forests

MDPI

Article

From Crown Detection to Boundary Segmentation: Advancing Forest Analytics with Enhanced YOLO Model and Airborne LiDAR Point Clouds

Yanan Liu [1,2,*], Ai Zhang [1,2] and Peng Gao [1,2]

[1] School of Geomatics and Urban Spatial Informatics, Beijing University of Civil Engineering and Architecture, Beijing 100044, China; 2108570023082@stu.bucea.edu.cn (A.Z.); 2108570022088@stu.bucea.edu.cn (P.G.)
[2] Engineering Research Center of Representative Building and Architectural Heritage Database, No.15 Yongyuan Road, Beijing 102616, China
* Correspondence: liuyanan@bucea.edu.cn

Abstract: Individual tree segmentation is crucial to extract forest structural parameters, which is vital for forest resource management and ecological monitoring. Airborne LiDAR (ALS), with its ability to rapidly and accurately acquire three-dimensional forest structural information, has become an essential tool for large-scale forest monitoring. However, accurately locating individual trees and mapping canopy boundaries continues to be hindered by the overlapping nature of the tree canopies, especially in dense forests. To address these issues, this study introduces CCD-YOLO, a novel deep learning-based network for individual tree segmentation from the ALS point cloud. The proposed approach introduces key architectural enhancements to the YOLO framework, including (1) the integration of cross residual transformer network extended (CReToNeXt) backbone for feature extraction and multi-scale feature fusion, (2) the application of the convolutional block attention module (CBAM) to emphasize tree crown features while suppressing noise, and (3) a dynamic head for adaptive multi-layer feature fusion, enhancing boundary delineation accuracy. The proposed network was trained using a newly generated individual tree segmentation (ITS) dataset collected from a dense forest. A comprehensive evaluation of the experimental results was conducted across varying forest densities, encompassing a variety of both internal and external consistency assessments. The model outperforms the commonly used watershed algorithm and commercial LiDAR 360 software, achieving the highest indices (precision, F1, and recall) in both tree crown detection and boundary segmentation stages. This study highlights the potential of CCD-YOLO as an efficient and scalable solution for addressing the critical challenges of accuracy segmentation in complex forests. In the future, we will focus on enhancing the model's performance and application.

Keywords: YOLO; individual tree segmentation; airborne LiDAR; crown detection; boundary segmentation

Academic Editor: Fa Li

Received: 31 December 2024
Revised: 21 January 2025
Accepted: 25 January 2025
Published: 28 January 2025

Citation: Liu, Y.; Zhang, A.; Gao, P. From Crown Detection to Boundary Segmentation: Advancing Forest Analytics with Enhanced YOLO Model and Airborne LiDAR Point Clouds. *Forests* **2025**, *16*, 248. https://doi.org/10.3390/f16020248

1. Introduction

Forest ecosystems, which account for approximately 77% of global terrestrial carbon stock, are essential for maintaining biodiversity and regulating biogeochemical cycles. As the largest carbon sink, forests play a crucial role in maintaining the global carbon balance

and mitigating climate change [1]. Precise monitoring and scientific management of forest resources are therefore of paramount importance. Traditional forest surveys, which rely on field investigations of individual tree parameters, are limited by their time-consuming, labor-intensive nature and their inability to scale across vast or inaccessible regions. LiDAR (light detection and ranging) technology, particularly airborne laser scanning (ALS), offers a rapid and precise alternative for acquiring large-scale three-dimensional forest structures. ALS enables the estimation of forest structural parameters such as tree height, crown width, biomass, and volume [2]. However, accurate segmentation of individual trees remains a critical prerequisite for utilizing this data effectively, as it directly influences the reliability of derived forest metrics.

Traditional ALS-based tree segmentation approaches can be broadly categorized into 2D image-based and 3D point cloud-based approaches. The 2D image methods utilize CHM (canopy height model) or DSM (digital surface model) raster data to represent the upper contour of the tree crown and identify the local maximum of the treetops to segment individual trees [3]. Common methods include watershed algorithms with marker-controlled methods [4], region growing algorithms [5], and others [6]. A key limitation of these methods is the difficulty in accurately detecting local maxima from input images, as variations in canopy height and tree crown overlap can result in false positives or missed detections, reducing the precision of tree segmentation. In contrast, 3D point cloud methods, such as K-means clustering [7], region growing algorithm [8], mean-shift clustering [9], graph segmentation [10], and spectral clustering [11], can leverage the spatial structure to improve segmentation but suffer from high computational complexity, sensitivity to noise, and reliance on manually tuned parameters, limiting their adaptability to diverse forest environments. Moreover, they rely heavily on prior knowledge of forest characteristics, which limits their ability to adapt to different forest types and datasets, and hinders their generalizability and transferability across various forest environments [12]. Machine learning-based methods, such as conditional random fields (CRF) [13] and random forests (RF) [14], have been applied for tree segmentation in 3D point clouds, effectively identifying tree structures and filtering non-tree points during the coarse segmentation, providing a foundation for subsequent instance segmentation optimization. However, these approaches rely heavily on feature engineering and face limitations in handling the irregularity and high dimensionality of point cloud, which restricts their scalability and adaptability to diverse forest environments. The proposed method offers the possibility to avoid feature engineering tasks like variable transformation and variable selection.

Recent advances in deep learning (DL), using the original 3D points or its derived products from 3D point clouds, have shown great potential for individual tree segmentation. These innovations have enabled more accurate and efficient analysis of forest structure, enhancing the ability to monitor tree growth, biodiversity, and ecological changes. For the former, it begins by segmenting or classifying the 3D point cloud into distinct parts using multilayer perceptron (MLP), followed by clustering to further refine the segmentation. Krisanski et al. [15] leveraged PointNet++ to directly perform classification, localization, and semantic segmentation tasks on point cloud. They combined this with clustering algorithms to effectively identify individual trees within dense terrestrial laser scanning (TLS) data. Similarly, Kang and Wang [16] utilized PointNet++ on multi-sensor fusion data to successfully segment fruit in natural orchards. However, the high point density requirement inherent in these methods restricts their widespread applicability. Wielgosz et al. [17] developed Point2Tree, a modular framework combining semantic segmentation, instance segmentation, and hyperparameter optimization. It classifies point clouds and uses graph-based methods with Dijkstra's algorithm to assign tree points to instances. Henrich et al. [18] proposed the TreeLearn model, an automated tree segmentation method using a 3D U-Net sparse convolutional network. It predicts tree points and

offsets, with clustering and post-processing for instance generation, achieving high accuracy and robustness without complex hyperparameter tuning. However, the high dimensionality and irregularity of the 3D point cloud increase computational complexity and make the training and inference more challenging [19]; these methods are often applied to terrestrial or mobile LiDAR systems. To address these dimensionality challenges, a method for creating a multi-feature point cloud map is introduced for the proposed method. For the latter, a convolutional neural network (CNN) [20] is applied to the segmented 2D images derived from 3D point clouds. These networks, by leveraging well-established CNN architectures, are capable of automatically learning and extracting complex spatial features, allowing for precise segmentation of the upper tree canopy. Additionally, they capture important information from the mid and lower layers of trees, thereby providing a robust foundation for automated tree segmentation tasks. In the task of crown segmentation, methods can be broadly categorized into pixel-level segmentation and instance segmentation. For the pixel-level segmentation methods, such as fully convolutional networks (FCN) [21], U-Net [22], and the DeepLab [23], classify each pixel to achieve precise delineation of crown boundaries, making them suitable for tasks with relatively simple targets and homogeneous backgrounds. U-Net is used for individual tree crown delineation in high-resolution remote sensing imagery [24], while DeepLab and domain-adaptive networks have been used to detect palm trees in the Amazon and Southeast Asia [25,26]. Additionally, the multi-task end-to-end optimized deep neural networks (MEON) has been used for oak and pine tree detection [27]. Durgut et al. [28] optimized tree detection by combining methods such as Swin Transformer [29], RCNN [30], Faster RCNN [31], YOLO [32], and DETR [33] using weighted box fusion, addressing the limitations of individual approaches. However, in complex scenarios such as tree overlap or occlusion, pixel-level segmentation methods often fail to achieve ideal results. Moreover, these methods cannot simultaneously detect tree positions and segment tree crowns. The instance segmentation methods have proven to be more effective than pixel-level segmentation methods for tree crown delineation. This is because they not only perform pixel-level segmentation but also identify and distinguish multiple individual instances, such as trees. There methods can be broadly classified into two-stage and single-stage models. Two-stage models, such as R-FCN [34] and Faster R-CNN [35], first generate candidate regions that potentially contain objects and then classify these regions while predicting their bounding boxes. These models typically achieve high detection accuracy but are constrained by slower inference speeds. In contrast, single-stage models, such as SSD [36] and YOLO [37], streamline the detection process by simultaneously predicting bounding boxes and class probabilities, offering faster inference at the cost of slightly reduced accuracy while maintaining good accuracy. Santos et al. [38] evaluated three object detection models for tree identification. Their findings revealed that the two-stage Faster R-CNN model, while achieving high detection accuracy, incurred the highest computational cost and the slowest inference speed. Furthermore, in tasks involving estimating the geographic distribution and identifying tree species, single-stage models have demonstrated superior performance in both detection accuracy and processing speed.

Recently, researchers extended object detection methods to perform pixel-level segmentation for individual tree crowns. This involves detecting all target instances within the input image and assigning pixel-level labels corresponding to each instance's category. Consequently, there has been a growing interest in leveraging YOLO models and other single-stage models for tree crown segmentation. These models offer faster execution speeds, facilitating the rapid completion of detection and segmentation tasks [39]. To improve detection speed and accuracy, the YOLO models [40] and its improvements [41–48], especially multi-scale detection, attention mechanisms, loss function optimization, and data augmentation [49–53], underwent various enhancements, making them highly

effective for tree crown detection. However, these models still face challenges in accurately capturing small-object details, managing interference from complex backgrounds, and accommodating the diverse characteristics of tree crowns. To obtain simultaneously the individual tree detection and tree crown segmentation results, researchers have explored the YOLO framework [54–57]. However, these approaches often face challenges arising from variations in tree crown color and texture, local lighting changes, and significant crown overlap. These factors can lead to uneven illumination within images, making it difficult to accurately locate tree tops and precisely delineate crown boundaries. Ultimately, these challenges can significantly impair the accuracy of crown delineation [58]. Consequently, a detailed and robust input data representation is crucial for the effective application of these methods, especially in dense forest environments.

To address the challenges of crown, overlap, and complex backgrounds in dense forests, a novel deep learning-based network, named CCD-YOLO, is proposed to segment individual trees with ALS point clouds. The key contributions are as follows:

(1) A new individual tree segmentation dataset is constructed, covering high-density areas, overlapping crowns, and complex backgrounds. The dataset consists of a multi-feature point cloud map derived from ALS point clouds, effectively capturing tree morphological features while reducing background interference.
(2) Integration of advanced modules, including the CReToNeXt backbone to enhance focus on critical regions, and CBAM attention mechanism to improve feature extraction efficiency and multi-scale feature fusion capabilities.
(3) A dynamic head is introduced to optimize feature layer weight and fusion strategies, improving the detection accuracy for target positions and boundary changes.

The main contribution of this study is to construct a novel model that can improve detection and segmentation accuracy in the complex forest, reducing both false positives and false negatives. Following the introduction, the materials are listed in Section 2, and the proposed CCD-YOLO and the new construct dataset are discussed in Section 3, The results are presented in Section 4, and a discussion is provided in Section 5, followed by conclusions.

2. Materials

2.1. Study Area

The study area (Figure 1) is located in the Epe Nature Reserve of Gelderland, Netherlands. It experiences a temperate maritime climate, characterized by mild and humid conditions. Annual precipitation typically ranges from 700 to 900 mm, and the average annual temperature falls between 9 and 11 °C. The topography of the region is characterized by gentle hills, with elevations ranging from 40 to 90 m. Forest cover typically ranges between 30% and 35%. The dominant tree species include Scots pine, Douglas fir, beech, and oak, forming a characteristic mixed forest ecosystem.

The study area is divided into Area 1 (***A1***) and Area 2 (***A2***), which can be used for the performance and generalization ability of the proposed model ***A1***, a larger and ecologically diverse region, that encompasses varying tree densities—ranging from sparse distributions to densely overlapping crowns—and a wide variety of tree species. ***A2***, located at a significant distance from ***A1***, is smaller but characterized by higher forest density, pronounced crown overlaps, and reduced species diversity, highlighting distinct ecological differences compared to ***A1***.

