Urban Forestry & Urban Greening 120 (2026) 129414

Contents lists available at ScienceDirect

Urban Forestry & Urban Greening

journal homepage: www.elsevier.com/locate/ufug

Original article

SATree: Structure-aware tree instance segmentation from 3D LiDAR point clouds

Shenglan Du [a], Jantien Stoter [a], Julian F.P. Kooij [b], Liangliang Nan [a,*]

[a] 3D Geoinformation Group, Faculty of Architecture and the Built Environment, Delft University of Technology, Delft, 2628 BL, The Netherlands
[b] Intelligent Vehicles Group, Faculty of Mechanical Engineering, Delft University of Technology, Delft, 2627 CD, The Netherlands

ARTICLE INFO

Dataset link: https://doi.org/10.6084/m9.figshare.c.6788358.v1, https://zenodo.org/records/8287792.

Keywords:
3D urban forestry
Point clouds
Tree instance segmentation
Structure awareness
Multi-task learning

ABSTRACT

Accurate segmentation and analysis of individual trees from 3D point clouds is a crucial yet challenging task in urbanism and environmental studies. Most existing methods for tree instance segmentation suffer from either under- or over-segmentation errors, mainly due to the complex nature of the environments and the varying tree geometries. In this paper, we propose SATree, a novel structure-aware approach that directly identifies important tree structures, such as crowns and stems, from point clouds, enabling robust tree instance segmentation against tree overlaps and varying tree sizes. Our method leverages a multi-task learning framework that simultaneously performs (i) semantic segmentation to classify a point as *crown*, *stem*, or *other*; (ii) heatmap prediction to assign a heat value to each point based on 2D Gaussian kernels centered at tree stem locations; (iii) offset prediction to estimate point-wise offset vectors pointing to the instance centroid. Key to our approach is the stem localization module, where we fuse the semantic and heatmap predictions to reliably localize tree stems from the network outputs. After that, we utilize a graph-based shortest path algorithm to group individual tree points by integrating the learned offset embeddings. Extensive experiments on two public forestry datasets, TreeML and ForInstance, demonstrate that SATree consistently outperforms state-of-the-art methods in terms of AP, AP_{50}, and AP_{25} scores, reducing significant under- or over-segmentation errors. Our research output supports downstream forestry inventory, 3D tree reconstruction, and fine-grained part segmentation of trees. Our source code of SATree is available at https://github.com/shenglandu/SATree.

1. Introduction

Trees play a critical role in urban ecosystems. They bring considerable ecological and economic benefits to human health by providing photosynthetic activity, maintaining the carbon balance, and regulating the temperature (Hyyppä et al., 2012). Accurate measuring and assessment of trees is a fundamental task for many applications, such as urban planning, forestry management, and environmental simulations. For example, estimating forest biomass and volume at the individual tree level is crucial for accurate carbon storage assessments, which further helps to develop climate change mitigation strategies (Shrestha et al., 2018); Modeling trees within urban green spaces is also important to landscape architects and urban designers, contributing to the planning and sustainability of modern cities. All these applications require precise tree inventory at the instance level.

Traditional inventories of trees and vegetation heavily rely on field surveys, which can be labor-intensive, time-consuming, and costly (Hyyppä et al., 2001). With recent advances in remote sensing technology, 2D satellite imagery and 3D Light Detection and Ranging (LiDAR) data have been used to efficiently characterize forest structure in large areas (Dassot et al., 2011). Specifically, a wide range of image-based Artificial Intelligence (AI) approaches have been proposed to facilitate vegetation segmentation (Arief et al., 2018), identify tree species (Hakula et al., 2023), delineate individual tree crowns (Weinstein et al., 2020; Yun et al., 2021), and analyze tree structures (Reche-Martinez et al., 2004). However, images suffer from low resolution, weather sensitivity, and occlusion issues that hinder accurate data acquisition (Wang et al., 2023). Furthermore, imagery naturally does not capture tree structure details and struggles to achieve fine-grained tree instance segmentation. In contrast, LiDAR point clouds directly capture object surfaces with accurate 3D measurements and rich geometrical details. Given their high spatial resolution and accuracy, LiDAR data have been widely adopted for individual tree segmentation, which further enables researchers to derive key botanical structural parameters such as tree height (Olofsson et al., 2014), Diameter at Breast Height (DBH) (Sun et al., 2022a), as well as tree volume and biomass (Fan et al., 2020).

* Correspondence to: Julianalaan 134, 2628BL, Delft, The Netherlands
E-mail addresses: shenglan.du@tudelft.nl (S. Du), j.e.stoter@tudelft.nl (J. Stoter), j.f.p.kooij@tudelft.nl (J.F.P. Kooij), liangliang.nan@tudelft.nl (L. Nan).

https://doi.org/10.1016/j.ufug.2026.129414
Received 27 November 2025; Received in revised form 28 February 2026; Accepted 20 March 2026
Available online 21 March 2026

Individual tree segmentation from 3D point clouds can be categorized into heuristic- and learning-based approaches. Heuristic-based methods often assume that tree tops are local maxima and thus can be detected by watershed algorithms (Chen et al., 2006). Another assumption is that tree crown points form dense clusters in Euclidean space. Therefore, they can be segmented through various clustering techniques, including mean-shift clustering (Malladi et al., 2024), density-based clustering (J. Wang et al., 2018; Hakula et al., 2023), hierarchical clustering (Lee et al., 2010), and graph shortest path algorithm (Livny et al., 2010; Tao et al., 2015). However, these methods highly demand domain-specific prior knowledge. On the other hand, driven by the success of deep learning in computer vision and point cloud analysis, several deep learning-based methods have been introduced. Early methods typically convert point clouds into discretized models such as Canopy Height Models (CHMs) or Digital Surface Models (DSMs) and apply image processing techniques, *e.g.*, Convolutional Neural Networks (CNNs), to achieve individual tree segmentation (Chang et al., 2022; Hamraz et al., 2019; Wang et al., 2019). The 2D segmentation results are then projected back to the original 3D space to obtain 3D tree instances. To avoid potential information loss during data transformation, more recent approaches (Wang et al., 2023; Z. Luo et al., 2021; H. Luo et al., 2021; Jiang et al., 2023a,b; Henrich et al., 2024) directly process point clouds and perform per-point instance predictions in 3D space. Among them, (Wang et al., 2023) designs a two-branch network that fuses the features of the semantic and instance branches for tree instance segmentation. H. Luo et al. (2021), Jiang et al. (2023a), Henrich et al. (2024) perform joint semantic segmentation and offset prediction (*i.e.*, a directional vector pointing to the tree instance centroid), followed by clustering the points into individual trees.

Despite their impressive performances, many approaches struggle to accurately identify individual trees in complex urban forest scenes. In these environments, heterogeneous canopy structures, varying tree sizes, and dense tree foliage can lead to significant crown overlap or occlusion. As a result, segmentation outputs often exhibit under-segmentation, where multiple close-by trees are segmented into a single instance, or over-segmentation, where a single crown is fragmented into several segments. These errors can propagate and ultimately impact downstream analyses such as tree attribute computation, biomass estimation, and urban ecosystem modeling.

In this paper, we propose SATree, a Structure-Aware Tree instance segmentation approach, to address challenging urban forestry scenes in large scale. Our method is inspired by an intuitive insight that a tree instance naturally contains a stem and a crown. By explicitly detecting the key structures, *e.g.*, stems, we can leverage them as geometric anchors to guide the segmentation of individual trees in challenging urban forests, especially for forestry scenes with dense crown overlaps, branch interference, occlusions, and varying tree shape complexities. We introduce a simple and unified point cloud learning framework, containing a semantic segmentation branch, a Gaussian heat prediction branch, and an instance centroid-oriented offset prediction branch. This unified framework enables automated detection of tree crowns and localization of tree stems in a parallel manner, without requiring any pre-processing or auxiliary learning modules.

Fig. 1 provides an overview of the SATree workflow, where we use one network to jointly perform three tasks. First, the semantic segmentation task classifies each point as *crown*, *stem*, or *other*. Then, the heatmap prediction task predicts a point-wise heat response based on 2D Gaussian kernels centered at tree stem locations. The heat value increases as the point approaches the tree stem or main branches, to support more accurate stem localization. In the last offset prediction task, a 3D offset vector is predicted for each point, directing it towards its corresponding tree instance centroid. Once tree stems are identified, we use a graph-based shortest path approach to isolate individual trees, which considers the proximity of points in the Euclidean space and their offset orientation towards the instance centroids. This strategy helps to precisely delineate tree instance boundaries. Our work provides three major contributions:

- We introduce a multi-task learning framework that jointly segments crowns and stems in urban forestry scenes. Leveraging the complementary predictions of these key structural components greatly improves overall tree instance segmentation.
- We propose a heatmap prediction module, learning high-response representations of major tree structures to effectively guide the tree instance segmentation.
- We develop a direction-aware, graph-based method to integrate the learned offset embeddings for precise delineation of tree boundaries in complex areas.

2. Related work

In this section, we review earlier studies relevant to individual tree detection and tree instance segmentation, ranging from the conventional heuristic-based approaches to the recent deep learning-based approaches.

2.1. Traditional tree instance segmentation

Automatic segmentation of individual trees is challenging due to the irregularity of tree shapes and the inherent complexity of forest ecosystems. Early approaches primarily focused on identifying and extracting treetops from 2D aerial imagery (Dralle and Rudemo, 1996; Wulder et al., 2000), which was later extended to 3D LiDAR data (Chen et al., 2006). Following treetop detection, classical image segmentation techniques, such as watershed segmentation (Beucher, 1979), can be applied to recognize individual trees. To enhance the segmentation of tree morphological shapes, Yun et al. (2021) introduced a dual Gaussian filter in combination with an anisotropic water expansion algorithm for crown boundary segmentation. These methods detect treetops as local maxima from images or CHMs, which may result in large commission errors (Chen et al., 2006).

Another line of approaches adopts grouping-based strategies to segment individual trees from point clouds, based on the observation that tree canopy points naturally form dense clusters in the 3D space. The clustering algorithms used for grouping the points are k-means (Gupta et al., 2010), hierarchical clustering (Lee et al., 2010), and mean-shift (Malladi et al., 2024). Studies of Ayrey et al. (2017), J. Wang et al. (2018), Hakula et al. (2023) propose a layer-wise stacking strategy to mitigate clustering errors in dense forest areas. These methods slice the forest canopy into layers, cluster points per layer, and aggregate the layer-wise clusters into tree instances. Several works also leverage graph structure for tree instance segmentation. Works of Livny et al. (2010), Tao et al. (2015) applied the graph shortest path algorithm to group individual trees. Heinzel and Huber (2018) constructed a similarity graph over the points and segmented the trees using a Markov Random Field framework. Wang et al. (2021) developed a hybrid approach combining the Delaunay graph and k-Nearest Neighbor (kNN) graph, where each node repeatedly walks to its lowest neighbor to locate its tree source. Burmeister et al. (2024) introduced a computationally efficient marker-controlled Watershed algorithm with 3D region growing to segment individual trees in a coarse-to-fine manner.

A major limitation of heuristics-based approaches is that they heavily rely on domain-specific knowledge as priors, making them difficult to generalize to a broad range of forestry scenes. Finding the optimal solution is often non-trivial for a specific urban scene or tree type.

