the k of the hierarchical k-nearest neighbors algorithm is set to 5.

### A. Performance Parameters

To evaluate the performance of trunk and tree segmentation, we use metrics such as precision (Pre), recall (Rec), and F1 score (F1). For the individual tree segmentation results, we additionally compare the achieved overlap accuracies via the Mean Intersection over Union (MIoU) at the point scale, and its specific formula is as follows:

$$MIoU = \frac{1}{N}\sum_{\varepsilon=1}^{N}\frac{\sum_{\gamma=1}^{T^{\varepsilon}} pre_{\gamma}^{\varepsilon} == cls_{\gamma}^{\varepsilon}}{\sum_{\alpha=1}^{N_p^{\varepsilon}} pre_{\alpha}^{\varepsilon} + \sum_{\beta=1}^{N_c^{\varepsilon}} cls_{\beta}^{\varepsilon} - \sum_{\gamma=1}^{T^{\varepsilon}} pre_{\gamma}^{\varepsilon} == cls_{\gamma}^{\varepsilon}} \quad (9)$$

where $N$ is the number of total trees, $N_p^{\varepsilon}, N_c^{\varepsilon}$ and $T^{\varepsilon}$ is denoted the number of predicted points, ground-truth points, and intersections in the $\varepsilon$th tree; $pre_{\alpha}^{\varepsilon}$ denotes the $\alpha$th point belonging to the $\varepsilon$th predicted tree; $cls_{\beta}^{\varepsilon}$ denotes the $\beta$th point of the $\varepsilon$th ground-truth tree; $pre_{\gamma}^{\varepsilon} == cls_{\gamma}^{\varepsilon}$ represents the logical judgment of whether the $\gamma$th predicted point of the $\varepsilon$th tree is equivalent to ground truth.

### B. Different Parameter Experiments

We analyze the segmentation accuracies obtained with different grid sizes during training. Moreover, we adjust the number of QRM contained in the instance segmentation branch; i.e., each QRM represents one layer, and various numbers of QRM layers are included in the branch. Fig. 4 shows that the accuracy increases rapidly when the number of layers increases from 1 to 4; it does so slowly when the accuracy increases further after 4 layers. Moreover, using a grid size of 2 cm obviously improves the resulting accuracy but also increases the computation time required for the run. Therefore, we use 4 QRM layers and a grid size of 2 cm to ensure higher precision during training.

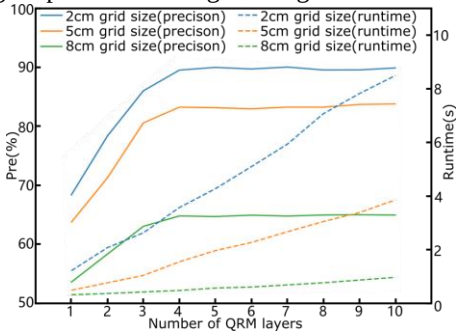


**Fig. 4.** Experiments from different number of QRM layers.

### C. Trunk and Tree Segmentation Results

To verify the feasibility of our method proposed in this paper, we performed trunk instance segmentation on the test datasets and analyzed the results.

Fig. 5 (a)-(b) shows the trunk segmentation results, where the point clouds are divided into three layers by height. The green part represents the higher than 2 m part of the point clouds; the blue color represents the 0-2 m part; and the grey color represents the ground. The colorful point clouds in the 0-2 m part are the result of the trunk segmentation and the (c)-(d) is a zoomed-in display. The results show the robustness and superiority of our method in complex tree scenes.

TABLE I. COMPARISON OF DIFFERENT METHODS IN TEST PLOTS

| Method | I | | | II [13] | | | Runtime (s) |
|---|---|---|---|---|---|---|---|
| | Pre | Rec | F1 | Pre | Rec | F1 | |
| 3DBoNet[16] | 64.8 | 61.3 | 63.0 | 64.7 | 63.0 | 63.8 | 51.33 |
| PointGroup[17] | 75.3 | 71.6 | 73.4 | 74.0 | 73.3 | 73.6 | 4.71 |
| HAIS[18] | 80.7 | 79.8 | 80.2 | 82.9 | 80.3 | 81.6 | 3.62 |
| SoftGroup[19] | 84.7 | 82.8 | 83.7 | 86.5 | 85.7 | 86.1 | 3.83 |
| ISBNet[20] | 88.7 | 87.2 | 88.0 | 87.3 | 88.4 | 87.9 | 3.58 |
| Ours | **89.4** | **88.8** | **89.1** | **87.4** | **89.8** | **88.6** | **3.23** |

Table I compares the six methods in test plots. It shows that the presented approach outperforms the reference approaches in terms of accuracy, quality, and time cost. Especially in terms of accuracy, the proposed approach exhibits significant advantages. The precision, recall and F1 of our method reach above 89.4 %, 88.8 % and 89.1 % in dataset I, respectively. Meanwhile, excellent performance was also achieved in dataset II [13]. Moreover, the operational efficiency is notably improved duo to reducing computational load at the grid scale.