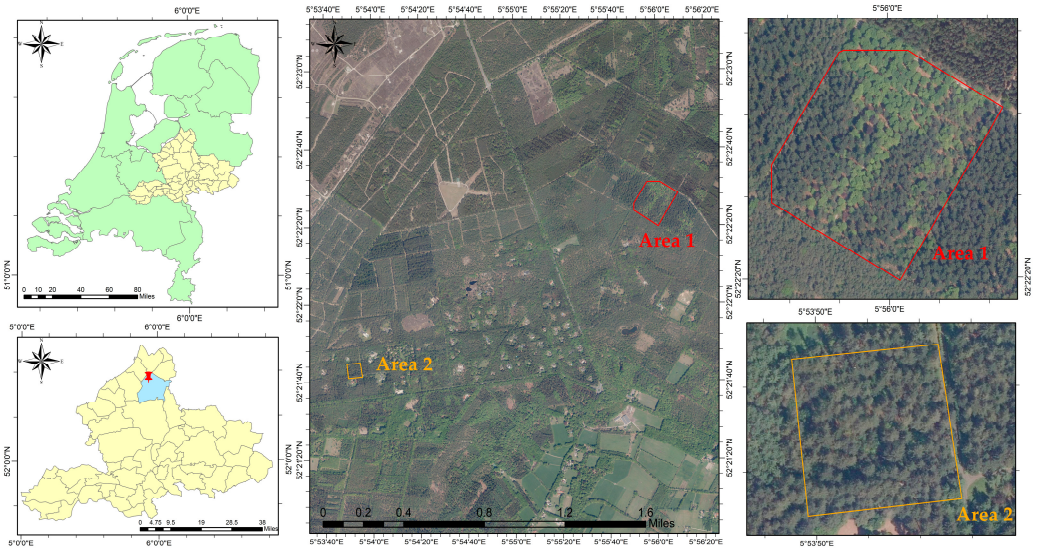


Figure 1. Overview of the study area. The true color image (right) of the study area captured by the drone, highlights the boundaries of Area 1 (in red) and Area 2 (in yellow). The blue region (Epe) in the bottom-left corner indicates the specific study area in Gelderland, while the map in the upper-left corner provides a scaled location map of the Netherlands.

2.2. Data Collection

The airborne LiDAR point cloud used in this study is part of the Netherlands geographic dataset and is publicly available from GeoTiles.nl (https://www.geotiles.nl/, accessed on 15 October 2023). It was collected on June 11, 2021, using the RIEGL VQ1560II LiDAR scanner in conjunction with high-precision DGPS and an inertial navigation system (IMU) to ensure data acquisition accuracy within 0.1 m. The point clouds have a density of 34.49 points per square meter, effectively capturing the intricate details of both terrain and vegetation. The point clouds, as illustrated in Figure 2, have been pre-processed and include RGB color information, making it easier to analyze and visualize the terrain features.

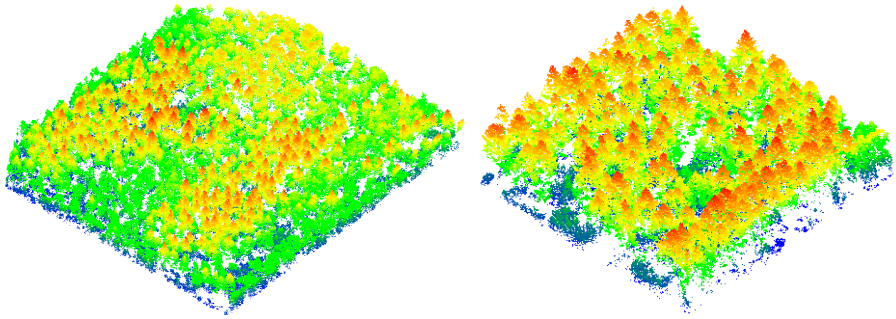


Figure 2. Visualization of ALS point clouds used in the study, colored by height.

3. Methods

This study aims to perform individual tree instance segmentation and object detection by the proposed CCD-YOLO; the pipeline is illustrated in Figure 3. It encompasses three key components that are individual tree segmentation (**ITS**) dataset generation (Section 3.1), CCD-YOLO model reconstruction (Section 3.2), and model training and accuracy evaluation (Section 3.3).

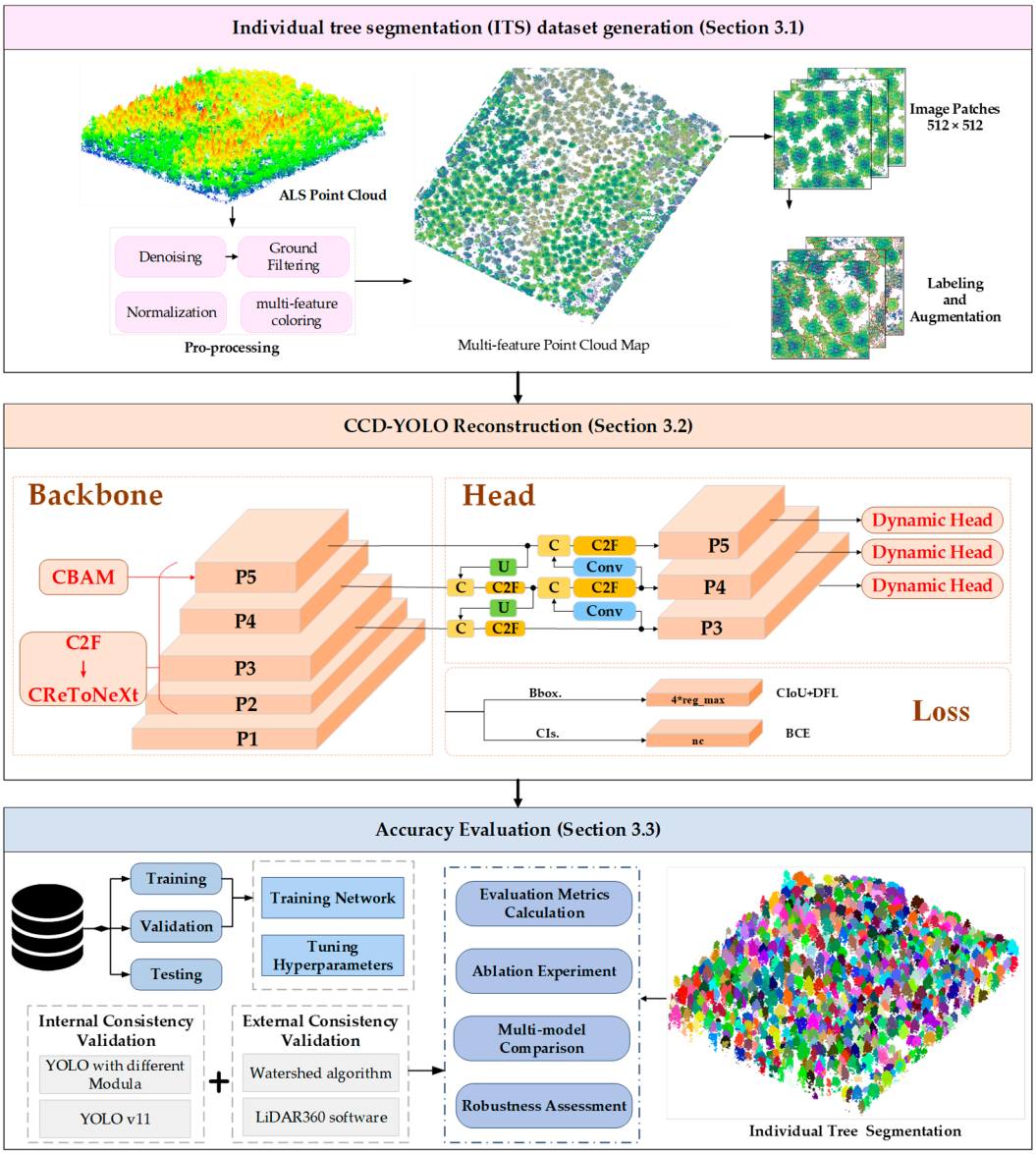


Figure 3. Workflow of the proposed approach using ALS point clouds.

3.1. Individual Tree Segmentation (ITS) Dataset Generation

The construction of the ITS dataset necessitates a series of processing steps for the downloaded point clouds, namely noise and outlier removal, extraction of non-ground point clouds, and the generation of multi-feature point cloud maps, as illustrated in Figure 3. For noise and outlier removal from the ALS point cloud, a statistical outlier removal (**SOR**) filtering [59] algorithm is applied. This algorithm identifies and removes points that deviate significantly from the local point distribution. To extract non-ground point clouds, the cloth simulation filtering (**CSF**) [60] algorithm implemented in an open-sourced software (CloudCompare, https://www.danielgm.net/cc/, accessed on 1 November 2023) is employed. This method optimizes the point cloud filtering process by simulating the physical properties of fabrics, thereby accurately extracting vegetation point clouds. In addition, a normalization operation [61] is applied to **CSF** results, which can remove the influence of terrain undulations.

For the multi-feature point cloud maps, these processed point clouds are vertically partitioned based on height, and the point density and intensity values are calculated for each height range. The height values assist in distinguishing trees at different vertical layers, while point density reflects the spatial distribution characteristics of the trees, and

point intensity provides detailed information about the surface structure of the trees. Based on these calculations, density-colored and intensity-colored point clouds are generated and visualized by color mapping. The RGB mean values of the two-colored point clouds are then fused to construct a multi-feature colored point cloud, enabling the extraction of comprehensive multidimensional information and enhancing the effectiveness and precision of feature extraction. Finally, these colored point clouds are mapped to the ground to generate a complete point cloud image that reflects the three-dimensional structural features, resulting in an individual tree segmentation (**ITS**) dataset.

These projected images, derived from the 3D point clouds, are further cropped into 512 × 512-pixel plots. To ensure continuity and consistency during subsequent processing, a 64-pixel overlap is maintained between adjacent plots, effectively preserving edge information.

Moreover, these images were annotated by LabelMe tool (https://github.com/wkentaro/labelme, accessed on 10 November 2023), generating a JSON file that contains tree position, anchor box size, and labels. To mitigate potential overfitting, data augmentation operations including flip-ping, rotation, contrast adjustment, and Gaussian noise addition were performed. The augmented dataset was randomly split into a training/validation dataset (70%) and a test dataset (30%). The training/validation was further divided into training data (80%) and validation data (20%) using a two-step training approach. By convention or without loss of generality, the dataset was constructed based on two different regions considering the diverse characteristics of the study area. A1 provides a controlled and diverse environment, divided into training, validation, and testing, which are used for model training, hyperparameter adjustment, and performance evaluation, respectively. A2, exhibiting certain environmental differences compared to A1, serves as a dedicated testing ground to assess the model's robustness and adaptability under challenging conditions and novel data distributions. This area division facilitates a comprehensive evaluation of the model's ability to generalize across different forest environments, ensuring its applicability and effectiveness in real-world scenarios. An example of a multi-feature point cloud map is shown in Figure 4.

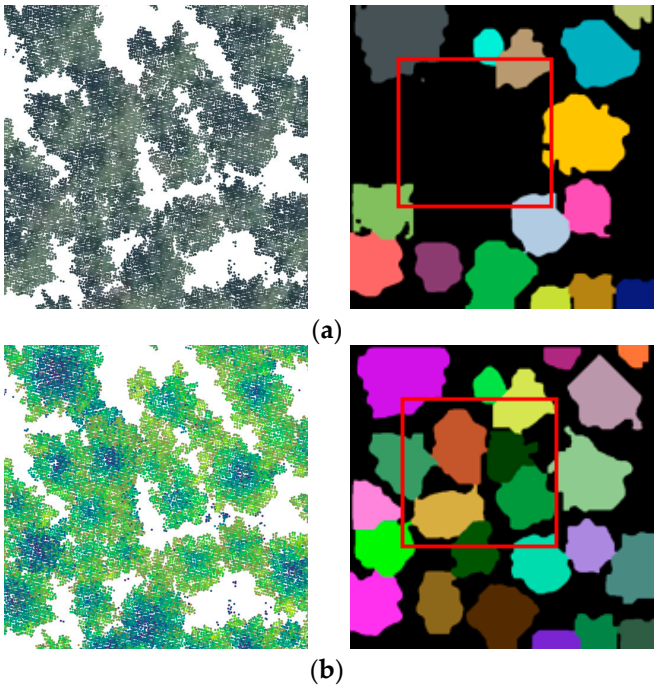


Figure 4. The labeled maps derived from the 3D point cloud and the original RGB image. (**a**) RGB point cloud map and mask result. (**b**) Multi-feature point cloud map and mask result.

3.2. CCD-YOLO Model Reconstruction

The proposed CCD-YOLO is built from YOLO v8, as illustrated in Figure 5, which effectively extracts tree locations and edge information within dense forested areas, where tree crowns frequently overlap and obscure each other. This model can be further enhanced by integrating a CReToNeXt module for improved feature extraction and multi-scale feature fusion, incorporating a convolutional block attention module (CBAM) to emphasize crown features and suppress background noise, and employing a dynamic head to enable adaptive multi-layer feature fusion.