2.2. Deep learning on 3D instance segmentation

Deep learning has revolutionized point cloud processing and 3D computer vision in the last decade. The seminal PointNet (Qi et al., 2017a) and PointNet++ (Qi et al., 2017b) have set the learning trend from raw point clouds without transforming points into intermediate

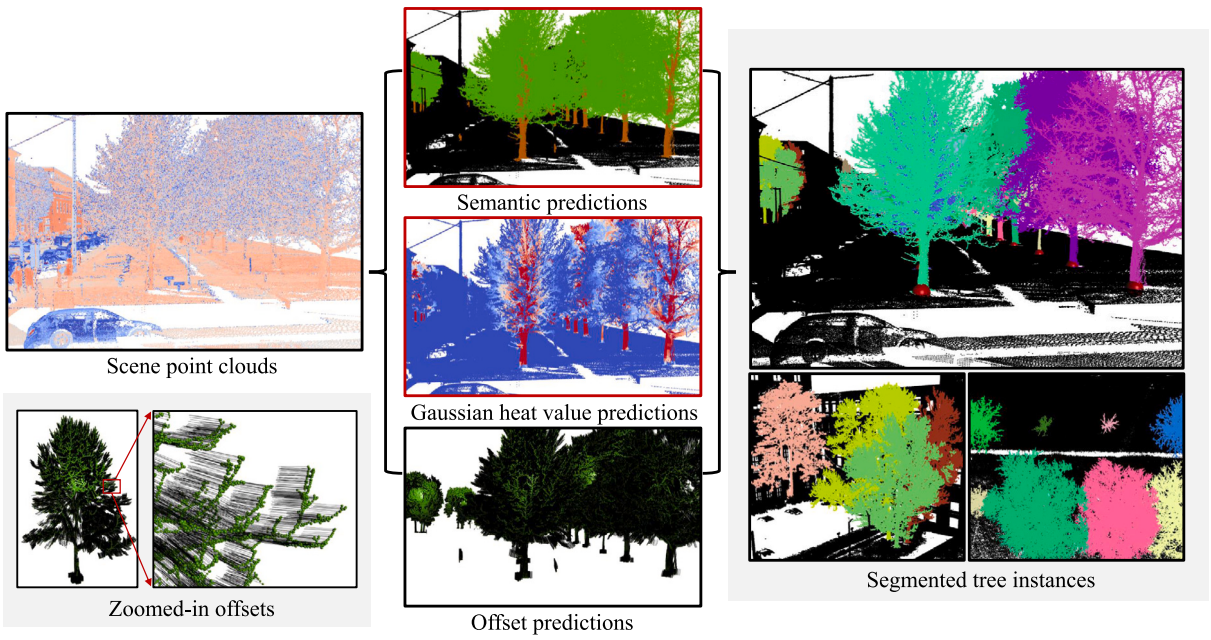


Fig. 1. SATree jointly perform three tasks: (i) semantic segmentation to classify if a point is *crown* (green), *stem* (brown), or *other* (black); (ii) heatmap prediction to assign a heat value to each point, ranging from 0 (blue) to 1 (red). High values indicate tree stems or main branches; (iii) offset prediction to predict a vector pointing to the corresponding tree instance centroid. Tree points and offset vectors are visualized in green and black (bottom left). By combining these predictions, SATree achieves accurate tree instance segmentation. Unlike existing methods (Jiang et al., 2023b; Henrich et al., 2024) that do not explicitly model tree structures, SATree jointly classifies crown and stem points and predicts a heatmap for robust stem localization. Specifically, the proposed semantic and heatmap branches, which are highlighted in red, have demonstrated their effectiveness across trees of varying sizes and shapes in challenging scenes (bottom right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

formats such as image planes or 3D voxels. Since then, many point-wise feature learning networks have been introduced, focusing on aggregating geometrical information in local neighborhoods. Based on the design of the local feature aggregation function, these methods can be roughly categorized into Multi-layer Perceptron (MLP) based (Qian et al., 2022; Lin et al., 2023), Convolution based (Thomas et al., 2019; Xu et al., 2021), Transformer and Attention based (Zhao et al., 2021; Lai et al., 2022).

Driven by the success of point-wise learning networks, several studies have been proposed to address the 3D instance segmentation task on point clouds, which aims to recognize each point at the object instance level. One of the first studies is SGPN (W. Wang et al., 2018), which learns a 3D instance segmentation representation by forming the feature similarity matrix. This method can be computationally expensive. PointGroup (Jiang et al., 2020) introduces a framework that learns to group points into instance clusters based on semantic and offset predictions. This idea was later extended to HAIS (S. Chen et al., 2021) and SoftGroup (Vu et al., 2022) for its simplicity and effectiveness. HAIS refines (Jiang et al., 2020) by adding a hierarchical aggregation scheme. SoftGroup performs bottom-up soft grouping, allowing each point to be assigned to multiple instances, and then refines the segmentation results top-down. Very recently, PointGroup and SoftGroup have been adopted as backbone architectures for 3D tree instance segmentation (Wielgosz et al., 2024; Henrich et al., 2024).

The design of SATree architecture is largely inspired by the emergence of recent deep learning-based methods on general 3D instance segmentation tasks. However, SATree is specifically designed for tree instance segmentation. It focuses on joint semantic segmentation of tree points and detection of particular tree parts as structure priors to guide the instance segmentation, which is lacking in existing studies to our best knowledge.

2.3. Deep learning on tree instance segmentation

Enabled by the broad applications of deep learning in computer vision and point cloud analysis, numerous learning-based approaches have been developed to address tree instance segmentation.

A number of studies have directly applied image-based object detection networks to identify 2D urban tree instances from captured RGB imagery. Among them, DeepForest (Weinstein et al., 2019, 2020) adopts the RetinaNet (Lin et al., 2017) detector to produce tree bounding box predictions from aerial images. Follow-up studies have adapted Region-based Convolutional Neural Networks (RCNNs), *e.g.*, Faster RCNN and C-Mask RCNN, for individual tree detection and counting (Sun et al., 2022b; Osco et al., 2020). Ammar et al. (2021) compared several CNN-based object detection approaches for palm tree counting in large farm areas, concluding that Yolov4 (Bochkovskiy et al., 2020) and EfficientDet-D5 (Tan et al., 2020) provide the best trade-off between accuracy and inference speed.

Besides, many research efforts have also been dedicated to 3D tree instance segmentation from LiDAR point clouds. Among them, discretization-based methods first discretize the tree points into grid-based models (*e.g.*, DSMs and CHMs), then adopt a 2D object detection network to locate tree bounding boxes. Chang et al. (2022) projected tree points onto the ground plane and utilized Yolov3 (Redmon, 2018) for tree instance segmentation. Xi and Hopkinson (2021) transformed tree points into bird's-eye-view images and employed CenterNet (Duan et al., 2019) for individual tree detection. Alternatively, point-based methods directly perform instance segmentation on 3D tree points, avoiding the potential information loss during data transformation. Wang et al. (2023) proposed a two-branch network that fuses features from the semantic branch and the instance branch to segment tree instances. X. Chen et al. (2021) used PointNet (Qi et al., 2017a) to classify tree points and obtained individual tree crown boundaries by analyzing height gradients. H. Luo et al. (2021) first performed semantic segmentation of trees, then incorporated an additional network

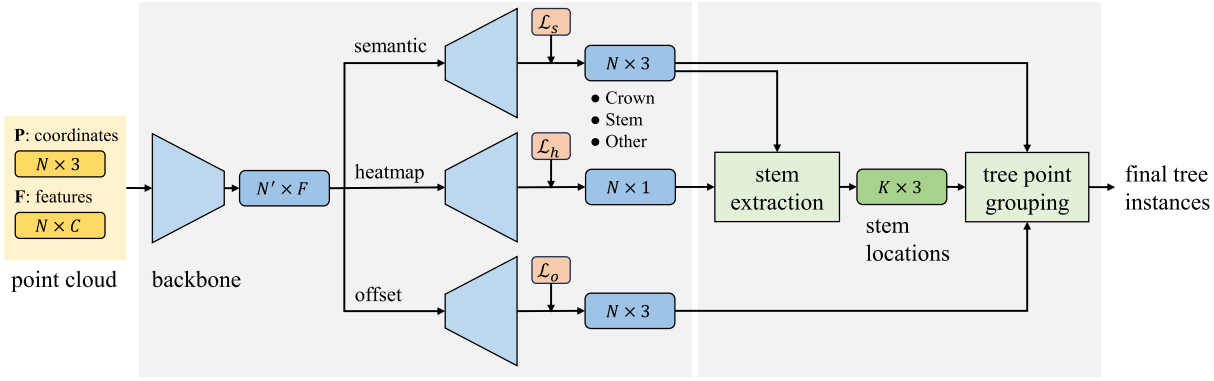


Fig. 2. Framework of SATree. The input to our method is a point cloud containing coordinates and additional features. We use intensity as the input feature across all the experiments, *i.e.*, $C = 1$. SATree has one shared point feature encoder and three decoders that jointly perform semantic segmentation, Gaussian heatmap prediction, and offset vector prediction. N is the number of input points, $N' \times F$ denotes sparse point features in the latent space, and $N' < N$. We use three distinct losses $\mathcal{L}_s$, $\mathcal{L}_h$, and $\mathcal{L}_o$ to supervise the corresponding tasks. Then, we combine the semantic outputs with the heatmap outputs to accurately localize tree stems. Lastly, the tree points are grouped based on the detected tree stem locations to generate the 3D tree instances.

to predict point-wise offset vectors directed towards object centroids, allowing points to aggregate into distinct tree instances. This strategy was later extended to the study of Jiang et al. (2023b), where tree centroids are mined from the learned offset embeddings to enhance the tree instance segmentation accuracy. Yang et al. (2024) adopted the Point Transformer (Zhao et al., 2021) for semantic segmentation of trees, then utilized morphological opening and closing techniques to achieve single tree segmentation. Recently, Segmentanytree (Wielgosz et al., 2024) and TreeLearn (Henrich et al., 2024) adopted state-of-the-art 3D instance segmentation networks, *e.g.*, PointGroup and SoftGroup, for individual tree segmentation.

To further handle challenging forestry areas with overlapping crowns and interfering branches, Ning et al. (2025) incorporated a boundary-ware module to analyze boundary points within adjacent trees, which enhances the individual tree boundary delineation. Ma et al. (2025) introduced a contrastive learning strategy, which improves the feature discriminability of tree instance representations. Additionally, several studies (Wang et al., 2019; Pu et al., 2023) focused on detecting tree stems to separate single trees, relying on transforming stem points to CHMs and detecting them using 2D image processing techniques.

Similar to some previous studies, SATree performs semantic segmentation and offset prediction. Nevertheless, SATree explicitly detects main tree parts such as stems and crowns. Using these structures as priors, we can precisely localize, identify, and delineate individual trees even in challenging urban areas, maintaining the robustness against tree overlaps and varying tree sizes.

3. Method

Our input is a point cloud $P \in R^{N\times(3+C)}$ captured from an urban scene containing both trees and non-tree objects, where N denotes the number of points and C denotes the input point feature dimension. Theoretically, any useful geometrical or spectral attributes can be used as input point features. In this study, we use the point intensity value as the feature for its simplicity and wide applicability, *i.e.*, $C = 1$. Taking P, we use a point-based deep learning backbone to encode point features, followed by three decoding branches:

- a semantic segmentation branch for segmenting points into categories of *crown*, *stem*, and *other*;
- a Gaussian heatmap prediction branch for assigning a heat value to each point. We generate a 2D Gaussian-based heatmap on the x–y plane with peaks at the locations of individual tree stems. As shown in Fig. 3, points near tree stems or main branches are predicted with high heat values, which helps to localize tree roots precisely;
- an offset prediction branch for predicting a point-wise offset vector directing towards tree instance centroids.

We can identify high-fidelity tree stem locations using the outputs from semantic segmentation and heatmap predictions. Subsequently, combining the outputs from all three branches, we group tree points based on the extracted stem locations to obtain the final tree instances. The overall framework of SATree is illustrated in Fig. 2.

3.1. Network architecture

Our network has one shared feature encoder and three separate decoding branches. We use PointMetaBase (Lin et al., 2023) as our backbone for the feature encoder to obtain high-level point features. In theory, any point-based learning networks (*e.g.*, PointNet++ (Qi et al., 2017b), KP-Conv (Thomas et al., 2019), Point transformer (Zhao et al., 2021), Stratified transformer (Lai et al., 2022)) can be used. In this work, we choose PointMetaBase since it achieves a good trade-off between performance accuracy and computational efficiency. Following the shared feature encoder, the network consists of three decoding branches: the semantic segmentation branch, the heatmap prediction branch, and the offset prediction branch. Each branch performs a distinct task, which is detailed as follows:

Semantic segmentation branch. This branch outputs a semantic logit map $S \in R^{N\times K}$, where K is the number of semantic categories. In this study, the categories include *crown*, *stem*, and *other*, and thus $K = 3$. Tree crowns and stems are treated as distinct classes, while all the rest points in the scene (*e.g.*, building, road, lamppost, pedestrian) are categorized as *other*, given our focus on tree objects. The high class imbalance poses challenges for standard supervision. For the urban environment, *other* (*e.g.*, grounds, buildings, roads) accounts for the majority of the dataset, while *stem* only accounts for a very small portion. Therefore, we use the weighted Cross Entropy loss to supervise this branch, where we assign a significantly higher weight to the class *stem* and a lower weight to *other*.

$$\mathcal{L}_s = -\sum_{i=1}^{N} w_k \log p_i^k, \tag{1}$$

where N is the total number of points, k is the Ground Truth (GT) semantic label of the ith point, p_i^k is the predicted probability of the ith point belonging to its GT category that can be obtained from the network softmax layer, and w_k is the weight of the class k.

