SPTNet: Sparse Convolution and Transformer Network for Woody and Foliage Components Separation From Point Clouds

Shuai Zhang, Yiping Chen, *Senior Member, IEEE*, Biao Wang, Dong Pan, Wuming Zhang, and Aiguang Li

***Abstract*— The separation of woody and foliage components is beneficial in estimating the physical parameters of forests. However, many current methods incur high computational costs and rely on extensive prior knowledge. These methods display weak abilities in generalization for component separation from various light detection and ranging (LiDAR) sensors and tree species. In this article, a network that combines sparse convolution (SpConv) and transform blocks is proposed for the separation of woody and foliage components in tree point clouds called SPTNet. The SpConv block facilitates efficient and effective local feature extraction, while the transformer block offers a solution for the inadequate global feature extraction in SpConv blocks. Point feature extraction blocks, called morphological detection coefficient (MDC) and normal difference operator (NDO), were specifically developed to aid in the segmentation task. Distinct adaptive radius strategies are implemented for each geometric feature block to minimize the need for a priori knowledge. Eight different tree species datasets were used to improve methods, including a simulated larch dataset. The other datasets consist of actual trees and comprise seven distinct tree species along with a large tropical tree dataset. Our experimental results demonstrate that our method attains state-of-the-art performance across all datasets. It is worth mentioning that SPTNet obtains an overall classification accuracy (OA) of 94.69% and 89.96% mean of intersection-over-union (mIoU) on the large tropical dataset, which encompasses 15 tree species. Moreover, SPTNet outperforms FWCNN, the current leading branch and leaf separation approach, by 0.43% OA and 0.72% mIoU.**

***Index Terms*— Adaptive radius, geometric feature, sparse convolution (SpConv), transformer, woody and foliage components separation.**

I. Introduction

FORESTS are important resources of the Earth and play an important role in the ecological environment. Tree canopy structures control energy transfer between the atmosphere and terrestrial ecosystems through photosynthesis and transpiration. The 3-D representation and structures of trees are widely used in forestry research. Diameter at breast height (DBH) as the key parameter of the trunk was used to estimate biomass and carbon storage [1], [2], [3], [4]. The characterization of forest structure, especially the distinction between photosynthetic parts (crown and leaves) and nonphotosynthetic parts (branches and trunks), has become the basis and key of forest research. Distinguishing woody and foliage components is helpful to estimate forest physical parameters, such as leaf area density [5], effective leaf area index [6], and woody-to-total area ratio [7].

With the development of technology, terrestrial laser scanning (TLS) technology is more and more widely used in forestry. Point cloud data acquired by light detection and ranging (LiDAR) have a wide range of applications in forestry research, such as ground filtering [8], [9], tree extraction [10], single tree segmentation [11], [12], woody and foliage components separation [13], [14], and other tasks. As an active remote sensing technology, TLS is different from satellite images. It is capable of obtaining the 3-D structure information of trees with millimeter-level accuracy and then calculating various forest parameters. Therefore, most studies on the differentiation of woody and foliage components are based on TLS data [15], [16].

At present, there are many studies on the separation of woody and foliage components from the tree point cloud. There are three kinds of methods proposed in many types of research: one is based on local geometric features [15], [17], [18] and the other is combined with radiometric features [laser radiation information (LRI)] [19], and there are several methods to combine the two types of features [16], [20], [21]. Generally, the acquired TLS data contain XYZ-coordinate geometry information, reflection intensity, and other reflection information. The methods based on radiation information are limited to the laser wavelength, and hence, the radiation information corresponding to different sensors will be slightly different. LRI is the ability of a sensor to pick up the intensity of reflections from different ground objects. The difference in reflected intensity can be obtained point by point for differentiating between wood and leaf components.

However, the method based on geometric information only relies on the XYZ-coordinates of trees. Because of the rotation invariance of point clouds, the method based on geometry is more universal to use in the data obtained by various sensors. In forest point cloud structure separation processing, many

Manuscript received 22 February 2023; revised 9 November 2023 and 7 February 2024; accepted 27 February 2024. Date of publication 18 March 2024; date of current version 11 April 2024. This work was supported in part by the National Nature Science Foundation of China under Grant 41971380 and Grant 42371343, and in part by Guangxi Natural Science Fund for Innovation Team under Grant 2019GXNSFGA245001. *(Corresponding authors: Yiping Chen; Wuming Zhang.)*

The authors are with the School of Geospatial Engineering and Science, Sun Yat-sen University, Zhuhai 519082, China (e-mail: zhangsh255@mail2.sysu.edu.cn; chenyp79@mail.sysu.edu.cn; wangb325@mail2.sysu.edu.cn; pand25@mail2.sysu.edu.cn; zhangwm25@mail.sysu.edu.cn; liaiguang@mail.sysu.edu.cn).