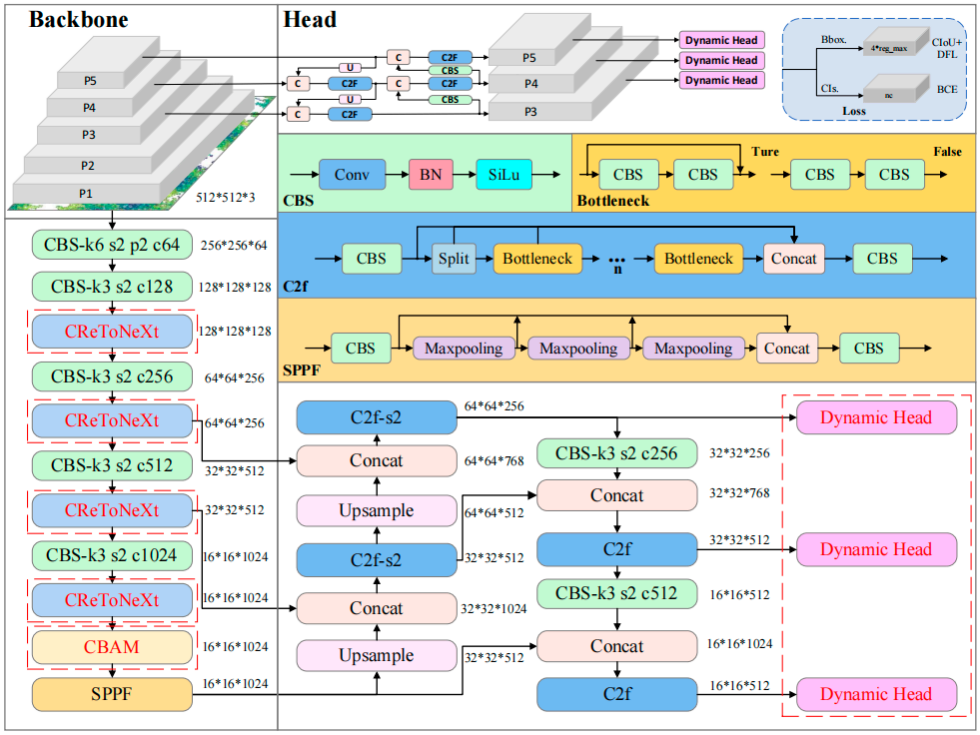


Figure 5. The proposed CCD-YOLO network architecture.

3.2.1. Improved Backbone with CReToNeXt

To enhance the efficiency of feature extraction and to capture multi-scale features and fine-grained details, this study introduces the CReToNeXt [62] module as a replacement for the original C2f module in the backbone network. CReToNeXt, an advanced deep learning module derived from Alibaba DAMO Academy's DAMO-YOLO model, integrates multiple innovative techniques. These enhancements significantly improve feature extraction capabilities and model efficiency. Its architecture incorporates depth-wise separable convolution (RepConv), residual connections, and multi-scale feature fusion mechanisms, enabling robust multi-level feature capturing. The RepConv reduces computational complexity while maintaining effective feature extraction, and the residual connections facilitate the flow of information across layers, improving training stability and enabling the network to learn deeper representations. In addition, the multi-scale feature fusion mechanisms can effectively capture features at different scales, enhancing the model's ability to detect objects of varying sizes.

As illustrated in Figure 6, the CReToNeXt module comprises two layers of CBSW modules integrated with the Swish activation function and three layers of basic block residual (BBR) modules. The input feature map first undergoes initial feature extraction within a CBSW structure. Subsequently, it is processed through a series of BBR modules. Within each BBR module, feature enhancement is achieved through the combined action of RepConv and residual connections. Feature maps that have undergone zero to three

BBR module operations are then concatenated. Finally, the concatenated feature map is fed into another CBSW structure for further feature extraction.

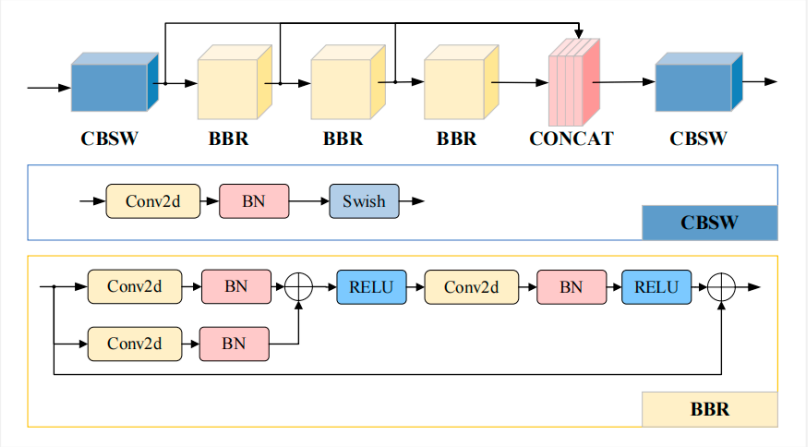


Figure 6. CReToNeXt module structure.

3.2.2. Improved Backbone with CBAM

To enhance focus on key regions and suppress irrelevant background information in dense forest environments, this study introduces the convolutional block attention module (CBAM) [63] into the backbone network. By refining the attention allocation, CBAM, as shown in Figure 7, emphasizes important features, ensuring that the model concentrates on target regions, thereby enhancing the precision of individual tree segmentation. It is a lightweight and efficient attention mechanism that adaptively enhances the feature map along both the channel and spatial dimensions, thereby improving the feature representation ability of the network.

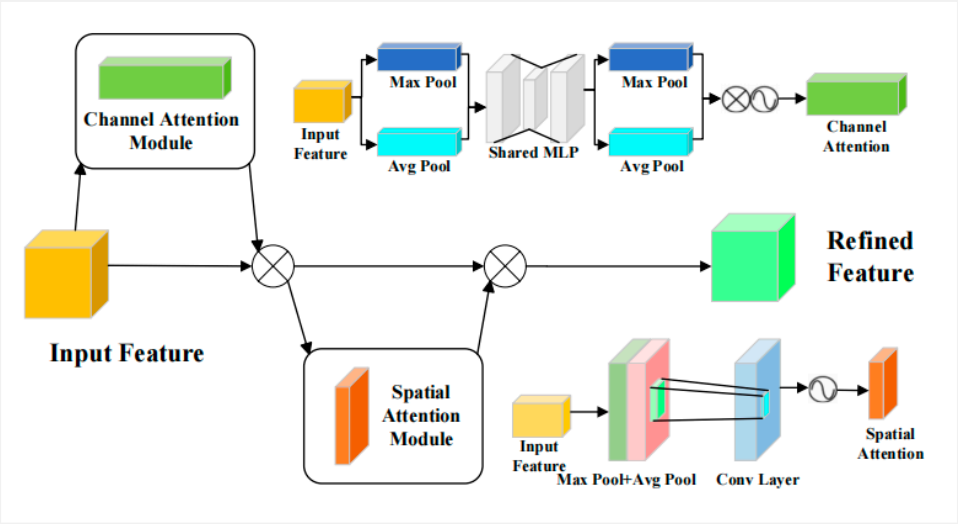


Figure 7. CBAM module structure.

As depicted in Figure 7, the CBAM module comprises two key components: the channel attention module (CAM) and the spatial attention module (SAM). In the CAM, two feature maps are generated by performing global maximum pooling and global average pooling on the input feature map. These two feature maps are then passed through a fully connected neural network, named multi-layer perceptron (*MLP*), which consists of one or more fully connected layers, followed by *ReLU* activation, batch normalization, and a sigmoid function to model the inter-channel dependencies effectively. These features are

then summed to produce the channel attention feature $M_C(F)$ through a sigmoid activation, as listed in Formula (1). It effectively emphasizes the target regions in the feature map, enhancing the model's ability to recognize relevant details:

$$M_C(F) = \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))), \quad (1)$$

In the spatial attention module (SAM), denoted by $M_S(F)$, average and maximum pooling are first applied to the input feature maps separately, producing two 1 × H × W feature maps. These feature maps are then concatenated along the channel dimension. Next, a 7 × 7 convolution is applied to the concatenated feature map to generate a single output feature map. Finally, a sigmoid activation function is applied to this output, yielding the spatial attention-oriented feature map. This process enhances the representation of spatial attention by emphasizing target locations and capturing detailed information effectively:

$$M_S(F) = \sigma(f^{7\times7}([AvgPool(F); MaxPool(F)])), \quad (2)$$

Finally, the feature maps generated by the channel attention module and the spatial attention module are combined into a single feature map, which is then multiplied element-wise with the original feature map. This process effectively reweights the original feature map, emphasizing target regions and significant features while suppressing irrelevant information and background noise. As a result, the model is better equipped to focus on task-relevant regions or channels within the image, resulting in improved performance in tree position detection and canopy boundary segmentation in dense forests.

3.2.3. Improved Detection Head with Dynamic Head

Due to the limited number of parameters in the prediction head, its expressive power is weak, making it difficult to fully exploit spatial information within the features, which limits the model's performance in multi-scale object detection. A dynamic head [64], as illustrated in Figure 8, was introduced, which can integrate contextual information and adjust the weight of feature layers for dense forests.

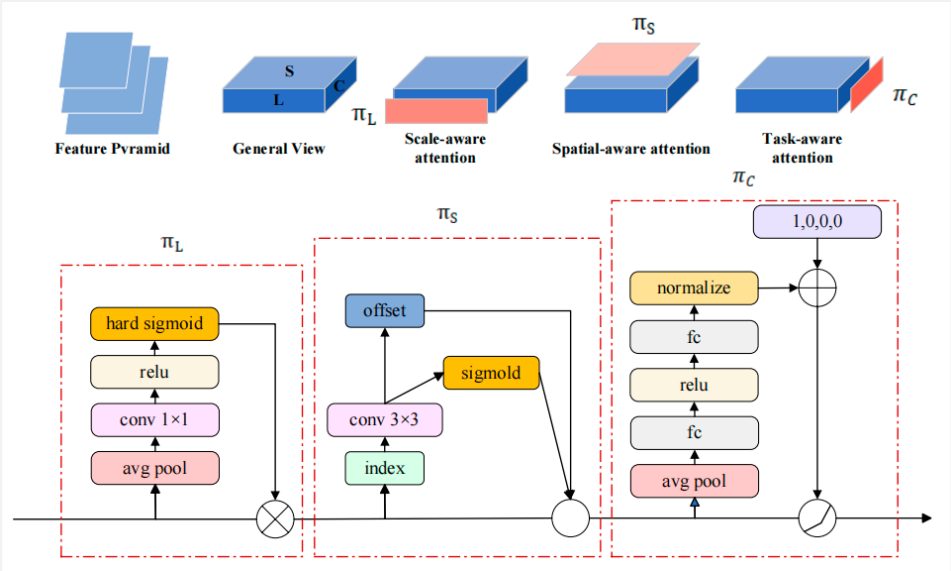


Figure 8. Dynamic head module structure.

The dynamic head module employs a self-attention mechanism to unify scale-aware, spatial-aware, and task-aware attention. This approach enhances the performance of the

model's object detection head without adding significant computational overhead. It integrates scale perception at the feature level, spatial perception at specific spatial positions, and inter-channel attention for task awareness. This attention mechanism is applied to the detecting head and can be stacked multiple times to enhance the model's performance. Given a three-dimensional feature tensor in the detection layer, the attention calculation is defined as follows:

$$W(F) = \pi_C(\pi_S(\pi_L(F) \cdot F) \cdot F) \cdot F, \quad (3)$$

where F represents a 3D tensor input L × S × C, π_L, π_S and π_C represents the scale-aware attention module, spatial-aware attention module, and task-aware attention module, respectively. They act only on dimensions ***L*** (level), ***S*** (space), ***C*** (channel) and the 3D tensor ***F***.

Scale-aware attention π_L (level-wise)—To solve the fusion problem of different scale features based on semantics, scale-aware attention is introduced:

$$\pi_L(F) \cdot F = \sigma(f(\tfrac{1}{SC}\textstyle\sum_{S,C} F)) \cdot F, \quad (4)$$

where $f(\cdot)$ corresponds to using a 1 × 1 convolutional approximation of linear functions, while $\sigma(\cdot)$ represents a hard ***S***-shaped activation function.

Spatial-aware attention π_S (spatial-wise)—This increases spatial awareness attention to highlight the ability to distinguish different spatial locations. Due to the large size of ***S***, it is decoupled into two stages—first, sparse attention learning is achieved through the use of deformable convolution, and then accomplished by integrating features at different scales:

$$\pi_S(F) \cdot F = \tfrac{1}{L}\textstyle\sum_{l=1}^{L}\sum_{k=1}^{K} w_{l,k} \cdot F(l; p_k + \Delta_{p_k}; c) \cdot \Delta_{m_k}, \quad (5)$$

where $p_k + \Delta_{p_k}$ is a shifted location by the self-learned spatial offset Δ_{p_k} to focus on a discriminative region, and K represents the count of sparsely selected positions. The remaining parameter details are similar to those in deformation convolution, which $w_{l,k}$ denotes a bias importance factor and Δ_{m_k} stands for an adaptive weighting importance factor, which is excluded here for conciseness.

Task-aware attention π_C (channel-wise)—To promote collaborative learning and enhance the scalability of target representation capabilities, and help completely different tasks by dynamically adjusting feature channels as needed:

$$\pi_C \cdot F = \max(\alpha^1(F) \cdot F_c + \beta^1(F), \alpha^2(F) \cdot F_c + \beta^2(F)), \quad (6)$$

As with ***DyReLU***, hyperparameters are essential for regulating activation thresholds. α and β were used for rescaling and reorienting, respectively. Multiple instances of the previously mentioned attention mechanism can be stacked by applying it successively

3.3. Accuracy Evaluation

To evaluate the performance of the model, this study comprehensively adopts metrics such as precision, recall, F1-score, and average precision (AP) for both object detection and instance segmentation in dense forests [65]. Precision is the proportion of correct positive predictions out of all predicted positives. Recall is the proportion of correct positive predictions out of all actual positives. The F1-score is the harmonic mean of precision and recall, offering a balanced evaluation. AP represents the average precision calculated over different recall rates across varying levels of confidence thresholds equivalent to the area of the precision–recall curve. In particular, AP@0.5 is computed at an intersection over the

union (IoU) threshold of 0.5. Higher values of these metrics indicate superior model performance. These metrics are as follows:

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{8}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{9}$$

$$AP = \int_0^1 P(R)dR \tag{10}$$

where ***TP*** (true positive) denotes the number of correctly detected (segmented) trees. ***FP*** (false positive) and ***FN*** (false negative) represent the number of incorrectly and missed detected (segmented) trees. ***P*** and ***R*** refer to precision and recall, respectively.