Heatmap prediction branch. To better localize tree stems, we produce a 2D Gaussian-based heatmap on the x–y plane with peaks at the locations of individual tree stems. Then, we use this branch

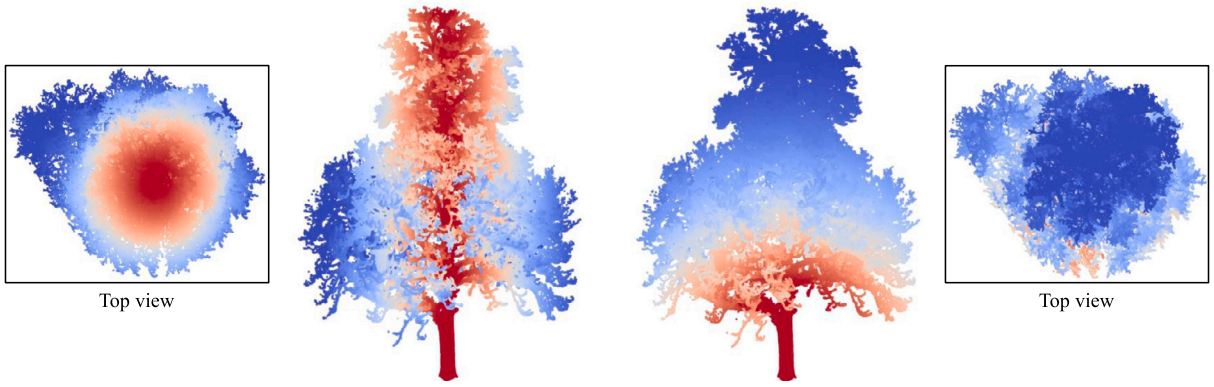


Fig. 3. Visualization comparison of 2D Gaussian representation (left) and 3D Gaussian representation (right) of the same tree instance. The Gaussian heat value ranges from 0 to 1, from blue to red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to predict a Gaussian heatmap $H \in R^{N\times 1}$. Our design principle is to ensure that points associated with tree stems or main branch structures are assigned high heat responses, while minor twigs and background points have low heat values. In theory, it is also possible to design a 3D Gaussian-based heatmap prediction task. However, as shown in Fig. 3, the proposed 2D Gaussian representation better highlights the stem and main branch structures than its 3D counterpart. This is because tree stem bases are vertically elongated but remain spatially consistent in the horizontal x–y plane across height. A 3D Gaussian formulation may introduce unnecessary degrees of freedom that are weakly constrained by data, potentially degrading localization robustness. Contrarily, a 2D representation can effectively suppress vertical sparsity, branching noise, and dense canopy interference. Furthermore, the 2D Gaussian heatmap representation naturally exhibits a local maximum for each tree instance, which directly supports subsequent tree stem localization (Section 3.2).

The GT heatmap $\hat{H}$ is created by retrieving each tree instance and placing a 2D Gaussian kernel centered at its respective tree stem location, which is defined as the geometrical center of the tree stem points. The GT heat value of the ith point is computed as follows:

$$\hat{h}_i = e^{-\alpha d_i / r}, \tag{2}$$

where d_i is the distance from the ith point to the stem centroid of its corresponding tree instance projected on the 2D x–y plane. r is the Gaussian kernel radius, defined as the maximum point-to-stem distance projected on the 2D x–y plane from the current tree instance. α is the hyperparameter that scales the Gaussian distribution. The GT Gaussian heatmap can be directly derived from the raw dataset and only needs to be generated for the training dataset. Then, during inference, the network learns to predict a point-wise Gaussian heat value for the unseen test dataset. We use the Mean Squared Error loss to supervise the heatmap branch:

$$\mathcal{L}_{\mathrm{h}} = \sum_{i=1}^{N} |h_i - \hat{h}_i|^2, \tag{3}$$

where h_i is the predicted heat value of the ith point and $\hat{h}_i$ is the corresponding GT value.

Offset prediction branch. Following the standard practice in previous studies (H. Luo et al., 2021; Jiang et al., 2023b,a; Henrich et al., 2024; Wielgosz et al., 2024), we use this branch to output an offset map $O \in R^{N\times 3}$, where for each point, we predict a 3D offset vector pointing to its tree instance centroid. The predicted offset vectors are further used in the tree point grouping step to enhance the segmentation of tree instances. We use Mean Squared Error loss to supervise this task, which is formulated as follows:

$$\mathcal{L}_{\mathrm{o}} = \sum_{i=1}^{N} \|\mathbf{d_i} - \hat{\mathbf{d}}_\mathbf{i}\|_2^2, \tag{4}$$

where $\mathbf{d_i} \in R^3$ is the predicted offset direction vector. $\hat{\mathbf{d}}_\mathbf{i}$ is the GT offset vector of the ith point. During training, the background (*i.e.*, *other*) points are masked out, whereas only the tree points (*i.e.*, *crown* and *stem*) contribute to the supervision.

Network supervision. The network is jointly supervised by the three loss terms defined for the three separate branches. The total loss is given as follows:

$$\mathcal{L} = \mathcal{L}_{\mathrm{s}} + \lambda_1 \mathcal{L}_{\mathrm{h}} + \lambda_2 \mathcal{L}_{\mathrm{o}}, \tag{5}$$

where λ_1 and λ_2 denote the hyperparameters to balance the corresponding losses.

3.2. Stem localization

Having obtained the network outputs, we first select the points predicted as *stem* and perform a density-based clustering on the x–y plane over the stem points to initially localize individual tree roots. This is inspired by the observation that stem points typically form dense clusters at tree roots.

In practice, not all identified stem clusters correspond to actual tree stems due to noise and errors in the previous semantic predictions. For example, objects such as lampposts are often misclassified as tree stems. To mitigate such errors, we perform a sequence of fidelity checks over the clustered stem candidates. Fig. 4 depicts the stem localization process, with the steps and criteria further detailed in Algorithm 1. Three criteria are designed based on the geometrical properties of tree roots and heat value distributions, which can be formulated as follows:

(i) *Proximity to the ground*. We measure if the lowest point in a tree stem candidate is close enough to the ground using a height threshold ϵ_z (Algorithm 1 Line 9). This ensures that detected stems are grounded appropriately within the scene.

(ii) *High heat value*. As illustrated in Fig. 3, true stem points are likely to be predicted with higher heat values. Therefore, we use an empirical heat threshold ϵ_h to filter out the stem cluster candidates with insufficient heat values (Algorithm 1 Line 10).

(iii) *Local maxima in the heat distribution*. In certain cases, such as urban scenes with small trees, true stem points may exhibit low predicted heat values as the heatmap prediction is implemented as a regression task that outputs continuous predictions. Simple thresholding using ϵ_h will likely ignore such small trees with low heat values. To enhance the robustness of stem detection, we apply a two-ring cylindrical neighborhood, with the inner-ring neighborhood a radius size of ϵ_r, to assess whether a stem cluster represents a local maximum in the heat value distribution. Stem cluster candidates that form such local maxima are also classified as true tree stems, which serves as a refinement for the second criterion. The local maxima-based stem filtering is detailed in Algorithm 1, Lines 14–16.

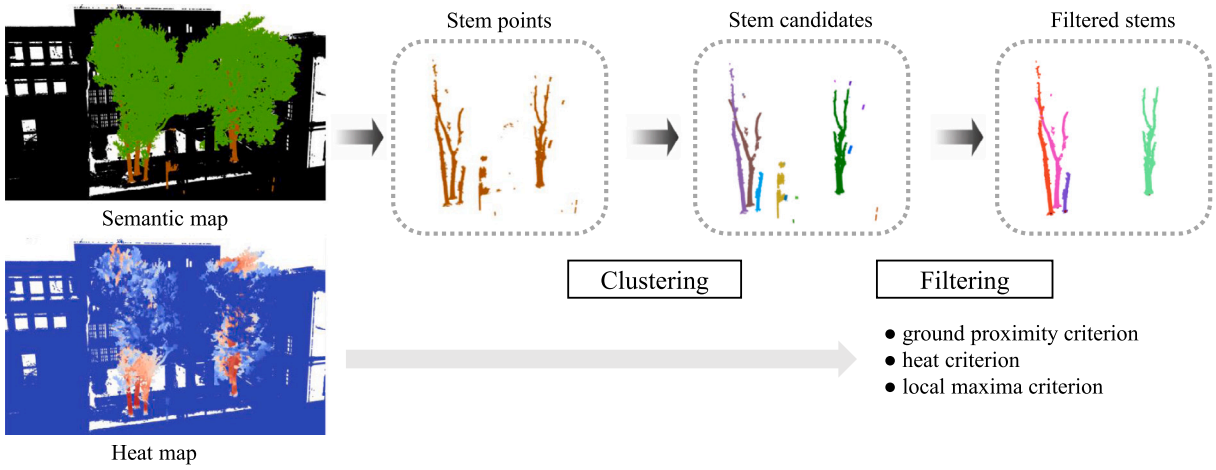


Fig. 4. Stem localization from the predicted scene semantic map and heat map. In the semantic map, crown points, stem points, and background points are visualized in green, brown, and black, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 1: Tree stem localization from the network predictions

Input: input point coordinates P, predicted semantic map S, and heatmap H
Output: stem clusters $C = \{C_1, C_2, C_3, ...\}$ with root locations $R = \{\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, ...\}$
Initialization: $C \leftarrow \emptyset$, $R \leftarrow \emptyset$, user-defined thresholds $\epsilon_h, \epsilon_z, \epsilon_r$
Extract points with semantic predictions of *stem* and *crown*,
  $P_s \leftarrow \{\mathbf{p_i} \in P | s_i = stem\}$;
  $P_c \leftarrow \{\mathbf{p_i} \in P | s_i = crown\}$
Apply density-based clustering on P_s on the x-y plane to obtain stem cluster candidate set $C' = \{C_1, C_2, C_3, ...\}$
for $C_i \in C'$ **do**
  Collect the stem cluster coordinates and heatmap predictions,
    $P_i \leftarrow \{\mathbf{p_j} | \mathbf{p_j} \in P_s \cap C_i\}$;
    $H_i \leftarrow \{h_j | \mathbf{p_j} \in P_s \cap C_i\}$
  $z_{min} \leftarrow min(P_i(z)) - z_{ground}$
  $h \leftarrow avg(H_i)$
  $\mathbf{r_i} \leftarrow avg(P_i)$
  if $z_{min} < \epsilon_z$ **then**
    if $h > \epsilon_h$ **then**
      $C \leftarrow C + \{C_i\}$; $R \leftarrow R + \{\mathbf{r_i}\}$
    end
    else
      $N_1 \leftarrow \{\mathbf{p_j} | \mathbf{p_j} \in P_i \wedge dist(\mathbf{p_j}, \mathbf{r_i}) \leq \epsilon_r\}$; // $dist(\cdot,\cdot)$ computes the 2D distance of points on x-y plane
      $N_2 \leftarrow \{\mathbf{p_j} | \mathbf{p_j} \in P_i \wedge dist(\mathbf{p_j}, \mathbf{r_i}) \leq 3\epsilon_r\}$;
      $h_1 \leftarrow max\{h_j | \mathbf{p_j} \in N_1\}$; $h_2 \leftarrow max\{h_j | \mathbf{p_j} \in N_2\}$
      if $h_1 \geq h_2$ **then**
        $C \leftarrow C + \{C_i\}$; $R \leftarrow R + \{\mathbf{r_i}\}$
      end
    end
  end
end

Our stem localization requires manually tuned hyperparameters, such as the heat value threshold ϵ_h and the radius size ϵ_r. We detail the choice of these hyperparameter values in Section 3.4, and further present the analysis of hyperparameter sensitivity in Section 5.2.

3.3. Tree point grouping

Following the localization of tree stems, we perform individual tree isolation using a graph-based approach. Compared to spatial clustering methods such as DBSCAN, the graph-based approach is less sensitive to data sparsity and outliers, thus allowing more robust tree instance segmentation across heterogeneous point-density distributions and varying tree geometric shapes. We construct a Delaunay triangulation graph $G = (V, E)$ over the input tree points (*i.e.*, points predicted to be either *stem* or *crown*). The detected tree root locations, $R = \{\mathbf{r_1}, \mathbf{r_2}, \mathbf{r_3}, ...\}$ obtained from Algorithm 1, are manually added to G as additional vertices. Then, we apply Dijkstra's shortest path algorithm to build the Minimum Spanning Tree (MST), determining the tree source for each vertex within the graph. We add pseudo edges between root vertices in R with zero edge weights, which guarantees that every tree vertex can be sourced back to a root vertex in the resulting MST.

Works of Livny et al. (2010), Tao et al. (2015) also utilize the shortest path approach to group individual tree points based on the tree roohe edges of G by the 3D Euclidean distances between vertex pairs. While straightforward, this edge-weighting strategy fails to account

Table 1
Details of network supervision and segmentation hyperparameters.

	Eq. (1)			Eq. (2)	Eq. (5)		Eq. (6)	Algorithm 1		
	w_{crown}	w_{stem}	w_{other}	α	λ_1	λ_2	β	ϵ_z	ϵ_h	ϵ_r
TreeML	2.0	15.0	1.0	10.0	10.0	0.05	3.5	2.0	0.5	0.15
ForInstance	2.0	6.0	3.0	10.0	10.0	0.2	2.5	2.0	0.9	0.1

for size variations among different tree instances in complex forest scenes. For example, branches or twigs from a large tree may be wrongly assigned to a neighboring smaller tree if their shortest paths to the neighboring tree's root are shorter. This often leads to inaccurate boundary delineation between tree instances.

To overcome their limitations, we propose shifting the vertex coordinates using the predicted offset embeddings (Section 3.1). Edges in G are then weighted by the 3D Euclidean distances between the shifted vertex pairs. Given the 3D coordinate of the original vertex $\mathbf{v_i}$, we shift it to a new coordinate $\mathbf{v_i}'$ by the following formula:

$$\mathbf{v_i}' = \mathbf{v_i} + \beta(1 - h_i)\frac{\mathbf{d_i}}{\|\mathbf{d_i}\|_2}, \tag{6}$$

where h_i and $\mathbf{d_i}$ are the predicted heat value and offset vector of the ith point in the graph. β is a user-defined hyperparameter. The shifting mechanism enables each tree point to move incrementally towards its corresponding instance centroid. In Eq. (6), the step direction is determined by the predicted offset $\mathbf{d}$, and the step magnitude is modulated by the predicted heat value h. Thus, points near tree stems or main branches will hardly shift, whereas points near instance boundaries undergo more significant shifts towards the instance centroids.