Digital Object Identifier 10.1109/TGRS.2024.3376454

geometric features are used for many studies. The geometric features are mainly used to calculate the eigenvalues and eigenvectors of a single point and then carry out further feature calculations, such as point cloud normal vector, curvature, and perpendicularity. These geometric features can be calculated in many ways, such as K-neighborhood-based [22], voxel-based [23], and sphere-based [24]. The method based on K-neighborhood is widely used in forestry research. However, this method needs to determine the neighborhood radius r, previous studies mostly set it based on prior knowledge and point cloud quality. Later, an adaptive radius determination method was proposed to make it more automatic [25].

There are not only methods based on the physical mechanisms to solve the problem of woody and foliage components separation but also some methods based on statistics have been developed, such as the supervised classification method random forest (RF) [16] and support vector machines (SVMs) [26]. However, RF is prone to overfitting. It is heavily influenced by the presence of multiple inputs at the same time, resulting in unreliable results. SVM is a quadratic programming-based method, which is difficult to apply to large-scale training samples. Meanwhile, there are some unsupervised classification methods such as DBSCAN [27] and LeWoS [13], which have good performance in the separation of woody and foliage components. Nevertheless, the unsupervised methods are slightly less accurate than supervised methods.

However, there are still many challenges in tree point cloud woody and foliage components.

1) For data acquired by different sensors, the optical features, such as reflectivity, will vary with the sensor. Also, it has great difficulties in data fusion due to different radiation features. Therefore, the method based on reflection features is difficult to adapt to the data and has poor generalization.
2) Many parameters are required in the traditional method to achieve woody and foliage components segmentation of tree point clouds. However, these parameters are mostly set manually, and different parameter choices can have a great impact on the results.
3) For different tree species, their structures vary greatly, and the existing methods have great shortcomings in terms of accuracy and efficiency. The differences in tree species have been a great challenge for the woody and foliage components separation problem, and no method can guarantee accuracy with high efficiency. Summarily, to the best of our knowledge, there are no deep learning methods used for the task of separating tree woody and foliage components.

In this article, we proposed a method for the separation of woody and foliage components of tree point clouds. First, inspired by the superior performance of Unet [28] structure on other tasks, a novel segmentation network is designed for implementing the tree point cloud component separation, named **SPTNet**. The method is implemented with Unet as the backbone network with sparse convolution (SpConv) and transformer blocks. Second, two blocks of morphological detection coefficient (MDC) and normal difference operator (NDO) describe the morphological information within the neighborhood of the point cloud. The experimental results show that our method is highly accurate and efficient in separating woody and foliage components of trees. For the task of separating woody and foliage components of tree point clouds, our method is superior to the state-of-the-art methods. In addition, the proposed method can be applied to different tree species for versatility and effectiveness. The main contributions of our method are summarized as follows.

1) A workflow is proposed to solve the woody and foliage components separation of the tree point cloud problem, which is a combination of physical features and deep learning networks. At the same time, the method uses only the geometric features of the point cloud data and does not require the use of radiometric features. This allows the network to be adapted to all LiDAR sensors. Meanwhile, the determination of physical feature parameters in our workflow is adaptive and therefore does not violate the idea of end-to-end.
2) A novel network based on SpConv and transformer is designed for feature extraction from point cloud data, called SPTNet. SPTNet combines the high efficiency of SpConv with the high performance of transformer and fuses local and global features. This allows SPTNet to be applied to larger samples, even with single samples exceeding 100 000 points.
3) Two geometric feature blocks (MDC and NDO) are used for assisting the separation of the woody and foliage components from tree point clouds. Both feature blocks are based on different combinations of eigenvalues within the neighborhood of the point cloud. Methods for determining the adaptive neighborhood radius were designed for each of these two blocks. This approach avoids errors and loss of time and manpower due to manual involvement.
4) SPTNet achieved good segmentation results on both broad-leaved forests and coniferous forests, demonstrating the effectiveness and generality of the network. The method has experimented on seven tree species datasets and a large tropical tree dataset. The overall classification accuracy (OA) of the large tropical tree dataset was 94.69%, and the mean of intersection-over-union (mIoU) was 89.96%. In addition, our network is superior to the state-of-the-art segmentation network in terms of accuracy and efficiency.