4. Results

A novel dataset was created, and the proposed method was primarily evaluated on two distinct airborne laser scanning (***ALS***) datasets acquired from dense forest environments. The performance of individual tree segmentation was rigorously assessed using a suite of internal consistency metrics, proving the effectiveness of the proposed approach.

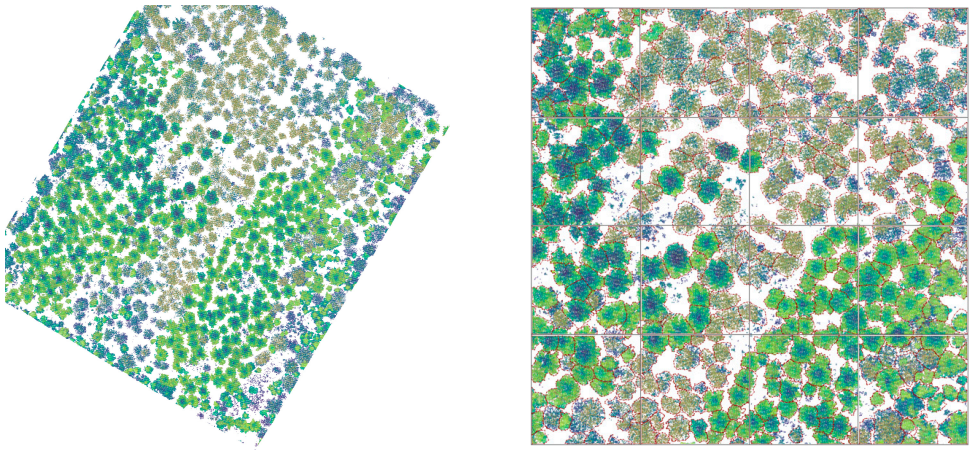
4.1. Results of the ITS Dataset

A comprehensive individual tree segmentation dataset, derived from the ALS point clouds, is constructed for the proposed method. It can effectively mitigate interference from solar radiation variations and phenological texture features while preserving crown morphological characteristics. The details of the ITS datasets are listed in Table 1.

Table 1. The description of ITS dataset.

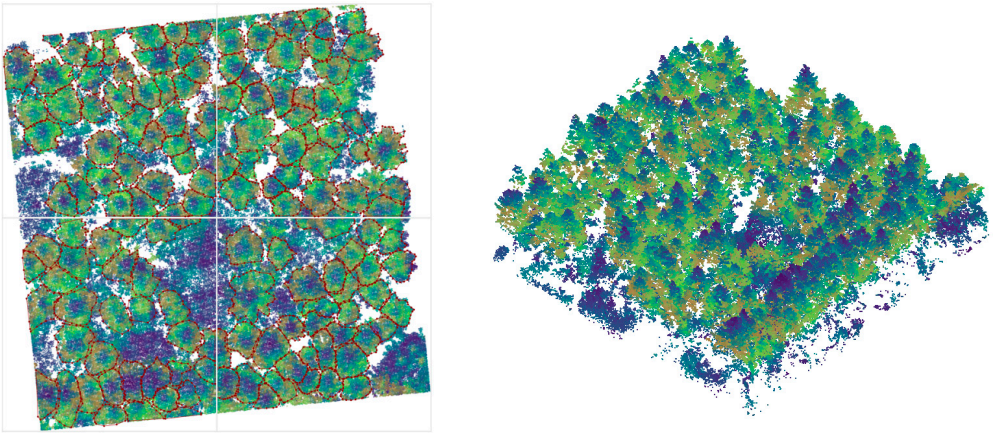
Area	Usage	Image Size (Pixel)	Number of Images	Number of Trees
A1	Training	512 × 512	96	1901
	Validation	512 × 512	24	582
	Test 1	512 × 512	52	1071
A2	Test 2	512 × 512	4	144

These generated new datasets are in two parts, located in various regions. The one named ***A1*** contains 173 images and is used for CCD-YOLO training and validation, while the ***A2*** dataset is only used for CCD-YOLO testing. Moreover, the constructed dataset is visualized in Figures 9 and 10.



(**a**) (**b**)

Figure 9. Visualization of the ITS dataset (parts of ***A1***). (**a**) Multi-feature point cloud map. (**b**) ITS dataset with labeled mark (***A1***).



(**a**) (**b**)

Figure 10. Visualization of the ITS dataset (***A2***), and its corresponding 3D point cloud. (**a**) ITS dataset with labeled mark (***A2***). (**b**) 3D tree point cloud.

4.2. Results of Individual Tree Segmentation

The proposed CCD-YOLO model is built upon the YOLOv8 framework, leveraging PyTorch 1.12 and an NVIDIA GeForce RTX 3080 for training and inference. For model training, we employed the Adam optimizer with the following default hyperparameters: momentum of 0.937, an initial learning rate of 0.01, and weight decay of 0.0005. The training process was conducted over 300 epochs. The obtained segmentation results by the proposed approach are illustrated in Figure 11, where each tree is visually distinguished by assigning it a unique random color.

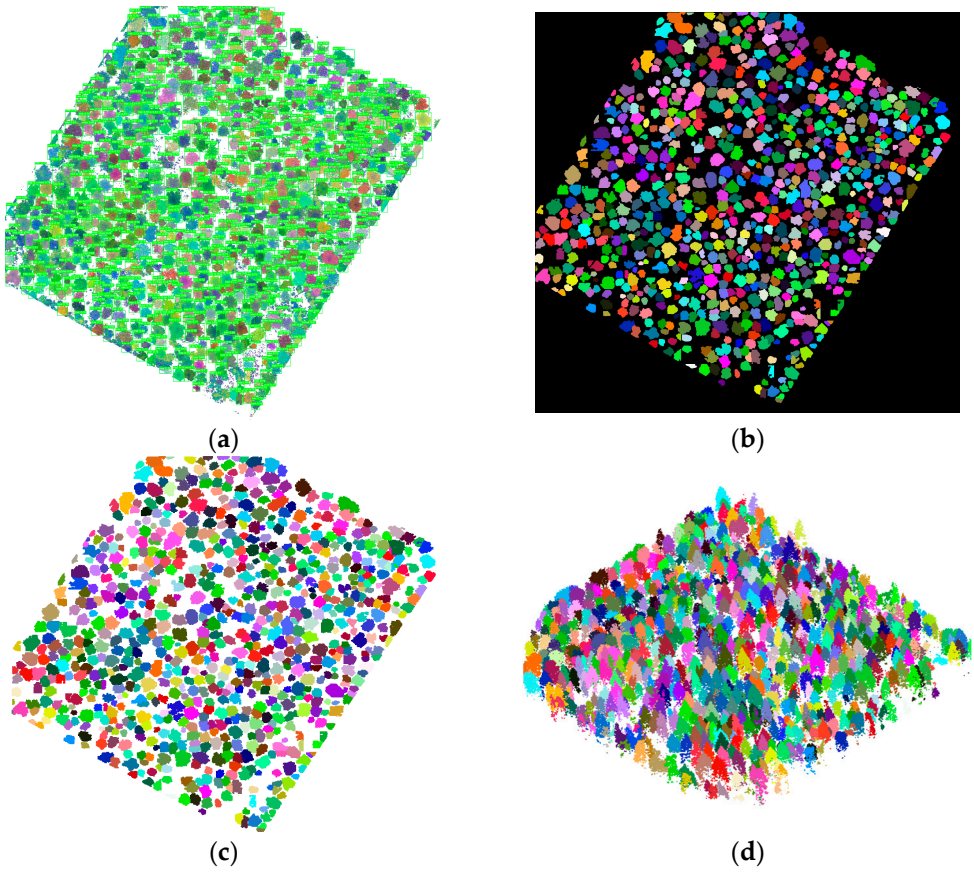
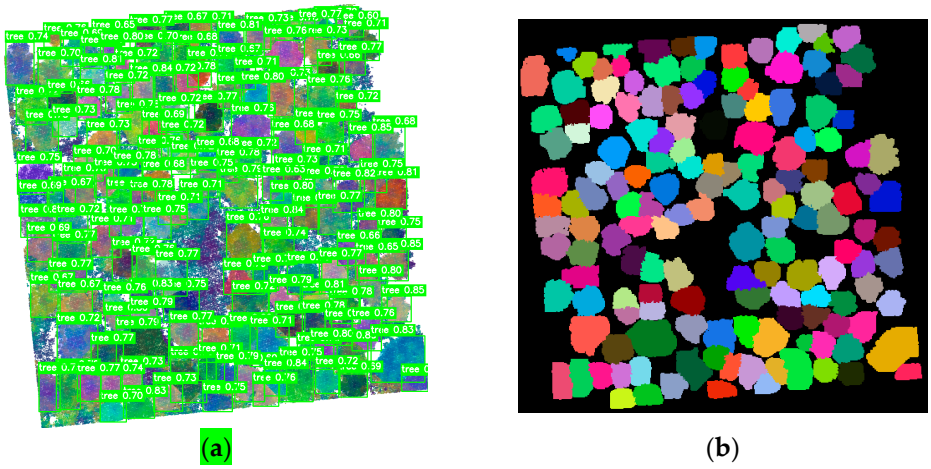
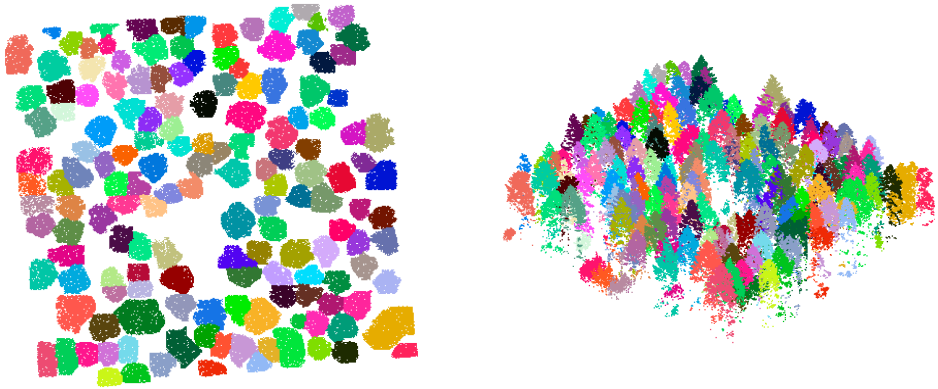


Figure 11. View of individual tree segmentation results in *A*1. (**a**) Prediction box result. (**b**) Mask result. (**c**) Top view of the segmentation result. (**d**) Oblique view of segmentation result.

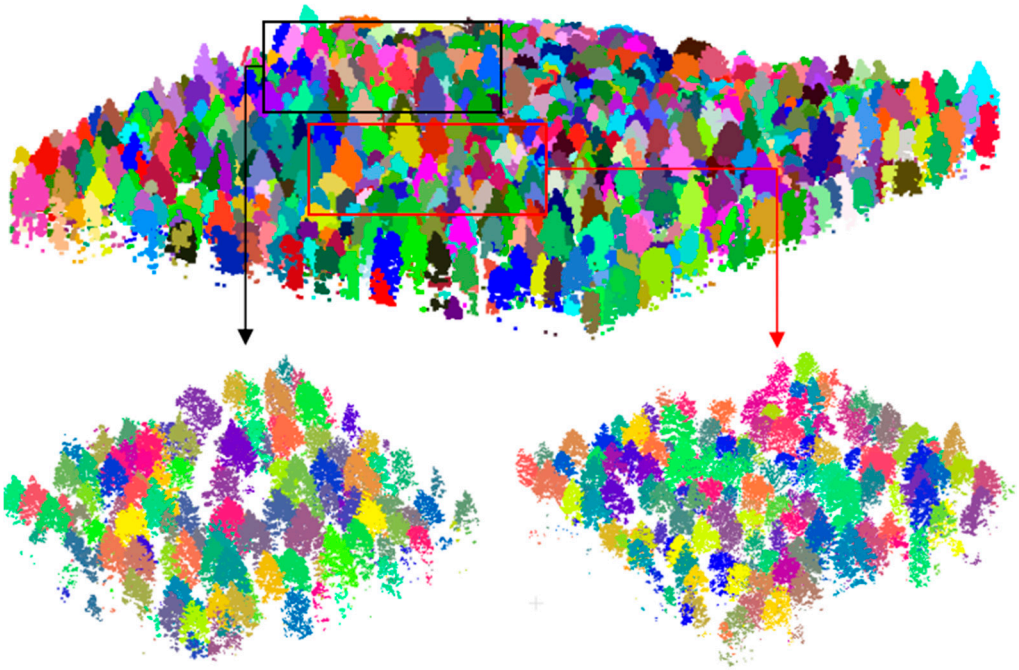
It can be seen from Figures 11 and 12 that the neighboring trees are rendered in distinct colors, clearly defining boundaries between them. This indicates the algorithm's ability to accurately distinguish and segment individual trees, resulting in point clouds that closely reflect their natural morphology. Moreover, the method effectively preserves the spatial structure of each tree while avoiding over-segmentation and under-segmentation. This ensures the successful extraction of complete individual tree point clouds. In addition, the results of individual tree segmentation across varying stand densities, ranging from low to medium and high tree density, are presented in Figure 13.