Then, the edge weight e_{ij} between the ith vertex and the jth vertex of the graph G is computed as follows:

$$e_{ij} = \|\mathbf{v_i}' - \mathbf{v_j}'\|_2. \tag{7}$$

We apply Dijkstra's shortest path algorithm to separate individual trees based on the edge weights obtained from Eq. (7), enhancing the robustness of tree segmentation against variations in tree size and shape.

3.4. Test dataset and implementation details

Since our method is primarily designed to segment urban trees from scene-level urban point clouds, we evaluate the effectiveness of the proposed SATree on TreeML (Yazdi et al., 2024), a large-scale labeled urban forest point cloud dataset that can be publicly accessed online. TreeML consists of 40 urban street scenes in Munich captured by Mobile Laser Scanner (MLS) and contains 3755 trees representing a wide range of sizes and species. Each tree is measured by the Quantitative Structure Modeling (QSM) approach, enabling us to extract GT tree stem points. We use 30 scenes for training, five scenes for validation, and five scenes for testing. The test scenes are selected based on their variability in tree instance counts and forest structural complexities.

Besides, we also assess the generalizability of our approach to natural forest scenes using the ForInstance (Puliti et al., 2023) dataset. ForInstance is an Airborne Laser Scanning (ALS) dataset that captures dense forest areas and comprises five collections from diverse global regions (*i.e.*, Norway, the Czech Republic, Austria, New Zealand, and Australia) representing varying forest types. It contains both instance-level annotations and part-level annotations, such as stem and branch. From the publicly available 32 forestry scenes, we use 21 scenes for training and 11 for testing. Our train-test split follows the official benchmark, ensuring that the test set covers all five distinct forest collections.

Given the different geometric properties of the two tree datasets, we tune the hyperparameters separately for each dataset. Table 1 summarizes all hyperparameter values used in SATree, with the top row corresponding to TreeML and the bottom row to ForInstance. In Eq. (1), class-specific weights w are assigned to the semantic categories *crown*, *stem*, and *other* to mitigate class imbalance in the semantic segmentation task. We approximately set these weights as inversely proportional to the number of samples in each class for the corresponding dataset. In Eq. (2), α is set to a large value, *e.g.*, 10.0, for both datasets to encourage a sharply peaked Gaussian heatmap distribution near tree stems and main branch structures. In Eq. (5), we set λ_1 to 10.0 to emphasize the heatmap prediction task during network training. λ_2 is assigned a larger value for ForInstance than for TreeML. Our choice is motivated by the fact that ForInstance mainly comprises coniferous trees, which exhibit smaller offset vector magnitudes than the broadleaf trees in TreeML. Thus, we use a larger λ_2 to increase the importance of the offset prediction task on the ForInstance dataset. Besides, hyperparameters β, ϵ_h, and ϵ_r are empirically selected for the two datasets considering their differences in tree species and geometry. In Section 5.2, we provide a comprehensive analysis of the effects of varying hyperparameters on tree segmentation performance.

For the network backbone, we choose PointMetaBase (Lin et al., 2023), an MLP-based architecture for point feature encoding. It represents a state-of-the-art approach in point cloud learning that provides a good trade-off between segmentation accuracy and computational efficiency. Since both datasets contain massive points that are challenging for the network to directly process, we crop the scanned scenes into small patches and use these as the input to the network. In PointMetaBase, a voxel subsampling strategy is adopted to further reduce the number of points. We set the voxel grid size as 20 cm, following the common practice for processing large-scale urban scenes (Hu et al., 2021). Point intensity is used as the input feature and is normalized to the range $[-1, 1]$. Besides, several data augmentation techniques are applied, such as adding jitter noises to point coordinates, randomly rotating points, and randomly dropping point intensities. During training, PointMetaBase samples randomly cropped points into batches, with each batch containing 24,000 points. The batch size is set to 16. The cosine learning rate scheduler is used with an initial rate of 0.01, and the AdamW optimizer is adopted with a weight decay of 0.0001. We follow this original training configuration of PointMetaBase and train the network on 50 epochs for both datasets. The training experiments are conducted on a single NVIDIA GeForce RTX 2080 Ti. Approximately, the training requires 30 h for the large-scale urban TreeML dataset and 5 h for the nature ForInstance dataset. The subsequent stem localization and tree point grouping are implemented in C++. For more implementation details, please refer to our open-source release of SATree.

4. Results

We assess the performance of the proposed SATree on TreeML and ForInstance datasets, in comparison with both a traditional heuristic-based approach and a recent deep learning–based method. In the following Sections 4.2 and 4.3, we present both the quantitative results and the qualitative results, respectively.

4.1. Comparison and evaluation

For experimental comparison, we select two state-of-the-art methods, TreeSeparation (J. Wang et al., 2018) and TreeLearn (Henrich et al., 2024). They are representative of heuristic-based and learning-based approaches, respectively. Moreover, both methods are open-source, which enables fair and reproducible comparisons.

Table 2
Tree instance segmentation results on TreeML (Yazdi et al., 2024) with AP, AP_{50}, and AP_{25}. SATree achieves the highest scores (in bold), outperforming the other two methods on all scenes by a large margin.

Scene street name	Metric	TreeSeparation (J. Wang et al., 2018)	TreeLearn (Henrich et al., 2024)	SATree
2023-01-09_tum_campus	AP	0.697	0.685	**0.935**
	AP_{50}	0.815	0.800	**0.969**
	AP_{25}	0.846	0.845	**0.985**
2023-01-10_47	AP	0.770	0.726	**0.869**
	AP_{50}	0.821	0.786	**0.893**
	AP_{25}	0.927	0.784	**0.964**
2023-01-12_57	AP	0.905	0.845	**0.972**
	AP_{50}	0.964	0.929	**1.000**
	AP_{25}	0.964	0.929	**1.000**
2023-01-13_70	AP	0.810	0.684	**0.918**
	AP_{50}	0.924	0.771	**0.983**
	AP_{25}	0.932	0.780	**0.983**
2023-01-16_44	AP	0.665	0.851	**0.981**
	AP_{50}	0.875	0.918	**0.997**
	AP_{25}	0.944	0.936	**1.000**

TreeSeparation is a heuristic-based approach that takes the pure tree points as input. Therefore, we use the semantic predictions of our network as its input. The core idea of TreeSeparation is to perform layer-wise clustering for ALS tree point cloud instance segmentation, where the clustering radius and the minimal number of points are the key hyperparameters. We re-tune these hyperparameters for the datasets used in our experiments for the comparison. Specifically, for the TreeML dataset, we set the radius to 1.0 m and the number of points to 500. For the ForInstance dataset, we set these values to 0.5 m and 200, correspondingly.

On the other hand, TreeLearn represents the state-of-the-art in deep learning-based approaches. It takes the scene point clouds as input and directly generates the tree instance predictions. TreeLearn adopts SoftGroup (Vu et al., 2022) as its backbone, with the training hyperparameters optimized for forestry scene segmentation. Therefore, we use their default training configuration. TreeLearn generates instance predictions on voxelized point clouds. We further project their voxel-level predictions back to the original point clouds. To ensure a fair comparison, we use the same input feature (*i.e.*, point intensity), voxel resolution, set the same train-test split for both TreeLearn and our method.

For quantitative evaluation, we adopt the widely used Average Precision (AP) metric to assess the tree instance segmentation performance. AP is calculated as the area under the Precision-Recall curve and is a robust indicator of model performance in instance segmentation tasks. Following the common practice, we report AP scores using Intersection over Union (IoU) thresholds of 25% and 50%, denoted as AP_{25} and AP_{50}. We also report the overall AP score averaged across IoU thresholds ranging from 50% to 95% with a step length of 5%.

4.2. Results of treeml

We perform tree instance segmentation of the proposed SATree on the TreeML dataset in comparison with TreeSeparation and TreeLearn. Table 2 reports the quantitative results. It has demonstrated that SATree achieves the highest scores across all five test scenes as evaluated by the AP, AP_{50}, and AP_{25}, surpassing the comparison methods by a large margin. In particular, SATree attains an AP score exceeding 0.9 in four out of the five scenes. This shows that SATree delivers promising instance segmentation of trees in most urban scenes, highlighting that our approach is consistently practical for the tree instance segmentation task in urban environments.

Fig. 5 visually compares the segmentation results achieved by TreeSeparation, TreeLearn, and SATree. TreeML captures complex urban street-level scenes exhibiting massive trees with varying sizes and shapes, which pose significant challenges for existing methods, resulting in under-segmentation or over-segmentation. For example, in Fig. 5 row 1, three trees with closely intertwined branches and twigs are erroneously segmented as a single tree by TreeSeparation and TreeLearn. In contrast, SATree successfully identifies and segments the three individual trees. Meanwhile, over-segmentation, where a single tree is divided into multiple smaller segments, is also common due to the large volume of tree canopies, which can be observed in rows 2, 4, and 5 of Fig. 5. Unlike TreeSeparation and TreeLearn, SATree mitigates most over-segmentation errors by leveraging its robust stem localization strategy.

We also observe that SATree can detect trees of petite sizes, which are often overlooked by the comparative method TreeLearn (see Fig. 5 row 3). In TreeLearn, trees smaller than 10 m are likely to be misclassified as background points due to the sparsity of small tree representations in the training dataset. However, despite using the same training data, SATree can still effectively identify small trees, due to its ability to accurately detect the roots of small trees. Moreover, our approach can be directly integrated into TreeSeparation. Using our semantic predictions as input, TreeSeparation also successfully identifies small trees.

4.3. Results of ForInstance

SATree is primarily designed to address the task of urban tree instance segmentation. To further assess its applicability to natural forestry environments, we also evaluate it on the ForInstance (Puliti et al., 2023) dataset. Table 3 reports our performance scores compared against TreeSeparation (J. Wang et al., 2018) and TreeLearn (Henrich et al., 2024). Overall, SATree outperforms the other two methods in most scenes, achieving the highest AP, AP_{50}, and AP_{25} scores in the forest scenes CULS, NIBIO, and RMIT. Notably, SATree achieves 100% precision in instance segmentation for the CULS scene. For the SCION and TUWIEN forest scenes, TreeLearn achieves the highest overall AP score. Nevertheless, SATree outperforms TreeLearn in AP_{50} and AP_{25}, demonstrating its robustness across different evaluation metrics.

Fig. 6 visually compares the tree instance segmentation results of the three methods. As in TreeML, SATree reduces under-segmentation and over-segmentation errors, outperforming the other two methods. In the CULS scene (Fig. 6, the first row), SATree avoids segmenting a single tree crown into multiple sub-trees. In the scenes of NIBIO (Fig. 6 row 2) and RMIT (Fig. 6 row 3), SATree successfully detects the trees ignored by other methods, demonstrating the robustness of its stem localization strategy. For the TUWIEN (Fig. 6 row 5) scene, SATree generates segmentation outputs closely aligned with the GT. However, despite this visual alignment, the reported AP scores of SATree are lower than those of TreeLearn. In SCION, SATree does not perform as well as TreeLearn in terms of both quantitative measures (Table 3) and qualitative results (Fig. 6 row 4). It is observed that for coniferous trees,