To evaluate the robustness and superiority of our tree segmentation method, we show two detailed examples of visual test results in Fig. 5 (e)-(f). Specifically, the results with different methods are shown in Table II. We can see that our proposed method achieves excellent performance. PDENet [10] achieves better results for simple scenarios with stranded trunk shapes. However, obtaining satisfactory results is often difficult for more complex cases. TCNet [11] first predicts the centroid of the tree point cloud as the basis for the instance tree and then performs individual tree segmentation. Centroid points are reliable signals because incomplete tree point clouds still have accurate centroids. In our approach, we

enhance the semantic and instance segmentation of tree trunks by deep learning models to obtain accurate initial growth locations of individual trees. Thus, this avoids the over-segmentation problem caused by incorrect tree top detection and the under-segmentation problem where trunks are not detected. The hierarchical k-nearest neighbors algorithm makes our method boundary-aware for segmenting single trees. Although our method still has some errors in the segmentation results, it is clear that the numerical comparison illustrates that the proposed framework is superior to the aforementioned methods by obtaining better results. Meanwhile, the precision, recall, F1 score, and MIoU of our method are superior to the second-best method with gaps of approximately 1.18%, 1.70%, 1.45%, and 1.28%.

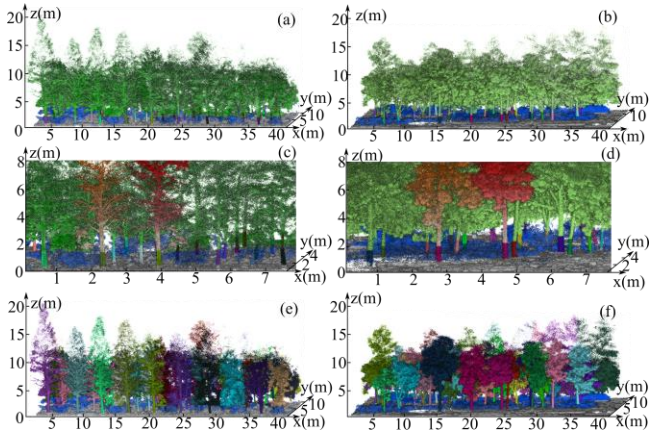


**Fig. 5.** Result in two test plots. (a)-(b) Results of the trunk segmentation. (c)-(d) Zoomed-in display of the trunk segmentation and two selected trees attached gradient colors represent the results of the vertical hierarchical k-nearest neighbors algorithm. (e)-(f) Results of the individual tree segmentation.

TABLE II. COMPARISON OF DIFFERENT METHODS IN INDIVIDUAL TREE SEGMENTATION

| Method | TP | FP | FN | Pre | Rec | F1 | MIoU |
|---|---|---|---|---|---|---|---|
| PDENet [10] | 421 | 65 | 72 | 86.63 | 85.40 | 86.01 | 82.42 |
| TCNet [11] | 438 | 57 | 63 | 88.48 | 87.43 | 87.95 | 89.56 |
| Ours | **451** | **52** | **55** | **89.66** | **89.13** | **89.40** | **90.84** |

## IV. CONCLUSION

Segmenting single trees from point clouds obtained from the MLS systems of complex urban parks is a difficult challenge. To address the related issues, we propose a new framework for segmenting trees in urban scenes of the LiDAR point clouds. The proposed framework includes a dual-branch deep learning network. A semantic segmentation branch for partitioning trunk semantics and an instance segmentation branch for extracting trunk instances in the understory layer at the grid scale. Then, based on the trunk instances, the method employs hierarchical k-nearest neighbors to segment individual tree instances. Ultimately, experimental and analytical evidence substantiates our framework's efficacy in segmenting individual trees from MLS point clouds. In summary, we posit that our contribution offers novel insights into the utilization of deep learning for individual tree segmentation.

## REFERENCES

[1] Y. Bao, M. Gao, D. Luo, and X. D. Zhou, "Urban Parks-A Catalyst for Activities! The Effect of the Perceived Characteristics of the Urban Park Environment on Children's Physical Activity Levels," *Forests*, vol. 14, no. 2, Feb 2023, Art. no. 423.

[2] W. X. Wang, Y. H. Fan, Y. Li, X. M. Li, and S. J. Tang, "An Individual Tree Segmentation Method From Mobile Mapping Point Clouds Based on Improved 3-D Morphological Analysis," *Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 2777-2790, 2023.