(**c**) (**d**)

Figure 12. View of individual tree segmentation results in *A*2. (**a**) Prediction box result. (**b**) Mask result. (**c**) Top view of the segmentation result. (**d**) Oblique view of segmentation result.



(**a**) (**b**)

Figure 13. Results of individual tree segmentation in low to medium tree density and high tree density regions. Extraction of low to medium tree density and high tree density region in ***A1***. (**a**) Low to medium tree density region. (**b**) High tree density region.

What is more, we visualized these individual tree segmentation results from the regions in various tree densities, as shown in Figure 14.

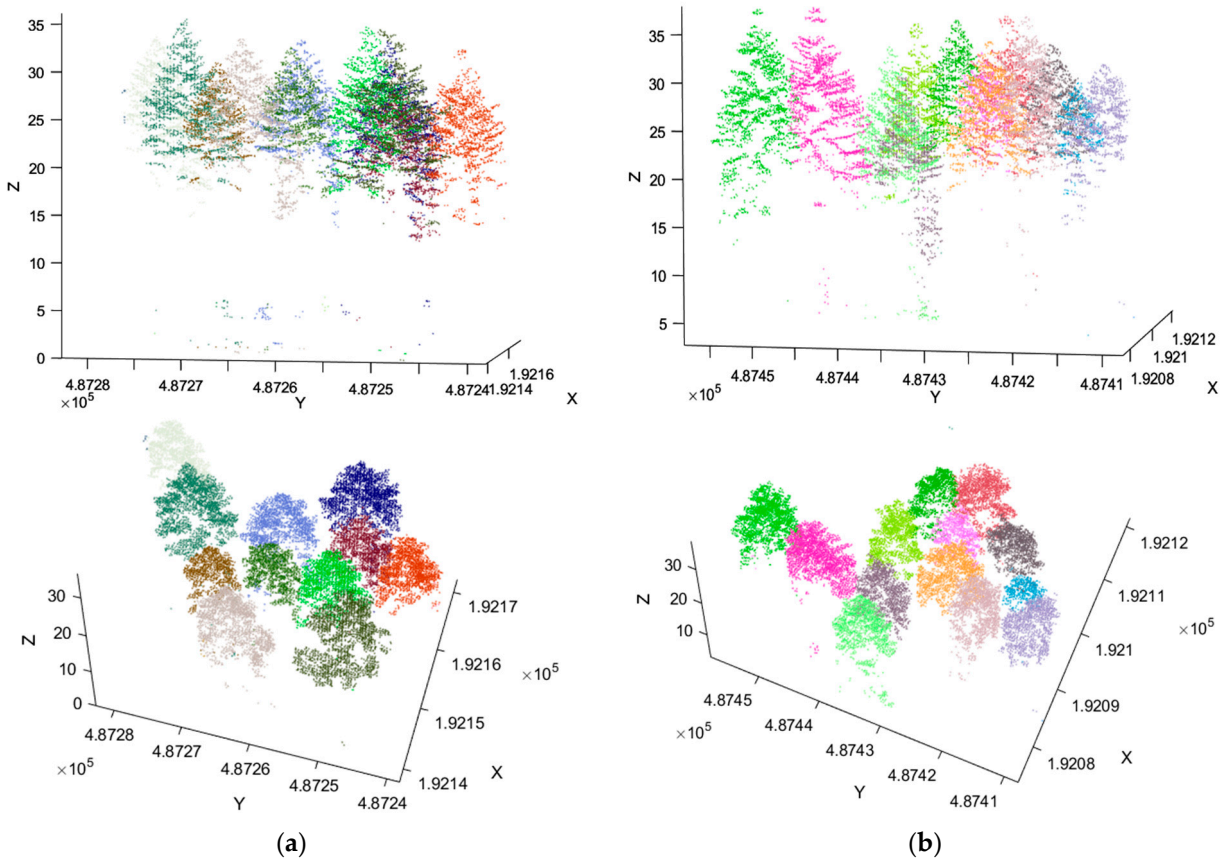


Figure 14. Visualization of the individual tree segmentation results across various densities. (**a**) Low to medium tree density region. (**b**) High tree density region.

4.3. Results of Accuracy Evaluation

To assess the effectiveness and feasibility of the proposed CCD-YOLO model, we conducted an accuracy evaluation to analyze the impact of varying environmental densities in different regions on the network's performance. The quantitative results are presented in Table 2.

Table 2. Results of CCD-YOLO between different stages.

Testing	Actual Trees	Tree Crown Detection			Tree Boundary Segmentation		
		TP	FP	FN	TP	FP	FN
A1	1071	677	157	237	675	144	252
		63.2%	14.7%	22.1%	63.1%	13.4%	23.5%
High density in ***A1***	143	83	23	37	85	22	36
		58.0%	16.1%	25.9%	59.4%	15.4%	25.2%
Low density in ***A2***	121	87	11	23	87	13	21
		71.9%	9.1%	19.0%	71.9%	10.7%	17.4%
A2	144	106	18	20	106	17	21
		73.6%	12.5%	13.9%	73.6%	11.8%	14.6%

For the tree extraction, the proposed CCD-YOLO will simultaneously work on tree crown detection and boundary segmentation. As can be seen from Table 2, there are 1071 and 144 trees in the dataset ***A1*** and ***A2***, respectively, and the fully extracted trees are 677 (63.2%) and 106 (73.6%). Furthermore, these improvements for CCD-YOLO can be adapted for both two-stage detectors and different forest densities. A visual map for these assessment metrics is illustrated in Figure 15.

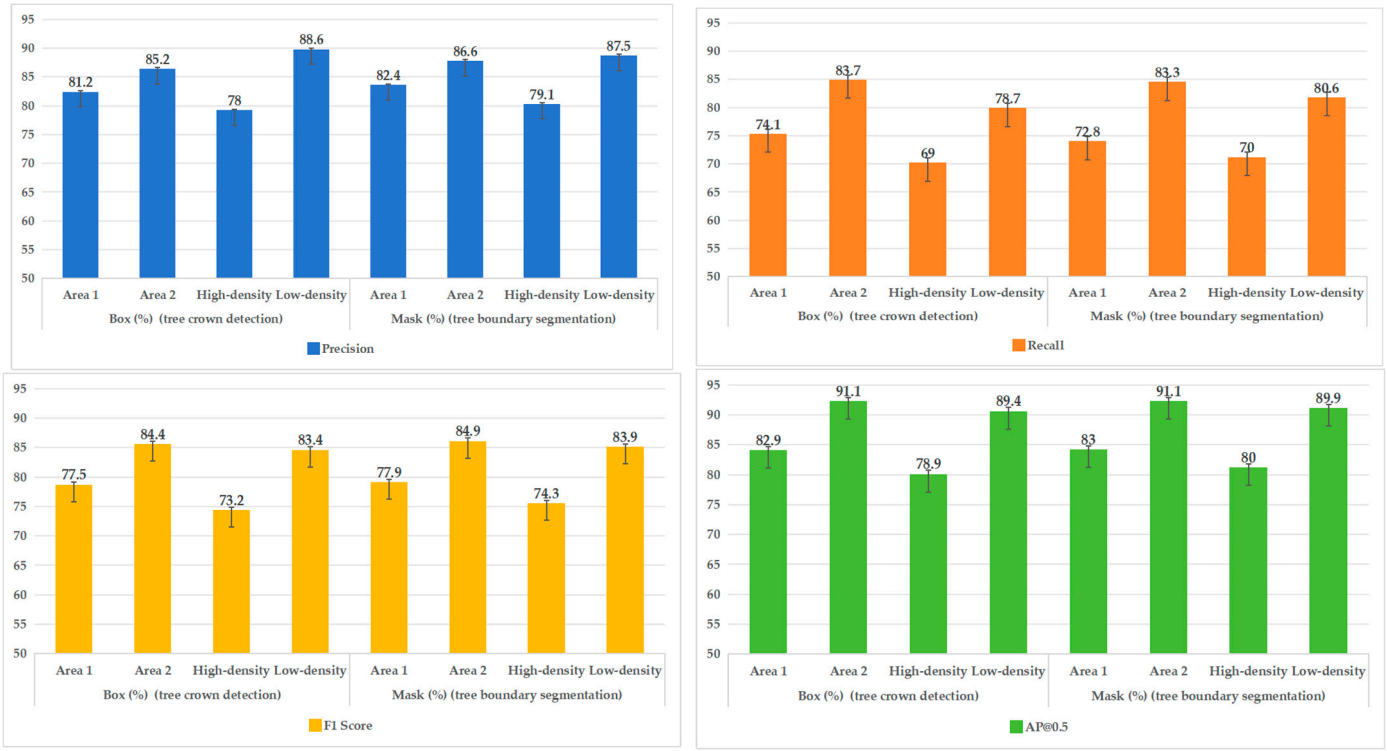


Figure 15. Comparison of evaluation metrics of different stage.

As shown in Table 2 and Figure 15, the model's performance in detecting and segmenting trees varies significantly across regions with different tree densities. The model performs well in simpler environments (such as low density), achieving high detection and segmentation accuracy with low false detection and missed detection rates. Benefiting from the large spacing between trees and minimal crown overlap, the model effectively minimizes FP and FN, accurately locating individual trees and precisely segmenting crown boundaries. However, in complex environments (such as ***A1*** and high density), trees grow closer together, their canopies overlap and block each other, and background noise further complicates the task, making it harder to tell individual trees apart. This makes it more difficult for the model to accurately detect and separate individual trees. As a result, the model misses more trees, leading to a lower recall rate and increased FP and FN values. The results indicate that as tree density increases, the complexity of the forest environment rises, leading to a decrease in the model's segmentation performance.

In addition, ***A1***, the main study site, is a large area with a mix of forests, from sparse to dense. The model's performance in this area was average. ***A2*** is a separate test site far away, with a moderate number of trees and fewer overlapping tree canopies. The fact that the model performed well in ***A2***, which is different from ***A1***, indicates that it can adapt to new environments, or generalize well.

Variations in environmental factors can lead to differences in tree growth density and forest vegetation complexity. In dense forests, overlapping tree canopies may affect the accuracy of extracting tree crown boundaries and the positions to some extent. By providing diverse environmental settings and conducting experiments in two distinct regions, the effectiveness of the proposed method can be further validated. The experimental results demonstrate that the method remains effective across different environmental conditions, highlighting its robustness and applicability.

5. Discussion

5.1. Comparison with Commonly Used Approaches

To validate the accuracy and reliability of the proposed approach for individual tree segmentation, this study compared it with commonly used single-tree segmentation methods, including the watershed algorithm [66] and the commercially available LiDAR360 software (https://www.lidar360.com/, accessed on 15 September 2024) [8]. We compared these methods across low-to-medium tree-density regions, high-tree-density regions, and the overall study area. The metrics for these three methods are the precision, recall, and F1 score, as illustrated in Figure 16.

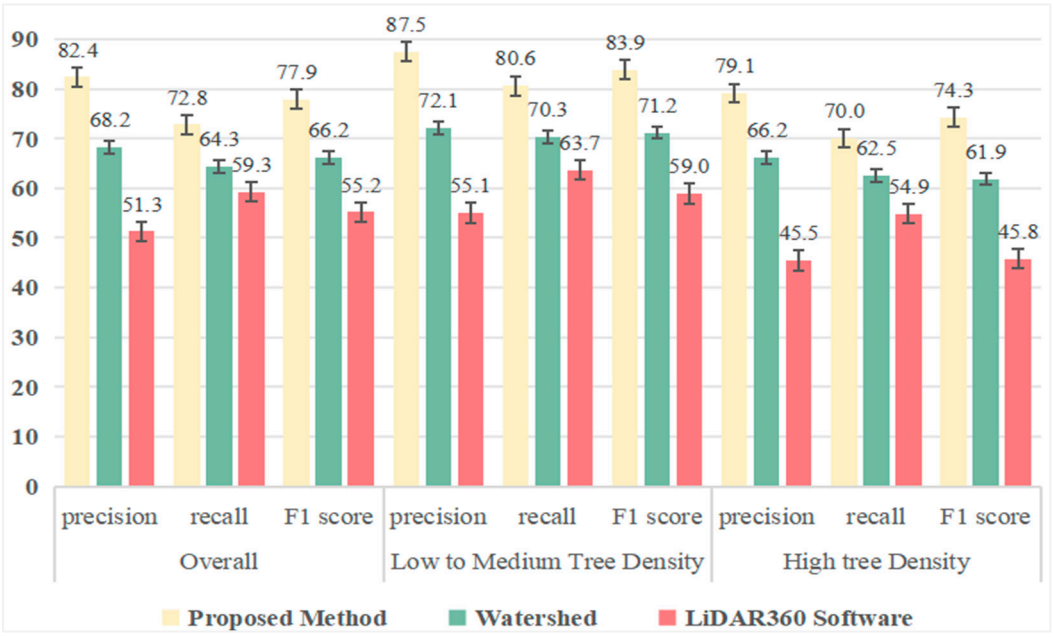


Figure 16. Comparison of the segmentation performance of individual trees.