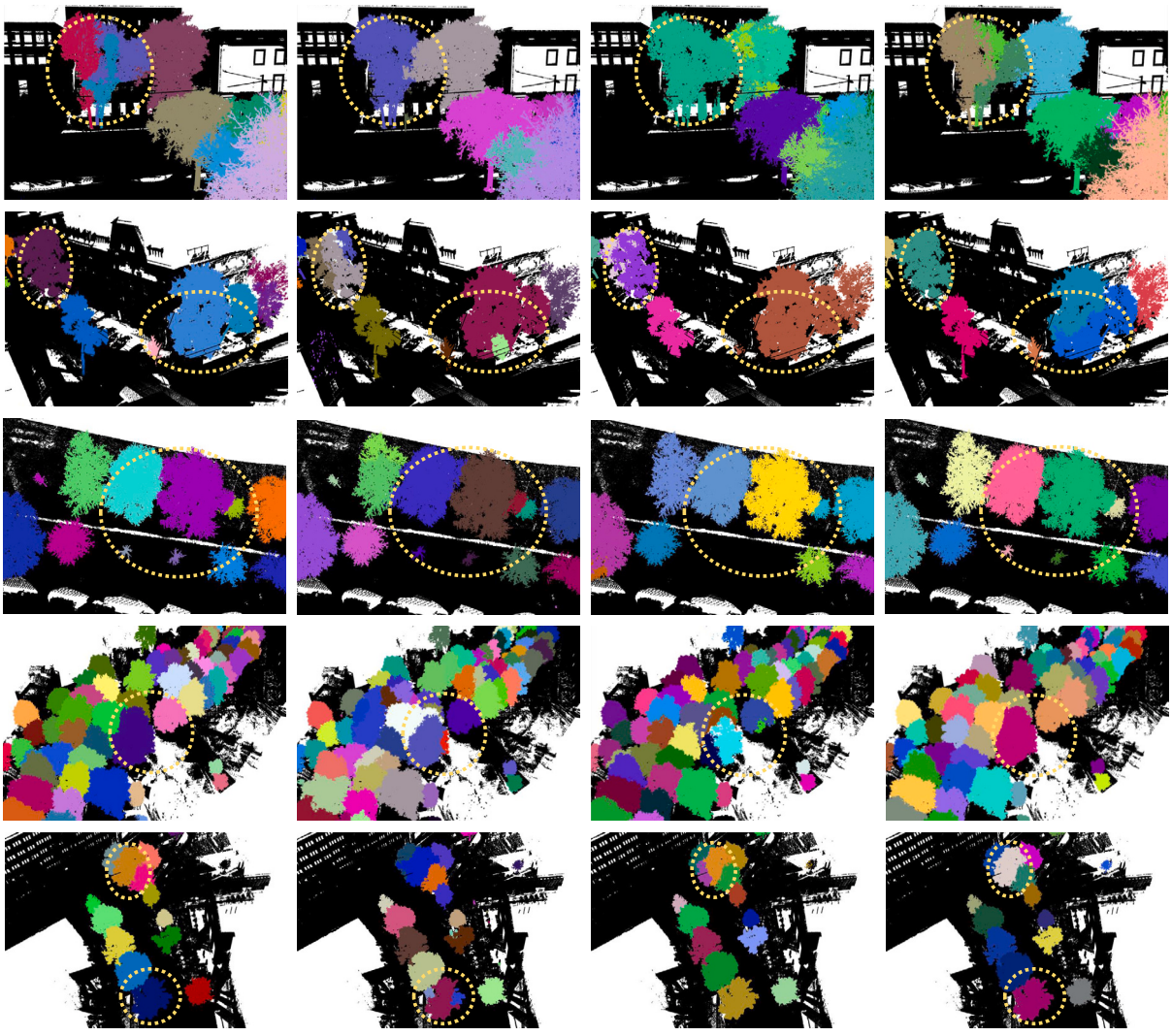
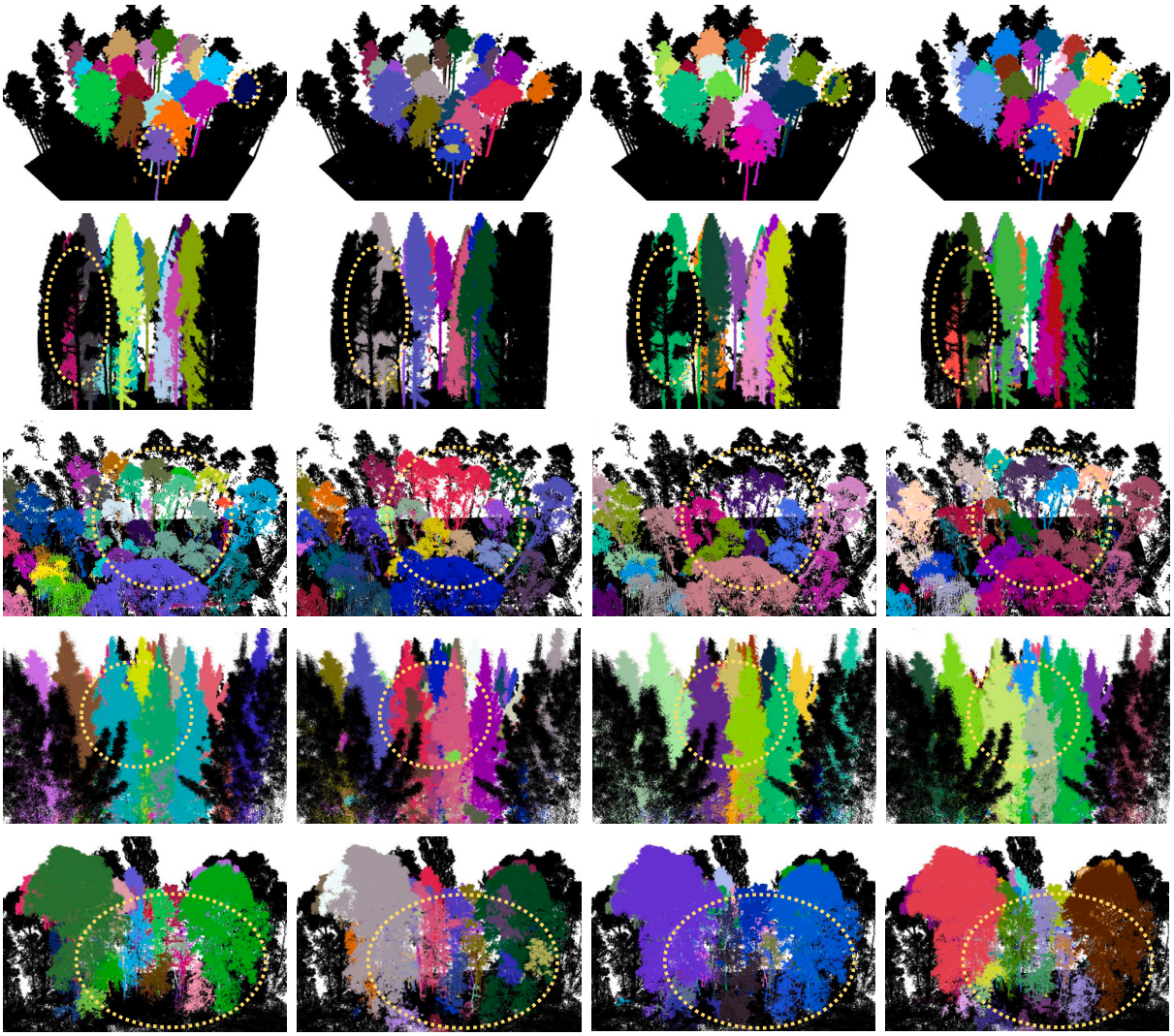


Fig. 5. Tree instance segmentation results achieved on the TreeML dataset by applying TreeSeparation, TreeLearn, and SATree, respectively. The segmented tree instances are randomly colored, and the background points are shown in black. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
3D instance segmentation results on ForInstance (Puliti et al., 2023) with AP, AP_{50}, and AP_{25} scores. SATree achieves the highest scores in three of the five forest scenes.

Forest scene name	Metric	TreeSeparation (J. Wang et al., 2018)	TreeLearn (Henrich et al., 2024)	SATree
CULS	AP	0.922	0.839	**1.000**
	AP_{50}	1.000	0.900	**1.000**
	AP_{25}	1.000	0.900	**1.000**
NIBIO	AP	0.456	0.647	**0.665**
	AP_{50}	0.604	0.764	**0.814**
	AP_{25}	0.629	0.783	**0.857**
RMIT	AP	0.348	0.153	**0.366**
	AP_{50}	0.500	0.230	**0.541**
	AP_{25}	0.616	0.343	**0.670**
SCION	AP	0.354	**0.822**	0.788
	AP_{50}	0.690	0.884	**0.907**
	AP_{25}	0.857	0.884	**0.930**
TUWIEN	AP	0.162	**0.349**	0.295
	AP_{50}	0.343	**0.514**	0.486
	AP_{25}	0.457	0.627	**0.743**



(a) Ground truth **(b)** TreeSeparation **(c)** TreeLearn **(d)** SATree (ours)

Fig. 6. Tree instance segmentation results achieved on the ForInstance dataset by applying TreeSeparation, TreeLearn, and SATree. The five forest scenes (CULS, NIBIO, RMIT, SCION, and TUWIEN) are shown from top to bottom. We use randomized colors to visualize segmented tree instances, while background points are shown in black. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

although SATree can correctly localize individual trees by detecting their stems, it struggles to delineate tree instance boundaries when they are too closely spaced.

Overall, SATree achieves comparable or superior results to other existing methods specifically designed for natural forest scenes, indicating its potential to generalize to such application scenarios.

5. Discussions

In this section, we present detailed discussions of the proposed SATree, including the ablation studies to verify the effectiveness of our approach design, hyperparameter sensitivity analysis, and point density analysis.

5.1. Ablation studies

The goal of ablation studies is to systematically analyze the contribution of individual components within the overall SATree pipeline. Specifically, we have introduced several novel strategies to enhance accurate tree instance segmentation: First, we predict a Gaussian heatmap in the network that benefits stem extraction (Section 3.1); Second, we leverage the local maxima detection from the heat distribution to robustly recognize tree stems from cluster candidates (Section 3.2); Moreover, we grouping tree points into distinct instances using shifted coordinates instead of raw coordinates to better delineate tree boundaries (Section 3.3, Eq. (6)). Ablative studies are performed to verify the effectiveness of these proposed strategies, where we use the *2023-01-09_tum_campus* test scene from TreeML for experiments. Given the challenging forestry structures of this scene, such as dense tree overlaps and significant variations in tree sizes, its performance score will provide a good measure of the proposed strategies. The results of ablative experiments are reported in Table 4.

5.1.1. Heatmap prediction

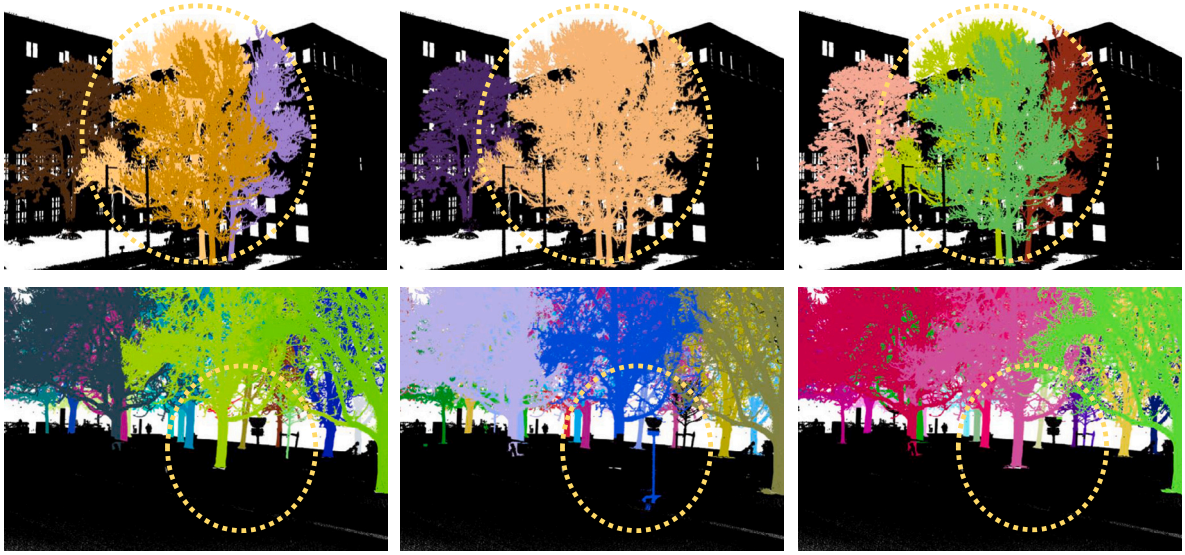
To assess the effectiveness of the heatmap prediction task, we remove this branch from the network along with the supervision term $\mathcal{L}_{\mathrm{h}}$ in Eq. (5). Additionally, we remove the heat value-related criteria from Algorithm 1, and purely rely on the geometrical criterion for stem extraction. Accordingly, Eq. (6) is modified to

$$\mathbf{v_i}' = \mathbf{v_i} + \mathbf{d_i}, \tag{8}$$

where we use the magnitude of the predicted offset vector $\mathbf{d}$ to determine the extent of shifting. As shown in Table 4, adding the heatmap

Table 4
Ablative results achieved by omitting the following key components: heatmap prediction (Section 3.1), local maxima identification for stem extraction (Section 3.2), and using shifted coordinates for tree point grouping (Section 3.3, Eq. (6)).

Test scene	Metric	w/o heatmap	w/o local maxima	w/o shifting coordinates for tree point grouping	SATree
2023-01-09_tum_campus	AP	0.831	0.892	0.713	**0.935**
	AP_{50}	0.892	0.923	0.862	**0.969**
	AP_{25}	0.938	0.938	0.969	**0.985**



(a) GT instances **(b)** without heatmap prediction **(c)** with heatmap prediction

Fig. 7. Visual comparison of the results achieved with and without adopting the heatmap prediction task in the network.

prediction task results in improved performance, achieving an increase of 0.104 in AP, 0.077 in AP_{50}, and 0.047 in AP_{25}. Fig. 7 presents the visual comparison between the results obtained without and with the heatmap prediction branch. These results suggest that heatmap prediction enhances stem localization accuracy and yields cleaner instance segmentation of trees.