II. Related Work

A. Woody and Foliage Components Separation

Currently, for the tree point cloud woody and foliage components separation task, commonly used methods are based on radiometric or geometric features or a combination of both. A further study [19] found stronger absorption at 1548 nm in foliage compared to woody components. Furthermore, a tree point cloud dendritic separation task was performed according to the sensitivity to different wavelengths of light. Since the radiation signature depends mainly on the wavelength used by a particular sensor, the method has high equipment

requirements and cannot be applied to most of the scenes. Pure geometry-based methods only need to utilize the XYZ-coordinates of the tree and can be applied to data obtained from different types of sensors, so there are more methods based on the geometric characteristics of the tree. GAFPC [29] is proposed, which uses the expectation–maximization (EM) algorithm to estimate the model parameters of the Gaussian mixture model (GMM) for each class, i.e., as a classifier for point-by-point classification. However, the classification results of this method are poor, and the authors propose the need for postprocessing with multiple filters, which shows that it is difficult to guarantee efficiency while ensuring accuracy. The radiometric and geometric first obtained features of the tree point clouds in 2018 [16]. Then, an RF machine learning algorithm was used to classify also the woody and foliage components. The method requires a combination of both features for classification and the acquisition of features requires human selection of parameters. In addition to supervised classification methods, there have been more unsupervised studies in recent years. Ferrara et al. [27] propose to divide the point cloud into voxels, which are used as input to generate clusters by the point density algorithm DBSCAN, which requires two global input parameters, the radius size and the minimum number of clustering points. The choice of parameters and the quality of the point cloud have a large impact on the method. LeWoS [13] as a new fully automatic tool to automate the separation of woody and foliage components is proposed, based only on geometric information at both the plot and individual tree scales.

B. Point Cloud Deep Learning Networks

With the rapid development of deep learning technology in 2-D images, more and more researchers use deep learning methods to solve the point cloud problem. Based on the minimum unit division of network processing, point cloud deep learning methods can be divided into voxel-based [30], [31], multiview-based [32], [33], and point-based [24], [34], [35]. Given the excellent performance of convolutional neural networks (CNNs) on images, there is more work looking to extend convolution to point cloud processing. However, point cloud data are discrete and disordered and thus require normalization of the data before convolution. As a result, a number of voxel-based and multiview-based point cloud neural networks have emerged. VoxNet [36] was proposed in 2015, as a point cloud with voxel specification that not only makes fuller use of 3-D information but also reduces the amount of data processing and increases efficiency. PointGroup [37] used SpConv to process point cloud data after voxelization and extended SpConv to various network structures such as UNet and FCN. The method achieves good results in the field of 3-D target recognition. However, SpConv is good for local information extraction, but the control of global information is obviously insufficient. MVCNN [33] is proposed, which performs the subsequent task by acquiring 12 different angular views of a 3-D object and later fusing the 12 images after extracting features by CNN. However, the multiview approach requires more preprocessing, as well as a significant amount of data loss during processing. PointNet [34] is the earliest point-based network. As a pioneering work, PointNet uses multiple multilayer perceptron (MLP) to model each point independently and then aggregate global features using symmetric functions. PointNet can guarantee the permutation invariance of point clouds. To continue to enhance the network's learning of neighborhood features, subsequent works, such as PointNet++ [24], KPConv [35], and others, have emerged that consider neighborhoods. PointMLP [38] is proposed, which is a pure MLP network based on PointNet++ for improvement. The authors use a skip-connected MLP to extract features while using a geometric affine transformation to solve the problem of density inhomogeneity and geometric structure uncertainty of the point cloud. The simple structure of MLP makes the method significantly better than other methods in terms of efficiency.