Seen from Figure 16, the proposed CCD-YOLO method exhibits superior performance in detection and segmentation tasks, showing good generalization ability and stability in all the testing experiments. It significantly outperforms watershed (F1 score = 66.2%) and LiDAR360 (F1 score = 55.2%) in overall performance, achieving the highest precision (82.4%), recall (72.8%), and F1 score (77.9%). It can be seen that the proposed CCD-YOLO has achieved the highest score in areas with various forest densities, and can accurately locate tree positions and segment crown boundaries, preserving the complete and natural shape of tree crowns. In low-to-medium tree density regions, CCD-YOLO demonstrated exceptional performance, achieving a precision of 87.5% and an F1 score of 83.9%. Even in areas with severe crown overlap and occlusion, the proposed model demonstrates strong robustness, achieving an F1 score of 74.3%, respectively, which effectively reduces false positives and false negatives, addressing the issue of blurred boundaries. However, watershed was limited by its performance in dense areas, with a significant drop in F1 score from 71.2% to 61.9% due to issues in seed point extraction and boundary delineation. LiDAR360 consistently underperformed, especially in dense areas (F1 = 45.8%), failing to effectively address crown overlap and occlusion. The method often resulted in incomplete crown shapes or missed trees due to incorrect merging or omission of tree point clouds.

Moreover, visualization comparison results are shown in Figure 17.

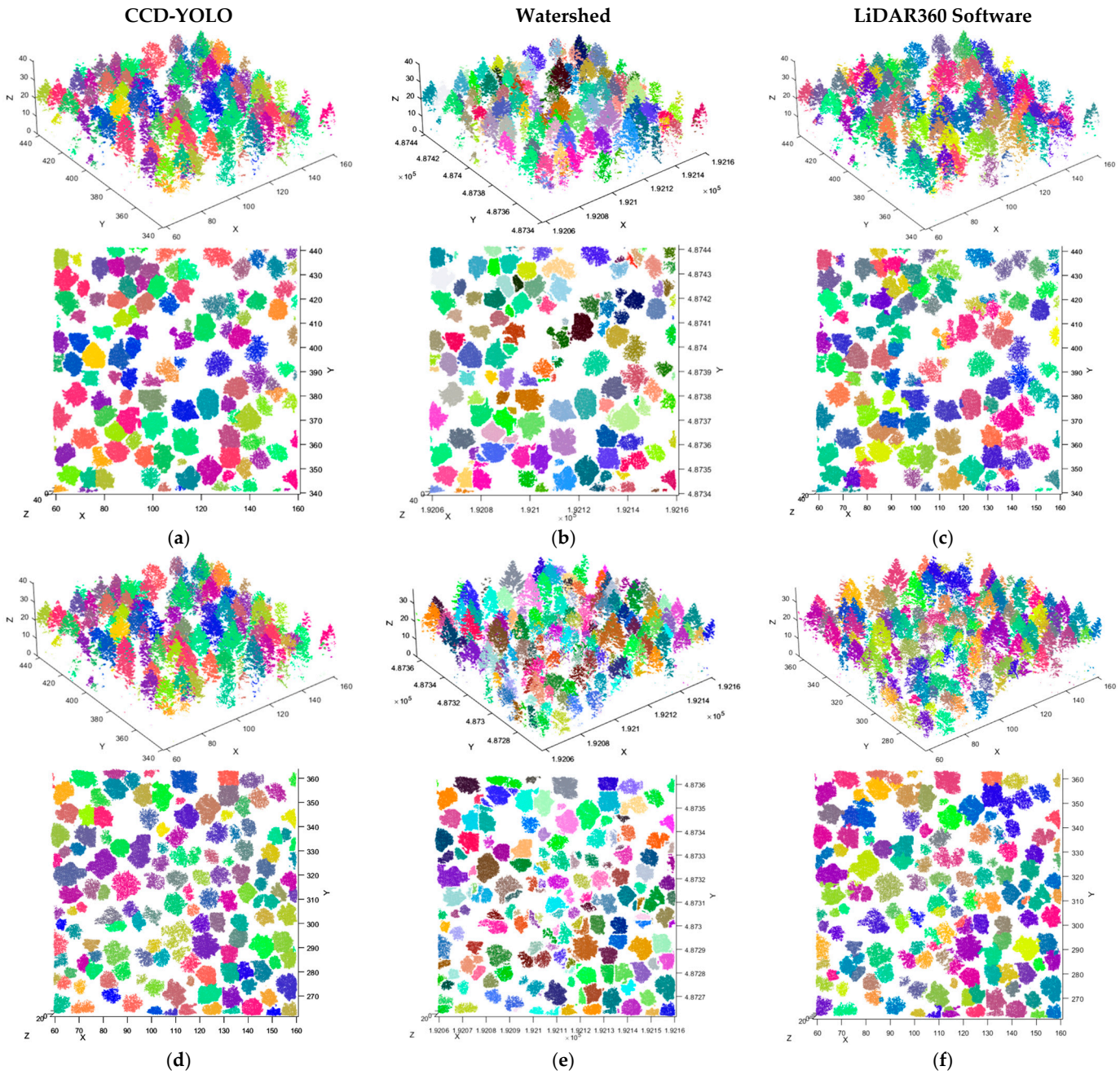


Figure 17. Comparison of individual tree segmentation results in various forest densities. (**a**–**c**) The oblique and top views of the individual tree segmentation results obtained by the three methods in regions with low to medium tree density; (**d**–**f**) The oblique and top views of the individual tree segmentation results obtained by three methods in regions with high tree density. (**a**) Results (oblique and top view) in forests with low-medium density using CCD-YOLO. (**b**) Results (oblique and top view) in forests with low-medium density using watershed. (**c**) Results (oblique and top view) in forests with low-medium density using LiDAR360 software. (**d**) Results (oblique and top view) in forests with high density using CCD-YOLO. (**e**) Results (oblique and top view) in forests with high density using watershed. (**f**) Results (oblique and top view) in forests with high density using LiDAR360 software.

5.2. The Effectiveness of the Introduced Modules

To evaluate the efficacy and feasibility of the proposed CCD-YOLO, we conducted multiple repeated ablation experiments to carefully examine the impact of different components, as listed in Table 3.

Table 3. Details of CCD-YOLO with different modules.

Basic Models	New Modules			Box (%) (For Tree Crown Detection)				Mask (%) (For Tree Boundary Segmentation)				FPS
	CReToNeXt	CBAM	Dynamic Head	Precision	Recall	F1 Score	AP@0.5	Precision	Recall	F1 Score	AP@0.5	
YOLO V8				78.9	70.9	74.7	80.0	77.9	71.8	74.7	79.9	85
	√			80.5	74.0	77.1	82.0	80.9	75.4	78.1	82.4	68
		√		77.5	66.5	71.6	76.3	79.9	68.3	73.6	78.9	84
			√	77.2	68.6	72.6	78.7	79.0	70.3	74.4	79.7	97
	√	√		80.1	71.1	75.3	80.4	80.7	72.9	76.6	81.2	67
	√		√	80.2	72.2	76.0	81.2	80.9	74.2	77.4	82.3	64
		√	√	75.4	69.5	72.3	77.1	77.8	71.0	74.2	79.3	83
Proposed	√	√	√	**81.2**	**74.1**	**77.5**	**82.9**	**82.4**	**72.8**	**77.9**	**83.0**	**64**
YOLO V11				79.0	72.8	75.8	80.1	81.0	74.5	77.6	82.3	79

As can be seen from Table 3, the individual contributions of the CReToNeXt, CBAM, and dynamic head modules, coupled with their synergistic interactions during joint optimization, are crucial for achieving the highest accuracy in tree crown detection and tree boundary segmentation, significantly enhancing the performance of the original YOLO model.

The CReToNeXt module replaces the original C2f module within the YOLO backbone. By employing re-parameterized convolutions and a residual connection, CReToNeXt significantly enhances feature extraction efficiency and multi-scale fusion. In object detection tasks, precision, recall, and F1-score improved by 1.6%, 3.1%, and 2.4%, respectively, demonstrating a substantial improvement in target localization and resulting in fewer missed and false detections. For segmentation, precision and recall increased by 3.0% and 3.6%, respectively, highlighting the model's superior ability to accurately capture tree crown regions and delineate their boundaries. However, this improvement comes with a reduction in inference speed, as the ***FPS*** decreased from 85 (baseline YOLOv8) to 68 due to the added complexity of re-parameterized convolutions.

Integrating the CBAM attention mechanism effectively focuses the model on critical regions, emphasizing features relevant to tree locations and crown edges while suppressing background noise. In segmentation tasks, this significantly improves feature extraction accuracy, with recall increasing by 3.4% and the F1 score by 2.1%. This optimization effectively enhances the segmentation of tree crown regions and demonstrates excellent performance in detail handling and feature extraction for key areas. Moreover, the CBAM module maintains a competitive inference speed of 84 ***FPS***, demonstrating minimal computational overhead while improving performance.

Furthermore, the model incorporates a dynamic head in its detection head, replacing the traditional decoupled head structure. This module enables the adaptive fusion of multi-layer features, enhancing the model's adaptability to subtle boundary variations and improving the precision of tree crown boundary delineation. In segmentation tasks, the dynamic head optimizes feature representation and boundary capture, resulting in a 1.5% improvement in recall and a 0.3% increase in the F1 score. This effectively strengthens the model's ability to handle target region edges with greater accuracy. The inference speed for this module is 97 ***FPS***, reflecting a slight increase compared to the baseline YOLO v8, while demonstrating its computational efficiency in achieving high performance for boundary detection.

Combining the CReToNeXt module with the CBAM attention mechanism, the CReToNeXt module with the dynamic head, or the CBAM attention mechanism with the dynamic head yields significant synergistic improvements. Notably, the combination of the CReToNeXt module and the dynamic head delivers the most outstanding performance in instance segmentation, achieving a 3.0% increase in precision, a 2.4% increase in recall,

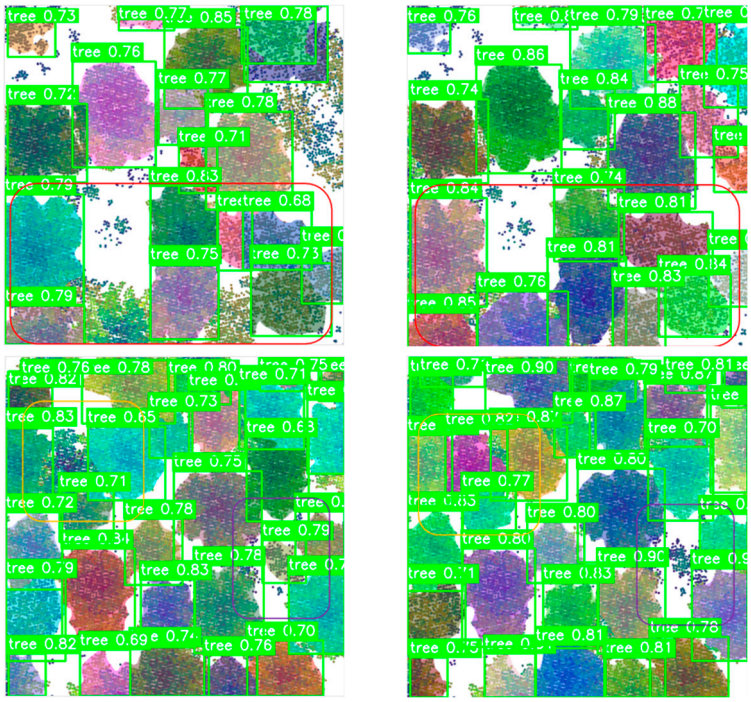
and a 2.7% increase in the F1 score. This combination significantly enhances the model's ability to delineate tree crown boundaries and effectively reduces both omission and commission errors. These results highlight the synergistic effects of integrating these two modules, demonstrating their collective potential to enhance the detection performance of the original YOLO. However, the inference speed drops to 74 ***FPS***, which is inferior to the other two combinations, highlighting the trade-off between accuracy and computational cost.

When the CReToNeXt, CBAM, and dynamic head modules are integrated simultaneously into the YOLO model, their synergistic effects lead to significant performance improvements. Compared to the baseline YOLO model, object detection precision increased by 3.4%, recall by 4.0%, and the F1 score by 4.3%. In instance segmentation, precision improved by 3.5%, recall by 4.4%, and the F1 score by 4.1%. Despite these substantial performance gains, the proposed CCD-YOLO model maintains an acceptable inference speed of 64 ***FPS***, balancing accuracy improvements with computational efficiency.