5.1.2. Stem extraction by detecting local maxima

In Section 3.2, we refine the stem localization by detecting whether a stem candidate location is a local maximum within the heat distribution. To verify the effectiveness of this strategy, we remove the detection of local maxima and instead only use the heat value threshold to select stems from cluster candidates. Table 4 shows that without the local maxima detection, performance scores drop 0.043 in AP, 0.046 in AP_{50}, and 0.047 in AP_{25}. The main reason is that detecting local maxima helps to recognize small trees with thin stem structures, which are often overlooked when using heat value thresholds alone. Fig. 8 illustrates that the absence of local maxima detection leads to poor segmentation of small trees, either merging them with nearby trees or labeling them as background noise. In contrast, including the local maxima detection significantly enhances performance by reliably identifying most of the smaller trees.

5.1.3. Tree point grouping with shifted coordinates

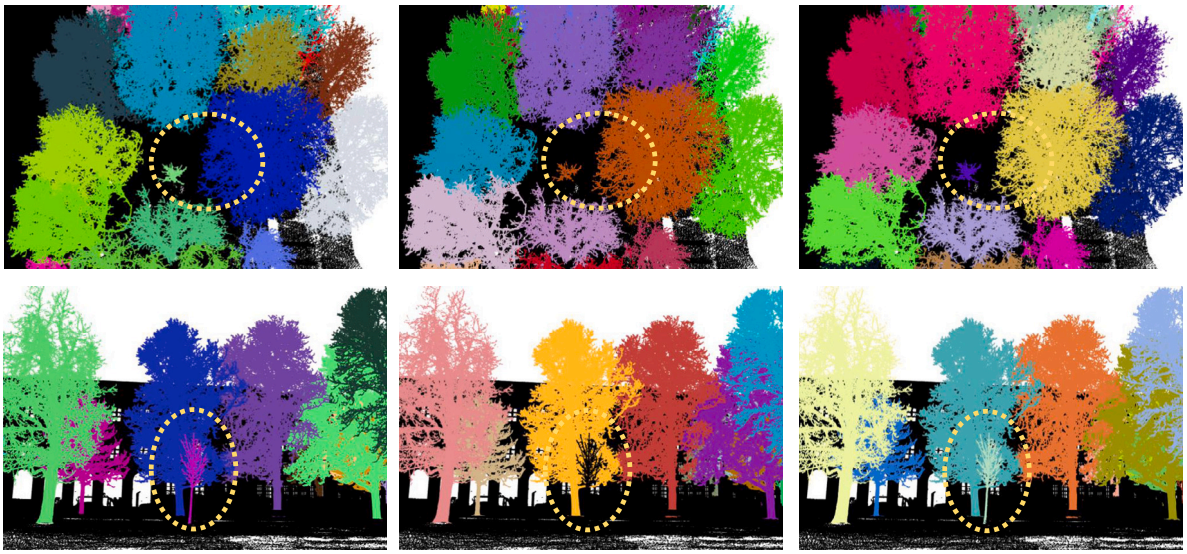
Constructing a forest graph and applying the shortest-path algorithm to isolate single trees has been explored in a few studies (Livny et al., 2010; Tao et al., 2015). Nevertheless, these methods primarily assign tree points to instance roots based on their spatial proxies in the 3D Euclidean space. Different from that, we shift the raw coordinates of 3D tree points using the learned offset embeddings and segment tree instances in the shifted 3D space. In Table 4, we compare the results obtained by grouping tree points based on their raw coordinates versus the results achieved using shifted coordinates. Without using shifted coordinates for tree point grouping, performance scores decrease 0.222 in AP, 0.107 in AP_{50}, and 0.016 in AP_{25}. Specifically, compared to the other ablation experiments, this configuration achieves a higher AP_{25} score but lower scores in AP_{50} and AP. The reason is that grouping tree points using raw coordinates does not impact the stem extraction process. Tree stems could still be accurately localized, leading to a high AP_{25} score. However, it fails to segment regions near tree boundaries, particularly in scenarios with significant overlap between tree branches, thus resulting in poorer performances in AP and AP_{50}. The visual comparison presented in Fig. 9 shows that grouping tree points with raw coordinates fails to preserve tree boundaries, especially when nearby trees exhibit varying sizes. As a result, points from an enormous tree are likely to be assigned to a smaller neighboring tree. In contrast, grouping tree points with shifted coordinates significantly prevents such errors and obtains more natural tree instance boundaries.

5.2. Hyperparameter sensitivity analysis

The proposed SATree involves multiple hyperparameters that need to be manually tuned. We select four important hyperparameters, including λ_2 from the network supervision loss (Eq. (5)), ϵ_h and ϵ_r from Algorithm 1, as well as β from Eq. (6). Then, we use the overall AP metric to evaluate their sensitivity to the resultant segmentation performance, as the AP score gives a comprehensive evaluation averaged across a large range of IoU thresholds. All experiments are performed on the *2023-01-09_tum_campus* set from TreeML and the NIBIO set from ForInstance to provide a consistent analysis.

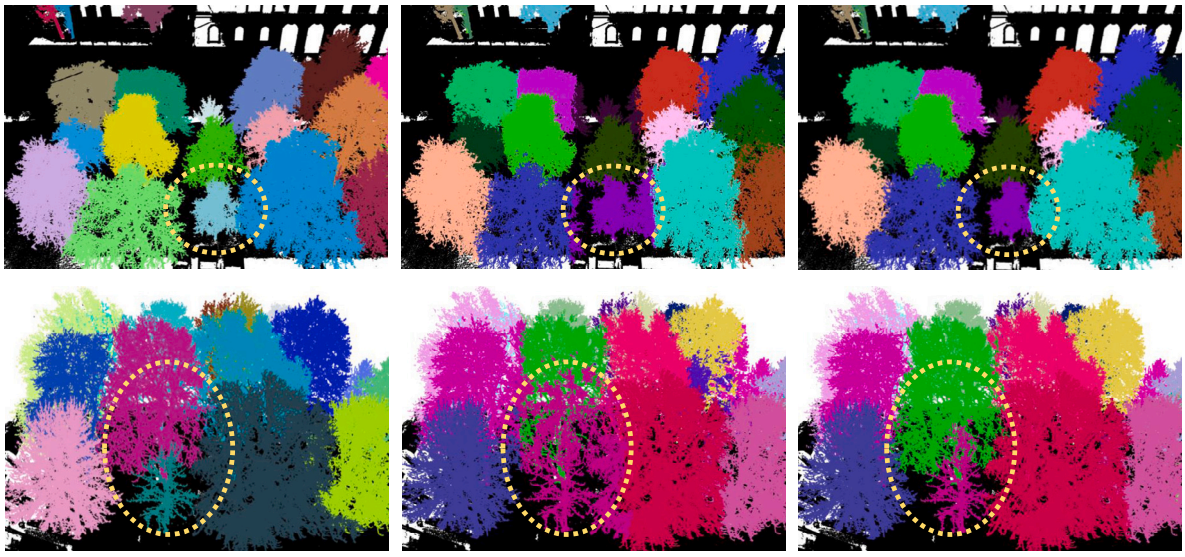
5.2.1. Effect of λ_2 from network supervision

λ_2 is a coefficient from the network supervision loss in Eq. (5), to control the contribution of the offset prediction task during network training. To find the optimal value of λ_2, we initialize it as 0.005 and gradually increase it to 0.4. Fig. 10(a) presents the obtained AP



(a) GT instances **(b)** without local maxima detection **(c)** with local maxima detection

Fig. 8. Visual comparison of the results achieved with and without using the local maxima of the heatmap distribution for stem identification.



(a) GT instances **(b)** grouping with raw coordinates **(c)** grouping with shifted coordinates

Fig. 9. Visual comparison of the results achieved using raw coordinates or using shifted coordinates in the tree point grouping.

scores with respect to different λ_2 values on both datasets. Generally, an excessively small λ_2 results in degraded segmentation performance. This is due to that the network tends to ignore the offset prediction task under a small coefficient value, which can lead to noisy offset predictions and impair the subsequent tree point grouping. On the other hand, a too large λ_2 overly prioritizes the offset prediction task that can negatively affect the accuracy of both semantic segmentation and heatmap regression. Therefore, it is important to find an appropriate value of λ_2 to achieve a balanced optimization among the three tasks. For the TreeML dataset, the optimal λ_2 is 0.05; On ForInstance, this optimal value is slightly larger, *e.g.*, 0.2. This is because ForInstance mostly comprises coniferous trees that exhibit smaller offset vector magnitudes than the broadleaf trees in TreeML, and thus requires a larger λ_2 to sufficiently emphasize the offset prediction task during network training.

5.2.2. Effects of ϵ_h and ϵ_r from stem localization

In the stem localization Algorithm 1, ϵ_h is a threshold hyperparameter discarding stem cluster candidates with insufficient heat values. However, purely relying on ϵ_h overlooks small trees that have weaker heat responses. ϵ_r, the searching radius for identifying local maxima in the heat map, serves as a complementary criterion to the ϵ_h-based filtering strategy. We incrementally increase ϵ_h from 0 to 0.9 and enlarge ϵ_r from 0.05 m to 0.25 m to analyze their influences on the segmentation performance.

The quantitative results visualized in Fig. 10(b)–(c) suggest that our proposed SATree method remains robust against varying ϵ_h and ϵ_r values on the TreeML dataset. This is attributed to the fact that the ϵ_h- and ϵ_r-based criteria mutually complement each other. If the ϵ_h-based thresholding fails to detect small tree stems with low heat responses, these stems can still be recovered through the ϵ_r-based local maxima detection. Only when ϵ_h is extremely low (*i.e.*, $\epsilon_h = 0$), it

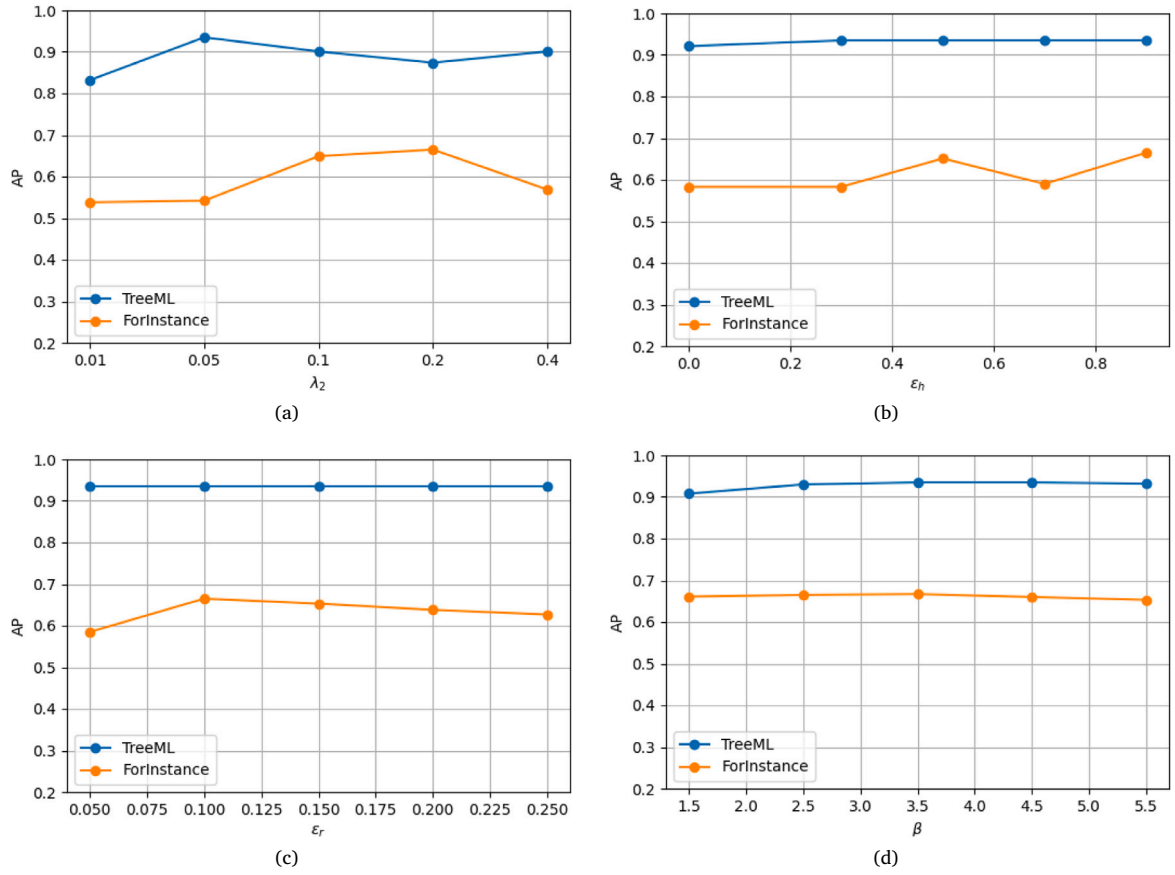


Fig. 10. Effect of the hyperparameters λ_2, ϵ_h, ϵ_r, and β on AP scores of the two datasets.

may introduce false positives tree stem detections, leading to a slight decrease in the AP score. Besides, as TreeML is an urban forestry dataset mainly containing street-level trees with regular spatial separation, we recommend setting ϵ_r between 0.05 m and 0.25 m for TreeML and similar urban-level forestry scenes.