In 2017, the transformer model came out of nowhere and quickly led the way in the field of natural language processing (NLP) as well as other related fields. With the emergence of ViT [39] and Swin Transformer [40], the transformer has created a boom in the field of computer vision. Recently, methods based on transformer [30], [41], [42] structures have also been usefully proposed in point cloud processing. Point Transformer [42] was proposed in 2021. The authors discarded the previously used convolutional structure and used k-nearest neighbor (KNN) to search the point set and used pure self-attention for feature aggregation within the point set. However, the method requires multiple stacks of self-attentive blocks, a large number of parameters, and a slow training speed. PointBert [41] is proposed by analogy with Bert, the most successful training method in NLP, which uses mask learning to train a bidirectional transformer, a self-supervised training model. Given that point-based transformers all suffer from slow speed, VoTr [30] is proposed, a voxel-based transformer model. This method proposes sparse transformer blocks for 3-D object detection tasks. Transformer is widely used for its global nature of extracting features, but its high computational complexity leads to slow efficiency. Traditional CNNs allow the network to obtain accurate local features because of the multiple stacking of small convolutional kernels. However, obtaining global features would require much larger computational resources. Hence, some methods based on the combination of convolution and transformer have appeared in 2-D images [43], [44], [45].

At the same time, there are a few deep learning-based methods to deal with tree component segmentation. The method [46] is used based on MLP and CNN to complete the segmentation of rosebush plants and compare the accuracy of the methods. Ao et al. [47] proposed the automatic segmentation of corn stem and leaf components by the convolution neural network method. These methods are aimed at specific plants, and it is difficult to have universalities with other plants. The data of the single plant are simple, and it is difficult to get good results in complex situations such as trees. Based on the PointNet [34] model, FWCNN [14] finally obtained the best classification results by combining geometric features (normal vector) of tree point cloud and tree radiation features (LRI) through a large number of parameter optimization processes.

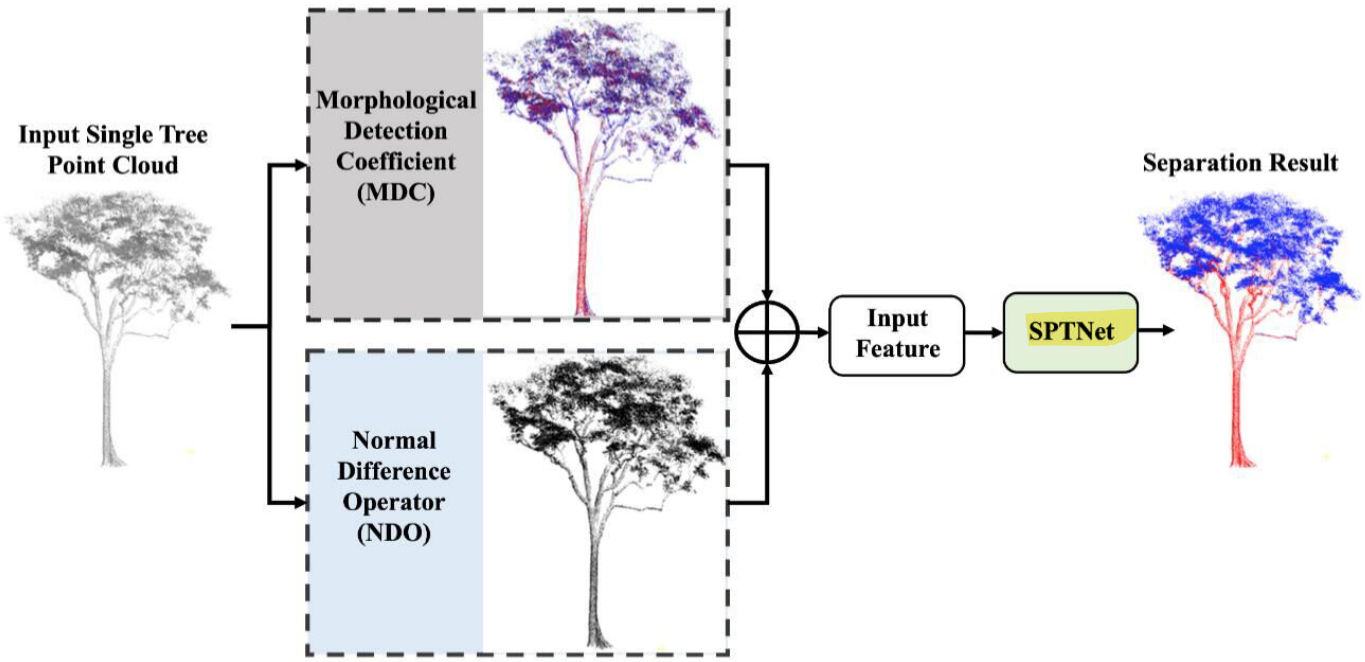

Fig. 1. Complete architecture of our proposed workflow.