In addition, we compared the improved network with YOLOv11 to evaluate its performance enhancements. YOLOv11 [67] represents the latest advancement in the Ultralytics YOLO series, building upon and improving YOLOv8. Compared to YOLOv8, the model replaces the C2f module with C3K2 and incorporates a C2PSA module after the SPPF layer to enhance feature representation capabilities. Additionally, the detection head integrates two DWConv layers, and the model's width and depth parameters have been significantly adjusted, resulting in improved detection accuracy and inference efficiency. Compared to YOLOv11, CCD-YOLO demonstrates significant advantages in detection accuracy. Notably, in the box category, precision, F1 score, and AP@0.5 are improved by 2.2%, 1.7%, and 2.8%, respectively. Additionally, in the mask category, AP@0.5 shows an improvement of 0.7%. These advancements highlight that CCD-YOLO delivers higher accuracy and reliability in object detection and segmentation tasks. Although CCD-YOLO achieves an inference speed of 64 ***FPS***, slightly lower than YOLOv11's 72 ***FPS***, it prioritizes accuracy, making it particularly suitable for accuracy-critical applications such as ecological studies that require detailed tree crown segmentation and precise boundary detection. In scenarios where speed is critical, like real-time monitoring, adjustments to reduce computational complexity could enhance CCD-YOLO's applicability.

These results demonstrate that integrating these three enhanced modules significantly strengthens the original YOLO model's detection and segmentation capabilities. The model's performance progressively improved with the addition of each module, achieving optimal performance after their combined optimization. The combination of these three modules substantially enhances the model's ability to accurately capture tree positions and boundaries, leading to improved detection precision and segmentation performance while reducing both omission and commission errors. This highlights the model's increased robustness and adaptability.

Figure 18 presents the local results obtained using different methods. From the figure, it is evident that the visualization results of the improved model show significant enhancements compared to the original network. The CReToNeXt module improves feature extraction and multi-scale fusion capabilities, enabling more accurate tree position and shape segmentation. The CBAM attention mechanism suppresses background noise and highlights tree crown boundaries, resulting in clearer boundary segmentation. The dynamic head module adaptively fuses multi-layer features, dynamically optimizing boundary processing to achieve more refined tree crown delineation. The synergy of these modules significantly enhances both the overall quality and the detailed performance of the segmentation results.



(**a**) (**b**)

Figure 18. Results of individual tree segmentation between YOLO v8 and CCD-YOLO. (**a**) YOLO V8 model. (**b**) The proposed CCD-YOLO.

5.3. Application of Tree Segmentation in Forest Management

The precise individual tree segmentation results obtained from the proposed model provide robust support for in-depth forest resource exploration. These results establish a reliable foundation for extracting structural parameters at the individual tree level, such as tree height, crown width, and crown projection area. These parameters not only reveal the microstructural characteristics of forests but also provide critical data for ecological studies and forest resource management at a macro scale. Their applications include forest health assessment, biomass and carbon stock estimation, and precision forestry management. The specific methodological workflow is illustrated in Figure 19, taking tree height, crown width, and crown volume as examples to demonstrate the key steps and logical framework for parameter extraction.

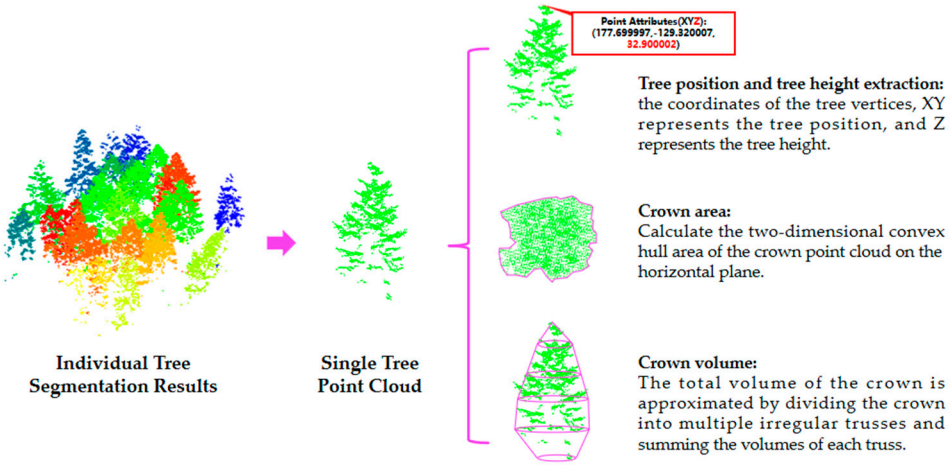


Figure 19. Framework for extracting individual tree structural parameters.

5.4. Limitation

The proposed method, by combining different modules, emphasizes the characterization ability of multi-scale tree morphological features and achieves excellent results in various aspects. However, the model exhibits certain limitations in practical applications, particularly in densely forested areas with complex backgrounds. Challenges such as crown overlap, occlusion, and noise interference can lead to segmentation inaccuracies in these regions. As illustrated in Figure 20, the red-highlighted areas in the left panel indicate instances of erroneous segmentation, while the right panel presents a detailed visualization of these errors.

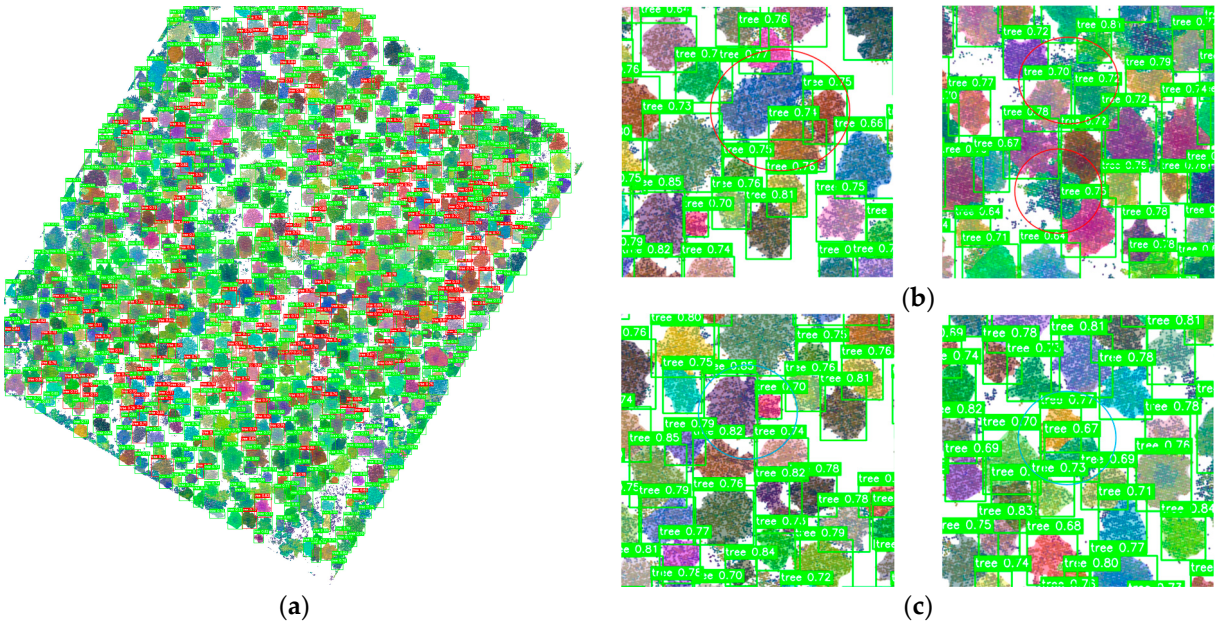


Figure 20. Results of the erroneous segmentation. (**a**) Erroneous segmentation. (**b**) Missed detection. (**c**) False detection.

Areas with high-density, severe crown overlaps and occlusion lead to blurred crown boundaries, causing some adjacent trees to be easily identified as a single crown object. Thus, some failures were observed in the testing areas. The main reason for these failed examples involves not only the misclassification of shrubs (non-trees) but also the detection of a single large crown as multiple crowns, as illustrated in Figure 21.

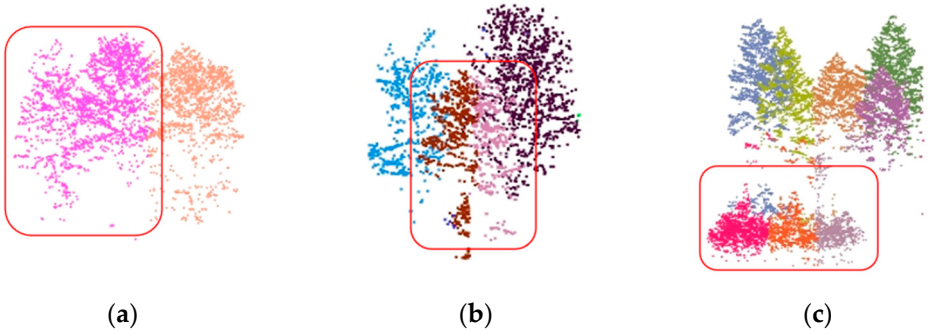


Figure 21. Results of the incomplete and failed segmented trees. (**a**) Adjacent trees are identified as a single crown object. (**b**) A single large crown as multiple crowns. (**c**) Misclassified shrubs (non-trees).

To address these challenges, future research could focus on introducing more advanced feature extraction mechanisms, coupled with sophisticated error-handling strategies and improved loss functions, to enhance model performance and effectively address challenges such as crown overlap and occlusion. Additionally, integrating richer semantic information (e.g., tree species classification) and multimodal data (e.g., multispectral and hyperspectral imagery) could provide comprehensive input, reducing misclassification in complex scenarios. Expanding the scale and diversity of training datasets to include varying forest densities, tree species, and terrain characteristics can further improve model robustness. Moreover, leveraging stronger pretraining strategies, such as self-supervised learning or pretraining on large-scale datasets, could significantly enhance the model's generalization capabilities under limited labeled data conditions. These improvements are expected to elevate the model's performance in complex forest environments while broadening its applicability to various real-world scenarios.

6. Conclusions

This paper presented CCD-YOLO, a novel deep learning-based method for tree detection. The key contributions of this work include the development of a dedicated single-tree segmentation dataset and the introduction of key architectural improvements. Specifically, the substitution of the C2f module with CReToNeXt, the incorporation of the CBAM attention mechanism, and the implementation of a dynamic head within the detection head collectively contribute to enhanced feature extraction, refined attention allocation, and optimized target localization. These modifications enable the CCD-YOLO to effectively capture both tree position and canopy boundary simultaneously, resulting in improved detection accuracy and addressing the prevalent issues of over- and under-segmentation in complex forest environments. Experimental comparisons with commonly adopted methods, along with evaluations using various internal consistency metrics, reveal the proposed model's superior performance, demonstrating strong adaptability and high detection and segmentation accuracy in complex forests.

CCD-YOLO has a few limitations, causing failures for the individual tree segmentation, which includes the misclassification of shrubs (non-trees) and over-segmentation. To address these challenges, future research could focus on optimizing feature extraction techniques and loss function design while integrating advanced error-handling strategies to improve the model's ability to handle crown overlap and occlusions, thus mitigating over-segmentation. In addition, integrating semantic information and multi-modal data would provide richer features, helping to reduce misclassification. Expanding the dataset and leveraging pretraining strategies (e.g., self-supervised learning) can significantly enhance the model's robustness and generalization capabilities, enabling its application in more complex scenarios. Moreover, an automatic method for extracting crown base height requires further investigation.