ForInstance dataset, on the contrary, is more sensitive to the values of ϵ_h and ϵ_r. This is due to the fact that the test set NIBIO scene majorly contains coniferous trees with thick and upright stems. These stems tend to exhibit high Gaussian heat responses. Consequently, we recommend setting a high ϵ_h to prevent false positive stem detections. As illustrated in Fig. 10(b), a ϵ_h of 0.9 leads to the optimal tree segmentation performance on this dataset. Regarding the searching radius ϵ_r, coniferous trees in natural forests are often densely and irregularly distributed. An overly large searching radius can encompass adjacent stems and result in erroneous local maxima detection. Thus, we recommend a value of 0.01 m for this dataset.

5.2.3. Effects of β from tree point grouping

The hyperparameter β serves as a user-defined step magnitude in the tree point grouping stage (Eq. (6)), and is designed to encourage individual tree points to shift towards their instance centroids. To analyze the effect of β on the segmentation performance, we initialize it as 1.5 m and gradually increase it to 5.5 m, reporting the resulting AP scores on two datasets in Fig. 10(d). It can be seen that SATree achieves the highest AP score on the TreeML dataset when β is within the range of [3.5 m, 4.5 m], and achieves the optimal segmentation performance on the ForInstance dataset when β lies in the interval of [2.5 m, 3.5 m]. This indicates that an ideal β should respect the natural spatial scale of different tree species. For handling urban broadleaf forests, a slightly larger β (*e.g.*, 3.5 m–4.5 m) is more suitable. To tackle coniferous trees, we recommend a smaller β (*e.g.*, 2.5 m–3.5 m). Nevertheless, overall, SATree attains relatively stable AP scores on both datasets across various β values, suggesting that the proposed method is robust to moderate variations in this hyperparameter.

5.3. Point density analysis

To evaluate the robustness of the proposed stem localization and graph-based individual tree point grouping strategy under varying point densities, we randomly drop a certain percentage of points from the point clouds generated in Section 3.1. Then, we perform stem localization and tree point grouping on subsampled point clouds. The *2023-01-09_tum_campus* set from TreeML and the NIBIO set from ForInstance are used for experiments to keep consistent with previous sections. Fig. 11 reports the performance scores on two datasets under different point drop rates. Our method shows strong robustness on the TreeML dataset, where AP scores remain stable until the point drop rate of 70%. Only when the drop rate increases to 90%, there is a noticeable performance degradation. This suggests that our method is highly effective in addressing urban forestry segmentation tasks. However, performance on the ForInstance dataset is more sensitive to varying point densities, suggesting that accurate segmentation of natural coniferous forests requires higher-quality and denser point clouds. In the next section, we further visualize a few failure cases to analyze the reason why the graph-based shortest-path algorithm is limited to such coniferous forest scenes.

6. Limitations and future work

Our proposed SATree is primarily designed for urban and semi-structured tree environments. It has demonstrated high-fidelity tree instance segmentation in complex urban environments by leveraging

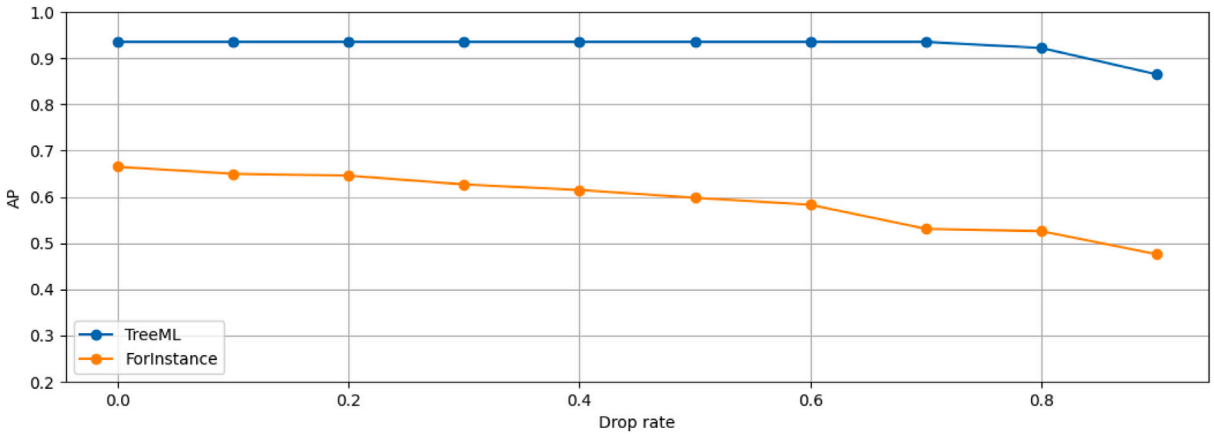
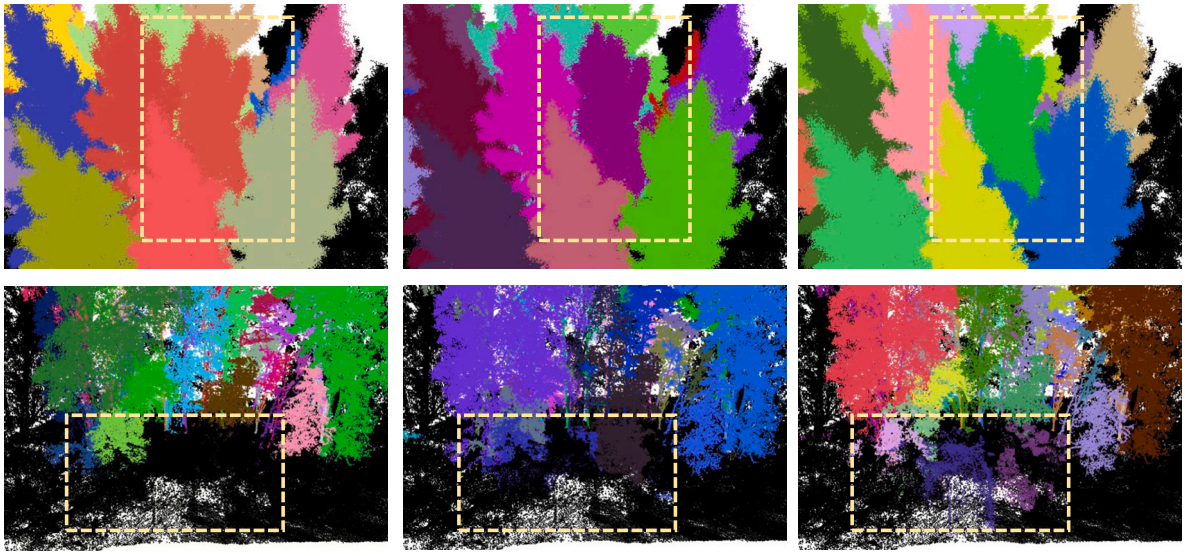


Fig. 11. Segmentation performance of SATree on the two datasets under different levels of downsampling. The horizontal axis represents the percentage of points randomly dropped from the point cloud.



(a) Ground truth **(b)** TreeLearn **(c)** SATree (ours)

Fig. 12. Performance degradation of SATree, in comparison with TreeLearn, on the two natural forest scenes from the ForInstance dataset. The top row represents SCION and the bottom row represents TUWIEN. We use yellow rectangle boxes to indicate the areas where we have observed a reduced segmentation performance.

detected tree structures, such as stems and crowns. However, it still suffers from several limitations.

First, when extending SATree from urban forestry scenes to natural forestry scenes, we have observed inferior performances on SCION and TUWIEN scenes (see Table 3), compared with TreeLearn (Henrich et al., 2024). SCION contains densely distributed coniferous trees with vertical stems and narrow-shaped crowns. Although our method can reliably detect tree stems, it fails to correctly delineate tree instance boundaries, as shown in the top row of Fig. 12. This is because we use the graph-based approach to source each tree point to a tree stem according to the shortest path in Euclidean space. For closely adjacent coniferous trees, even with the guidance of learned offset embeddings, the shortest-path algorithm can wrongly assign points from neighboring trees to the same tree instance. Unlike SCION, TUWIEN represents a more heterogeneous natural forest environment containing trees that greatly vary in height, size, species, and stem morphologies. This is a highly challenging scene for all three segmentation methods TreeSeparation (J. Wang et al., 2018), TreeLearn (Henrich et al., 2024), and SATree. Compared to TreeLearn, our method incorrectly segments low vegetation and shrubs as individual trees, leading to more false positive tree instances and a decrease in the AP score. In the future, we aim to incorporate additional structural constraints into the shortest-path-based segmentation strategy by enforcing consistency of the learned offset embeddings and by modeling biologically plausible branch growth patterns. These enhancements will reduce erroneous assignments of adjacent tree points or data outliers.

Another limitation lies in the reliance on manually tuned hyperparameters for stem localization and tree point grouping. The hyperparameter analysis in Fig. 10 has illustrated that, while the post-processing strategies show relatively robust performance on the urban TreeML dataset against various hyperparameter values, the segmentation performance on the natural ForInstance dataset is more sensitive to hyperparameter selection. Nevertheless, as the key hyperparameters (*e.g.*, ϵ_h, ϵ_r, β) in our method have clear geometric or spectral meanings, selecting values within a reasonable range that reflect tree species characteristics and tree morphologies naturally yields a good segmentation performance. In the future, we plan to incorporate a learnable regression module to explicitly predict and regress 3D tree stem locations based on the learned Gaussian heat responses. This module will replace the current stem localization Algorithm 1, discarding the need for manual hyperparameter tuning and improving the generalizability to diverse natural forest environments.

Last, there exist potential challenges when scaling SATree to extremely large forestry scenes. Currently, we patch the large-scale scenes into smaller sub-scenes as network inputs, which achieves a good tree segmentation performance. For example, the *2023-01-16_44* scene in TreeML is the largest test scene containing 327 trees. Using the patching strategy, SATree achieves an AP score of 0.981. However, patching introduces additional complexity in data pre-processing and post-processing and may fragment complete tree instances across patch boundaries. In future work, we consider exploring alternative data representations, such as supervoxel- or superpoint-based 3D representations, to achieve efficient processing of extremely large scenes without patching and guarantee instance integrity.

7. Conclusions

3D instance segmentation of trees is a fundamental yet challenging task in forestry remote sensing studies. Due to the complexity of 3D scenes as well as variations in tree geometric shapes, sizes, and species, most existing methods suffer from either over- or under-segmentation errors. To address this issue, we have proposed SATree, a novel deep learning-based method to accurately segment 3D tree instances in challenging urban forestry scenes at large scales.

Driven by an intuitive observation that an individual tree is structurally composed of a stem and a crown, SATree is proposed to explicitly detect and leverage key structural components, *e.g.*, stem, to guide tree instance segmentation in complex urban scenes. Specifically, the detected stem structures serve as geometric anchors to effectively delineate individual trees in scenes with dense crown overlaps, branch interference, occlusions, and varying tree shape complexities. Unlike existing methods (Z. Luo et al., 2021; Jiang et al., 2023b; Henrich et al., 2024) that do not seek to model tree structural components, we develop a multi-task learning network that simultaneously classifies crown points and stem points. A 2D Gaussian heatmap is further predicted to enable robust tree stem localization. Key strategies, such as localizing stems by detecting local maxima in the heatmap and graph-based tree point grouping by integrating offset embeddings, proved effective in obtaining accurate tree instances. Extensive experiments on two public forestry datasets have demonstrated the superiority of SATree over existing methods. SATree shows strong performance in segmenting trees in large urban street-level scenes and generalizes well to natural forest environments.

Our research outputs are accurately segmented 3D tree instances in urban environments, which benefit various downstream applications such as urban forestry inventory, tree structural parameter measurement, biomass estimation, and carbon cycle modeling. Besides, SATree learns to generate Gaussian heatmaps for urban trees, where high heat values indicate tree stems or main branch structures, and low heat values correspond to small twigs near tree boundaries. This characteristic of the heatmaps suggests their potential usage in fine-grained part-level segmentation of trees. Another by-product is the generated MST graphs in tree point grouping (Section 3.3). These MSTs approximate the skeletal structures of tree instances and can offer valuable information for reconstructing 3D forest structures from raw input data. As an extension of this work, we plan to apply SATree for the subsequent estimation of tree structural parameters, tree carbon storage, and biomass. Together with these efforts, we aim to strengthen evidence-based urban greening strategies, improve decision-support tools for planners, and offer new insights for research in climate resilience, biodiversity, and green infrastructure performance.

CRediT authorship contribution statement

Shenglan Du: Writing – original draft, Visualization, Validation, Methodology. **Jantien Stoter:** Writing – review & editing, Supervision, Project administration, Funding acquisition. **Julian F.P. Kooij:** Writing – review & editing, Supervision, Funding acquisition. **Liangliang Nan:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the 3D Urban Understanding Lab funded by the TU Delft AI Initiative.

Data availability

Both forestry datasets used in this study are publicly available. Please refer to TreeML at https://doi.org/10.6084/m9.figshare.c.6788358.v1and ForInstance at https://zenodo.org/records/8287792.