[3] Z. Y. Guo, H. Liu, H. Y. Shi, F. Li, X. Y. Guo, and B. H. Cheng, "KD-Tree-Based Euclidean Clustering for Tomographic SAR Point Cloud Extraction and Segmentation," *Ieee Geoscience and Remote Sensing Letters*, vol. 20, 2023, Art. no. 4000205.

[4] L. S. Zhong, L. Cheng, H. Xu, Y. Wu, Y. M. Chen, and M. C. Li, "Segmentation of Individual Trees From TLS and MLS Data," *Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 2, pp. 774-787, Feb 2017.

[5] M. Kukkonen, T. Lähivaara, and P. Packalen, "Combination of Lidar Intensity and Texture Features Enable Accurate Predition of Common Boreal Tree Species With Single Sensor UAS Data," *Ieee Transactions on Geoscience and Remote Sensing*, vol. 62, 2024, Art. no. 4401508.

[6] M. Weinmann, M. Weinmann, C. Mallet, and M. Brédif, "A Classification-Segmentation Framework for the Detection of Individual Trees in Dense MMS Point Cloud Data Acquired in Urban Areas," *Remote Sensing*, vol. 9, no. 3, p. 277, 2017.

[7] Z. K. Yang *et al.*, "Segmenting Individual Trees From Terrestrial LiDAR Data Using Tree Branch Directivity," *Ieee Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 17, pp. 956-969, 2024.

[8] W. Fan, B. S. Yang, Z. Dong, F. X. Liang, J. H. Xiao, and F. S. Li, "Confidence-guided roadside individual tree extraction for ecological benefit estimation," *International Journal of Applied Earth Observation and Geoinformation*, vol. 102, Oct 2021, Art. no. 102368.

[9] X. Ning *et al.*, "Trunk-Constrained and Tree Structure Analysis Method for Individual Tree Extraction from Scanned Outdoor Scenes," *Remote Sensing*, vol. 15, no. 6, p. 1567, 2023.

[10] H. F. Luo, K. Khoshelham, C. C. Chen, and H. X. He, "Individual tree extraction from urban mobile laser scanning point clouds using deep pointwise direction embedding," *Isprs Journal of Photogrammetry and Remote Sensing*, vol. 175, pp. 326-339, 2021 2021.

[11] T. P. Jiang, Y. J. Wang, S. Liu, Q. Y. Zhang, L. Zhao, and J. Sun, "Instance recognition of street trees from urban point clouds using a three-stage neural network," *Isprs Journal of Photogrammetry and Remote Sensing*, vol. 199, pp. 305-334, May 2023.

[12] K. Q. Liu, W. G. Wang, R. Tharmarasa, J. Wang, and Y. Zuo, "Ground Surface Filtering of 3D Point Clouds Based on Hybrid Regression Technique," *Ieee Access*, vol. 7, pp. 23270-23284, 2019.

[13] X. Han, C. Liu, Y. Z. Zhou, K. Tan, Z. Dong, and B. S. Yang, "WHU-Urban3D: An urban scene LiDAR point cloud dataset for semantic instance segmentation," *Isprs Journal of Photogrammetry and Remote Sensing*, vol. 209, pp. 500-513, Mar 2024.

[14] A. Vaswani *et al.*, "Attention is All you Need," in *Neural Information Processing Systems*, 2017.

[15] C. Choy, J. Gwak, S. Savarese, and I. C. Soc, "4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks," in *32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, 2019, pp. 3070-3079, 2019.

[16] B. Yang *et al.*, "Learning Object Bounding Boxes for 3D Instance Segmentation on Point Clouds," in *33rd Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, CANADA, 2019, vol. 32, 2019.

[17] L. Jiang *et al.*, "PointGroup: Dual-Set Point Grouping for 3D Instance Segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Electr Network, 2020, pp. 4866-4875, 2020.

[18] S. Y. Chen, J. M. Fang, Q. Zhang, W. Y. Liu, X. G. Wang, and Ieee, "Hierarchical Aggregation for 3D Instance Segmentation," in *18th IEEE/CVF International Conference on Computer Vision (ICCV)*, Electr Network, 2021, pp. 15447-15456, 2021.

[19] T. Vu, K. Kim, T. M. Luu, T. Nguyen, C. D. Yoo, and S. O. C. Ieee Comp, "SoftGroup for 3D Instance Segmentation on Point Clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, New Orleans, LA, 2022, pp. 2698-2707, 2022.

[20] T. D. Ngo, B. S. Hua, K. Nguyen, and Ieee, "ISBNet: a 3D Point Cloud Instance Segmentation Network with Instance-aware Sampling and Box-aware Dynamic Convolution," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, CANADA, 2023, pp. 13550-13559, 2023.