Author Contributions: Conceptualization, Y.L. and P.G.; methodology, Y.L. A.Z., and P.G.; software, Y.L. and P.G.; validation, Y.L.; writing—original draft preparation, Y.L. and A.Z.; writing—review and editing, Y.L. and A.Z.; funding acquisition, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the R&D Program of Beijing Municipal Education Commission (No. KM202410016006), the Pyramid Talent Training Project for Beijing University of Civil Engineering and Architecture (No. JDYC20220824), the Fundamental Research Funds for Beijing University of Civil Engineering and Architecture (No. Y2207), and the National Natural Science Foundation of China (No. 42001379).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cao, Y.; You, W.B.; Wang, F.Y.; Wu, L.Y.; He, D.J. Research progress on carbon storage of coarse woody debris in forest ecosystems. *Acta Ecol. Sin.* **2021**, *41*, 7913–7927.
2. Luo, H.B.; Yue, C.R.; Zhang, G.F.; Long, F.; Yang, W.J.; Xu, W.T. Application of Airborne LiDAR in Inversion Forest Leaf Area Index at Different Regional Scales. *J. West China For. Sci.* **2021**, *50*, 33–40.
3. Kaartinen, H.; Hyyppä, J.; Yu, X.; Vastaranta, M.; Hyyppä, H.; Kukko, A.; Holopainen, M.; Heipke, C.; Hirschmugl, M.; Morsdorf, F.; et al. An International Comparison of Individual Tree Detection and Extraction Using Airborne Laser Scanning. *Remote Sens.* **2012**, *4*, 950–974.
4. Dalponte, M.; Ørka, H.O.; Ene, L.T.; Gobakken, T.; Næsset, E. Tree crown delineation and tree species classification in boreal forests using hyperspectral and ALS data. *Remote Sens. Environ.* **2014**, *140*, 306–317. https://doi.org/10.1016/j.rse.2013.09.006.
5. Zhen, Z.; Quackenbush, L.; Zhang, L. Impact of Tree-Oriented Growth Order in Marker-Controlled Region Growing for Individual Tree Crown Delineation Using Airborne Laser Scanner (ALS) Data. *Remote Sens.* **2014**, *6*, 555–579.
6. Zhou, Y.; Wang, L.; Jiang, K.; Xue, L.; An, F.; Chen, B.; Yun, T. Individual tree crown segmentation based on aerial image using superpixel and topological features. *J. Appl. Remote Sens.* **2020**, *14*, 022210.
7. Morsdorf, F.; Meier, E.; Kötz, B.; Itten, K.I.; Dobbertin, M.; Allgöwer, B. LIDAR-based geometric reconstruction of boreal type forest stands at single tree level for forest and wildland fire management. *Remote Sens. Environ.* **2004**, *92*, 353–362. https://doi.org/10.1016/j.rse.2004.05.013.
8. Li, W.; Guo, Q.; Jakubowski, M.K.; Kelly, M. A New Method for Segmenting Individual Trees from the Lidar Point Cloud. *Photogramm. Eng. Remote Sens.* **2012**, *78*, 75–84.
9. Ferraz, A.; Bretar, F.; Jacquemoud, S.; Gonçalves, G.; Pereira, L.; Tomé, M.; Soares, P. 3-D mapping of a multi-layered Mediterranean forest using ALS data. *Remote Sens. Environ.* **2012**, *121*, 210–223. https://doi.org/10.1016/j.rse.2012.01.020.
10. Dong, T.; Zhang, X.; Ding, Z.; Fan, J. Multi-layered tree crown extraction from LiDAR data using graph-based segmentation. *Comput. Electron. Agric.* **2020**, *170*, 105213. https://doi.org/10.1016/j.compag.2020.105213.
11. Pang, Y.; Wang, W.; Du, L.; Zhang, Z.; Liang, X.; Li, Y.; Wang, Z. Nyström-based spectral clustering using airborne LiDAR point cloud data for individual tree segmentation. *Int. J. Digit. Earth* **2021**, *14*, 1452–1476.
12. Straker, A.; Puliti, S.; Breidenbach, J.; Kleinn, C.; Pearse, G.; Astrup, R.; Magdon, P. Instance segmentation of individual tree crowns with YOLOv5: A comparison of approaches using the ForInstance benchmark LiDAR dataset. *ISPRS Open J. Photogramm. Remote Sens.* **2023**, *9*, 100045. https://doi.org/10.1016/j.ophoto.2023.100045.
13. Polewski, P.; Yao, W.; Heurich, M.; Krzystek, P.; Stilla, U. Learning a constrained conditional random field for enhanced segmentation of fallen trees in ALS point clouds. *ISPRS J. Photogramm. Remote Sens.* **2018**, *140*, 33–44. https://doi.org/10.1016/j.isprsjprs.2017.04.001.
14. Li, Q.; Yan, Y.; Li, W. Coarse-to-fine segmentation of individual street trees from side-view point clouds. *Urban For. Urban Green.* **2023**, *89*, 128097. https://doi.org/10.1016/j.ufug.2023.128097.
15. Krisanski, S.; Taskhiri, M.S.; Gonzalez Aracil, S.; Herries, D.; Muneri, A.; Gurung, M.B.; Montgomery, J.; Turner, P. Forest Structural Complexity Tool—An Open Source, Fully-Automated Tool for Measuring Forest Point Clouds. *Remote Sens.* **2021**, *13*, 4677.
16. Kang, H.; Wang, X. Semantic segmentation of fruits on multi-sensor fused data in natural orchards. *Comput. Electron. Agric.* **2023**, *204*, 107569. https://doi.org/10.1016/j.compag.2022.107569.
17. Wielgosz, M.; Puliti, S.; Wilkes, P.; Astrup, R. Point2Tree(P2T)—Framework for Parameter Tuning of Semantic and Instance Segmentation Used with Mobile Laser Scanning Data in Coniferous Forest. *Remote Sens.* **2023**, *15*, 3737.
18. Henrich, J.; van Delden, J.; Seidel, D.; Kneib, T.; Ecker, A.S. TreeLearn: A deep learning method for segmenting individual trees from ground-based LiDAR forest point clouds. *Ecol. Inform.* **2024**, *84*, 102888.
19. Liu, X.; Hu, C.; Li, P. Automatic segmentation of overlapped poplar seedling leaves combining Mask R-CNN and DBSCAN. *Comput. Electron. Agric.* **2020**, *178*, 105753. https://doi.org/10.1016/j.compag.2020.105753.
20. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.
21. Shelhamer, E.; Long, J.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 640–651.
22. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597.
23. Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.
24. Freudenberg, M.; Magdon, P.; Nölke, N. Individual tree crown delineation in high-resolution remote sensing images based on U-Net. *Neural Comput. Appl.* **2022**, *34*, 22197–22207.

25. Ferreira, M.P.; Almeida, D.R.A.d.; Papa, D.d.A.; Minervino, J.B.S.; Veras, H.F.P.; Formighieri, A.; Santos, C.A.N.; Ferreira, M.A.D.; Figueiredo, E.O.; Ferreira, E.J.L. Individual tree detection and species classification of Amazonian palms using UAV images and deep learning. *For. Ecol. Manag.* **2020**, *475*, 118397. https://doi.org/10.1016/j.foreco.2020.118397.
26. Zheng, J.; Fu, H.; Li, W.; Wu, W.; Zhao, Y.; Dong, R.; Yu, L. Cross-regional oil palm tree counting and detection via a multi-level attention domain adaptation network. *ISPRS J. Photogramm. Remote Sens.* **2020**, *167*, 154–177. https://doi.org/10.1016/j.isprsjprs.2020.07.002.
27. Weinstein, B.G.; Marconi, S.; Bohlman, S.A.; Zare, A.; White, E.P. Cross-site learning in deep learning RGB tree crown detection. *Ecol. Inform.* **2020**, *56*, 101061. https://doi.org/10.1016/j.ecoinf.2020.101061.
28. Durgut, O.; Ünsalan, C. Multi-model tree detection in satellite images with weighted boxes fusion. *Signal Image Video Process.* **2024**, *19*, 32.
29. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 9992–10002.
30. Girshick, R.B.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
31. Liu, Y.; Sun, P.; Wergeles, N.; Shang, Y. A survey and performance evaluation of deep learning methods for small object detection. *Expert Syst. Appl.* **2021**, *172*, 114602. https://doi.org/10.1016/j.eswa.2021.114602.
32. Wang, Y.; Bashir, S.M.A.; Khan, M.; Ullah, Q.; Wang, R.; Song, Y.; Guo, Z.; Niu, Y. Remote sensing image super-resolution and object detection: Benchmark and state of the art. *Expert Syst. Appl.* **2022**, *197*, 116793. https://doi.org/10.1016/j.eswa.2022.116793.
33. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. *arXiv* **2020**, doi:arXiv:2010.04159.
34. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object Detection via Region-based Fully Convolutional Networks. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, arXiv.1605.06409.
35. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149.
36. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.E.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016.
37. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
38. Santos, A.A.D.; Marcato Junior, J.; Araujo, M.S.; Di Martini, D.R.; Tetila, E.C.; Siqueira, H.L.; Aoki, C.; Eltner, A.; Matsubara, E.T.; Pistori, H.; et al. Assessment of CNN-Based Methods for Individual Tree Detection on Images Captured by RGB Cameras Attached to UAVs. *Sensors* **2019**, *19*, 3595.
39. Wei, P.; Yan, X.; Yan, W.; Sun, L.; Xu, J.; Yuan, H. Precise extraction of targeted apple tree canopy with YOLO-Fi model for advanced UAV spraying plans. *Comput. Electron. Agric.* **2024**, *226*, 109425. https://doi.org/10.1016/j.compag.2024.109425.
40. Ali, M.L.; Zhang, Z. The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection. *Computers* **2024**, *13*, 336.
41. Chen, Y.; Xu, H.; Zhang, X.; Gao, P.; Xu, Z.; Huang, X. An object detection method for bayberry trees based on an improved YOLO algorithm. *Int. J. Digit. Earth* **2023**, *16*, 781–805.
42. Jintasuttisak, T.; Edirisinghe, E.; Elbattay, A. Deep neural network based date palm tree detection in drone imagery. *Comput. Electron. Agric.* **2022**, *192*, 106560. https://doi.org/10.1016/j.compag.2021.106560.
43. Xu, S.; Wang, R.; Shi, W.; Wang, X. Classification of Tree Species in Transmission Line Corridors Based on YOLO v7. *Forests* **2023**, *15*, 61.
44. Wardana, D.P.T.; Sianturi, R.S.; Fatwa, R. Detection of Oil Palm Trees Using Deep Learning Method with High-Resolution Aerial Image Data. In Proceedings of the 8th International Conference on Sustainable Information Engineering and Technology, Badung, Bali, Indonesia, 24–25 October 2023; pp. 90–98.
45. Bolya, D.; Zhou, C.; Xiao, F.; Lee, Y.J. YOLACT: Real-Time Instance Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9156–9165.
46. Ultralytics YOLOv5. Available online: https://github.com/ultralytics/yolov5 (accessed on 10 May 2024).
47. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
48. Ultralytics YOLOv8. Available online: https://github.com/ultralytics/ultralytics (accessed on 1 March 2024).

49. Nan, G.; Zhao, Y.; Lin, C.; Ye, Q. General Optimization Methods for YOLO Series Object Detection in Remote Sensing Images. *IEEE Signal Process. Lett.* **2024**, *31*, 2860–2864.
50. Badgujar, C.M.; Poulose, A.; Gan, H. Agricultural object detection with You Only Look Once (YOLO) Algorithm: A bibliometric and systematic literature review. *Comput. Electron. Agric.* **2024**, *223*, 109090. https://doi.org/10.1016/j.compag.2024.109090.
51. Dong, C.; Cai, C.; Chen, S.; Xu, H.; Yang, L.; Ji, J.; Huang, S.; Hung, I.K.; Weng, Y.; Lou, X. Crown Width Extraction of Metasequoia glyptostroboides Using Improved YOLOv7 Based on UAV Images. *Drones* **2023**, *7*, 336.
52. Liu, Y.; Zhao, Q.; Wang, X.; Sheng, Y.; Tian, W.; Ren, Y. A tree species classification model based on improved YOLOv7 for shelterbelts. *Front Plant Sci* **2023**, *14*, 1265025.
53. Zhao, Z.; Li, D.; Zhao, D.; Cheng, Z.; Guo, X. Canopy Segmentation and Biomass Estimation Based on Deep Learning. *For. Eng.* **2024**, *40*, 145–155.
54. Sun, C.; Huang, C.; Zhang, H.; Chen, B.; An, F.; Wang, L.; Yun, T. Individual Tree Crown Segmentation and Crown Width Extraction From a Heightmap Derived From Aerial Laser Scanning Data Using a Deep Learning Framework. *Front. Plant Sci.* **2022**, *13*, 914974.
55. Puliti, S.; McLean, J.P.; Cattaneo, N.; Fischer, C.; Astrup, R. Tree height-growth trajectory estimation using uni-temporal UAV laser scanning data and deep learning. *For. Int. J. For. Res.* **2022**, *96*, 37–48.
56. Zhang, F.; Zhao, P.; Xu, S.; Wu, Y.; Yang, X.; Zhang, Y. Integrating multiple factors to optimize watchtower deployment for wildfire detection. *Sci. Total Environ.* **2020**, *737*, 139561. https://doi.org/10.1016/j.scitotenv.2020.139561.
57. Zhang, L.; Wang, M.; Liu, M.; Zhang, D. A Survey on Deep Learning for Neuroimaging-Based Brain Disorder Analysis. *Front. Neurosci.* **2020**, *14*, 779.
58. Puliti, S.; Astrup, R. Automatic detection of snow breakage at single tree level using YOLOv5 applied to UAV imagery. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102946. https://doi.org/10.1016/j.jag.2022.102946.
59. Li, P.; Wu, H.; Jing, J.; Li, R. Noise classification denoising algorithm for point cloud model. *Comput. Eng. Appl.* **2016**, *52*, 188–192.
60. Zhang, W.; Qi, J.; Wan, P.; Wang, H.; Xie, D.; Wang, X.; Yan, G. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sens.* **2016**, *8*, 501.
61. Wang, R.R.; Li Y.R.; Shi W.; Li W.J. Single Wood Extraction Algorithm Based on LIDAR Data. *J. Northwest For. Univ.* **2021**, *36*, 182–189.
62. Xu, X.; Jiang, Y.; Chen, W.; Huang, Y.-L.; Zhang, Y.; Sun, X. DAMO-YOLO : A Report on Real-Time Object Detection Design. *arXiv* **2022**, arXiv:2211.15444.
63. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.-S. CBAM: Convolutional Block Attention Module. *arXiv* **2018**, arXiv:1807.06521.
64. Dai, X.; Chen, Y.; Xiao, B.; Chen, D.; Liu, M.; Yuan, L.; Zhang, L. Dynamic Head: Unifying Object Detection Heads with Attentions. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 7369–7378.
65. Wang, J.; Zhang, H.; Liu, Y.; Zhang, H.; Zheng, D. Tree-Level Chinese Fir Detection Using UAV RGB Imagery and YOLO-DCAM. *Remote Sens.* **2024**, *16*, 335.
66. Vincent, L.; Soille, P. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Mach. Intell.* **1991**, *13*, 583–598.
67. Ultralytics YOLO11. Available online: https://github.com/ultralytics/ultralytics (accessed on 4 December 2024).