References

Ammar, A., Koubaa, A., Benjdira, B., 2021. Deep-learning-based automated palm tree counting and geolocation in large farms from aerial geotagged images. Agronomy 11 (8), 1458.

Arief, H.A., Strand, G.-H., Tveite, H., Indahl, U.G., 2018. Land cover segmentation of airborne LiDAR data using stochastic atrous network. Remote. Sens. 10 (6), 973.

Ayrey, E., Fraver, S., Kershaw Jr., J.A., Kenefic, L.S., Hayes, D., Weiskittel, A.R., Roth, B.E., 2017. Layer stacking: A novel algorithm for individual forest tree segmentation from LiDAR point clouds. Can. J. Remote Sens. 43 (1), 16–27.

Beucher, S., 1979. Use of watersheds in contour detection. In: Proc. Int. Workshop on Image Processing, Sept. 1979. pp. 17–21.

Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y.M., 2020. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

Burmeister, J.-M., Richter, R., Reder, S., Mund, J.-P., Döllner, J., 2024. Tree instance segmentation in urban 3D point clouds using a coarse-to-fine algorithm based on semantic segmentation. ISPRS Ann. Photogramm. Remote. Sens. Spat. Inf. Sci. 10, 79–86.

Chang, L., Fan, H., Zhu, N., Dong, Z., 2022. A two-stage approach for individual tree segmentation from TLS point clouds. IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens. 15, 8682–8693.

Chen, Q., Baldocchi, D., Gong, P., Kelly, M., 2006. Isolating individual trees in a savanna woodland using small footprint lidar data. Photogramm. Eng. Remote Sens. 72 (8), 923–932.

Chen, S., Fang, J., Zhang, Q., Liu, W., Wang, X., 2021. Hierarchical aggregation for 3d instance segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15467–15476.

Chen, X., Jiang, K., Zhu, Y., Wang, X., Yun, T., 2021. Individual tree crown segmentation directly from UAV-borne LiDAR data using the PointNet of deep learning. Forests 12 (2), 131.

Dassot, M., Constant, T., Fournier, M., 2011. The use of terrestrial LiDAR technology in forest science: application fields, benefits and challenges. Ann. For. Sci. 68, 959–974.

Dralle, K., Rudemo, M., 1996. Stem number estimation by kernel smoothing of aerial photos. Can. J. Forest Res. 26 (7), 1228–1236.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q., 2019. Centernet: Keypoint triplets for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6569–6578.

Fan, G., Nan, L., Dong, Y., Su, X., Chen, F., 2020. Adqsm: A new method for estimating above-ground biomass from TLS point clouds. Remote. Sens. 12 (18), 3089.

Gupta, S., Weinacker, H., Koch, B., 2010. Comparative analysis of clustering-based approaches for 3-D single tree detection using airborne fullwave lidar data. Remote. Sens. 2 (4), 968–989.

Hakula, A., Ruoppa, L., Lehtomäki, M., Yu, X., Kukko, A., Kaartinen, H., Taher, J., Matikainen, L., Hyyppä, E., Luoma, V., et al., 2023. Individual tree segmentation and species classification using high-density close-range multispectral laser scanning data. ISPRS Open J. Photogramm. Remote. Sens. 9, 100039.

Hamraz, H., Jacobs, N.B., Contreras, M.A., Clark, C.H., 2019. Deep learning for conifer/deciduous classification of airborne LiDAR 3D point clouds representing individual trees. ISPRS J. Photogramm. Remote Sens. 158, 219–230.

Heinzel, J., Huber, M.O., 2018. Constrained spectral clustering of individual trees in dense forest using terrestrial laser scanning data. Remote. Sens. 10 (7), 1056.

Henrich, J., van Delden, J., Seidel, D., Kneib, T., Ecker, A.S., 2024. TreeLearn: A deep learning method for segmenting individual trees from ground-based LiDAR forest point clouds. Ecol. Informatics 84, 102888.

Hu, Q., Yang, B., Khalid, S., Xiao, W., Trigoni, N., Markham, A., 2021. Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4977–4987.

Hyyppä, J., Kelle, O., Lehikoinen, M., Inkinen, M., 2001. A segmentation-based method to retrieve stem volume estimates from 3-d tree height models produced by laser scanners. IEEE Trans. Geosci. Remote Sens. 39 (5), 969–975.

Hyyppä, J., Yu, X., Hyyppä, H., Vastaranta, M., Holopainen, M., Kukko, A., Kaartinen, H., Jaakkola, A., Vaaja, M., Koskinen, J., et al., 2012. Advances in forest inventory using airborne laser scanning. Remote. Sens. 4 (5), 1190–1207.

Jiang, T., Liu, S., Zhang, Q., Xu, X., Sun, J., Wang, Y., 2023a. Segmentation of individual trees in urban MLS point clouds using a deep learning framework based on cylindrical convolution network. Int. J. Appl. Earth Obs. Geoinf. 123, 103473.

Jiang, T., Wang, Y., Liu, S., Zhang, Q., Zhao, L., Sun, J., 2023b. Instance recognition of street trees from urban point clouds using a three-stage neural network. ISPRS J. Photogramm. Remote Sens. 199, 305–334.

Jiang, L., Zhao, H., Shi, S., Liu, S., Fu, C.-W., Jia, J., 2020. Pointgroup: Dual-set point grouping for 3d instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4867–4876.

Lai, X., Liu, J., Jiang, L., Wang, L., Zhao, H., Liu, S., Qi, X., Jia, J., 2022. Stratified transformer for 3d point cloud segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8500–8509.

Lee, H., Slatton, K.C., Roth, B.E., Cropper Jr., W., 2010. Adaptive clustering of airborne LiDAR data to segment individual tree crowns in managed pine forests. Int. J. Remote Sens. 31 (1), 117–139.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2980–2988.

Lin, H., Zheng, X., Li, L., Chao, F., Wang, S., Wang, Y., Tian, Y., Ji, R., 2023. Meta architecture for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17682–17691.

Livny, Y., Yan, F., Olson, M., Chen, B., Zhang, H., El-Sana, J., 2010. Automatic reconstruction of tree skeletal structures from point clouds. In: ACM SIGGRAPH Asia 2010 Papers. pp. 1–8.

Luo, H., Khoshelham, K., Chen, C., He, H., 2021. Individual tree extraction from urban mobile laser scanning point clouds using deep pointwise direction embedding. ISPRS J. Photogramm. Remote Sens. 175, 326–339.

Luo, Z., Zhang, Z., Li, W., Chen, Y., Wang, C., Nurunnabi, A.A.M., Li, J., 2021. Detection of individual trees in UAV LiDAR point clouds using a deep learning framework based on multichannel representation. IEEE Trans. Geosci. Remote Sens. 60, 1–15.

Ma, J., Han, T., Wang, C., Zhang, X., Zhang, X., Zhang, W., Chen, Y., 2025. Individual tree segmentation via contrastive learning and semantic priors in point clouds. Urban For. Urban Green. 129018.

Malladi, M.V., Guadagnino, T., Lobefaro, L., Mattamala, M., Griess, H., Schweier, J., Chebrolu, N., Fallon, M., Behley, J., Stachniss, C., 2024. Tree instance segmentation and traits estimation for forestry environments exploiting LiDAR data collected by mobile robots. In: 2024 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 17933–17940.

Ning, X., Ma, Y., Jin, H., Wu, Z., Zhou, W., Zhang, X., Guo, J., 2025. Tree and shrub instance segmentation by boundary-aware ecological structural probability analysis. Urban For. Urban Green. 129123.

Olofsson, K., Holmgren, J., Olsson, H., 2014. Tree stem and height measurements using terrestrial laser scanning and the RANSAC algorithm. Remote. Sens. 6 (5), 4323–4344.

Osco, L.P., De Arruda, M.d.S., Junior, J.M., Da Silva, N.B., Ramos, A.P.M., Moryia, É.A.S., Imai, N.N., Pereira, D.R., Creste, J.E., Matsubara, E.T., et al., 2020. A convolutional neural network approach for counting and geolocating citrus-trees in UAV multispectral imagery. ISPRS J. Photogramm. Remote Sens. 160, 97–106.

Pu, Y., Xu, D., Wang, H., Li, X., Xu, X., 2023. A new strategy for individual tree detection and segmentation from leaf-on and leaf-off UAV-LiDAR point clouds based on automatic detection of seed points. Remote. Sens. 15 (6), 1619.

Puliti, S., Pearse, G., Surovỳ, P., Wallace, L., Hollaus, M., Wielgosz, M., Astrup, R., 2023. For-instance: a uav laser scanning benchmark dataset for semantic and instance segmentation of individual trees. arXiv preprint arXiv:2309.01279.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Adv. Neural Inf. Process. Syst. 30.

Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B., 2022. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. Adv. Neural Inf. Process. Syst. 35, 23192–23204.

Reche-Martinez, A., Martin, I., Drettakis, G., 2004. Volumetric reconstruction and interactive rendering of trees from photographs. In: ACM SIGGRAPH 2004 Papers. pp. 720–727.

Redmon, J., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Shrestha, D.B., Sharma, R.P., Bhandari, S.K., 2018. Individual tree aboveground biomass for castanopsis indica in the mid-hills of nepal. Agrofor. Syst. 92, 1611–1623.

Sun, Y., Jin, X., Pukkala, T., Li, F., 2022a. Predicting individual tree diameter of larch (larix olgensis) from UAV-LiDAR data using six different algorithms. Remote. Sens. 14 (5), 1125.

Sun, Y., Li, Z., He, H., Guo, L., Zhang, X., Xin, Q., 2022b. Counting trees in a subtropical mega city using the instance segmentation method. Int. J. Appl. Earth Obs. Geoinf. 106, 102662.

Tan, M., Pang, R., Le, Q.V., 2020. Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10781–10790.

Tao, S., Wu, F., Guo, Q., Wang, Y., Li, W., Xue, B., Hu, X., Li, P., Tian, D., Li, C., et al., 2015. Segmenting tree crowns from terrestrial and mobile LiDAR data by exploring ecological theories. ISPRS J. Photogramm. Remote Sens. 110, 66–76.

Thomas, H., Qi, C.R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L.J., 2019. Kpconv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6411–6420.

Vu, T., Kim, K., Luu, T.M., Nguyen, T., Yoo, C.D., 2022. Softgroup for 3d instance segmentation on point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2708–2717.

Wang, J., Chen, X., Cao, L., An, F., Chen, B., Xue, L., Yun, T., 2019. Individual rubber tree segmentation based on ground-based LiDAR data and faster R-CNN of deep learning. Forests 10 (9), 793.

Wang, D., Liang, X., Mofack, G.I., Martin-Ducup, O., 2021. Individual tree extraction from terrestrial laser scanning data via graph pathing. For. Ecosyst. 8, 1–11.

Wang, J., Lindenbergh, R., Menenti, M., 2018. Scalable individual tree delineation in 3D point clouds. Photogramm. Rec. 33 (163), 315–340.

Wang, P., Tang, Y., Liao, Z., Yan, Y., Dai, L., Liu, S., Jiang, T., 2023. Road-side individual tree segmentation from urban MLS point clouds using metric learning. Remote. Sens. 15 (8), 1992.

Wang, W., Yu, R., Huang, Q., Neumann, U., 2018. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2569–2578.

Weinstein, B.G., Marconi, S., Aubry-Kientz, M., Vincent, G., Senyondo, H., White, E.P., 2020. DeepForest: A python package for RGB deep learning tree crown delineation. Methods Ecol. Evol. 11 (12), 1743–1751.

Weinstein, B.G., Marconi, S., Bohlman, S., Zare, A., White, E., 2019. Individual tree-crown detection in RGB imagery using semi-supervised deep learning neural networks. Remote. Sens. 11 (11), 1309.

Wielgosz, M., Puliti, S., Xiang, B., Schindler, K., Astrup, R., 2024. SegmentAnyTree: A sensor and platform agnostic deep learning model for tree segmentation using laser scanning data. Remote Sens. Environ. 313.

Wulder, M., Niemann, K.O., Goodenough, D.G., 2000. Local maximum filtering for the extraction of tree locations and basal area from high spatial resolution imagery. Remote Sens. Environ. 73 (1), 103–114.

Xi, Z., Hopkinson, C., 2021. Detecting individual-tree crown regions from terrestrial laser scans with an anchor-free deep learning model. Can. J. Remote Sens. 47 (2), 228–242.

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

Yang, J., Gan, R., Luo, B., Wang, A., Shi, S., Du, L., 2024. An improved method for individual tree segmentation in complex urban scenes based on using multispectral LiDAR by deep learning. IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens. 17, 6561–6576.

Yazdi, H., Shu, Q., Rötzer, T., Petzold, F., Ludwig, F., 2024. A multilayered urban tree dataset of point clouds, quantitative structure and graph models. Sci. Data 11 (1), 28.

Yun, T., Jiang, K., Li, G., Eichhorn, M.P., Fan, J., Liu, F., Chen, B., An, F., Cao, L., 2021. Individual tree crown segmentation from airborne LiDAR data using a novel Gaussian filter and energy function minimization-based approach. Remote Sens. Environ. 256, 112307.

Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V., 2021. Point transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16259–16268.