However, FWCNN has obvious limitations. First, FWCNN requires both geometric and reflective features to be input in order to obtain more competitive results. Second, FWCNN requires tree point cloud preprocessing and a lot of manual parameterization in the calculation of geometric features, which leads to the fact that the whole method is not automated. Third, FWCNN cannot be applied to more tree species due to network performance and input limitations. These issues are addressed in the SPTNet network.

III. Method

The complete architecture of our proposed workflow is shown in Fig. 1. Our proposed method can be divided into the following two phases: the first phase is point feature extraction and the second phase is neural network feature aggregation and segmentation. In the point feature extraction stage, two auxiliary feature extraction blocks (MDC and NDO) are designed. Accordingly, the adaptive neighborhood radius r is addressed for each block. In the neural network feature aggregation and classification stage, a data organization strategy is used based on a combination of voxels and points. Inspired by UNet, the feature aggregation method combining SpConv and transformer is used.

It will explain some details of the whole architecture in detail. First, two geometric features are designed for the auxiliary segmentation and the determination of the adaptive radius accordingly (Section III-A). Then, the structure of each block in our proposed segmentation network will be described in detail (Section III-B).

A. Point Feature Extraction

In previous studies on the separation of woody and foliage components, geometric and optical features are usually used to assist in segmentation. However, these two features have different properties and require different sensors for the acquisition, which will increase the acquisition cost. Moreover, it believes that the information contained in the most basic 3-D coordinates of the point cloud has not been exploited. Therefore, only use purely geometric features (relying only on the XYZ-coordinates of the point cloud) to help separate the woody and foliage components.

Two point feature blocks are extracted for each point by only the coordinate information of the point and the neighbor relationship in the point cloud.

1) MDC Block: MDC is a parameter that describes the geometric characteristics of the points in the neighborhood.

For a point set $P = \{p_i(x_i, y_i, z_i), i = 1, 2, \ldots, n\}$, at first, the row matrix p_i can be computed as

$$\vec{p} = \frac{1}{n}\sum_{i=1}^{n}(x_i, y_i, z_i)^T \tag{1}$$

where T means the transposed matrix and $\vec{p}$ is the row matrix of the mean coordinates for the n points in the given local point set P.

The point cloud covariance matrix is calculated according to row matrix $\vec{p}$. Also, the covariance matrix (C_{cov}) of p can be computed as

$$C_{\text{cov}} = \frac{1}{n}\sum_{i=1}^{n}\left(\vec{p}_i - \vec{p}\right)^T\left(\vec{p}_i - \vec{p}\right). \tag{2}$$

Three nonnegative eigenvalues (λ_1–λ_3) are calculated for each point of the point cloud according to the covariance matrix. Also, the three eigenvalues meet the following conditions: $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$. α_1–α_3 are obtained by multiplying the three eigenvalues by 10 000. The purpose is to magnify the differences between different features and better segmentation.

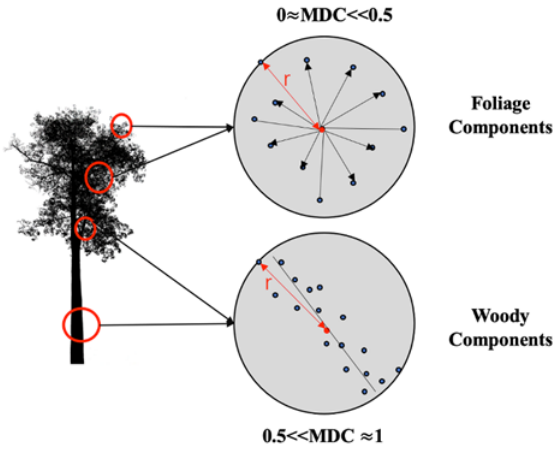

Fig. 2. MDC between woody and foliage components.

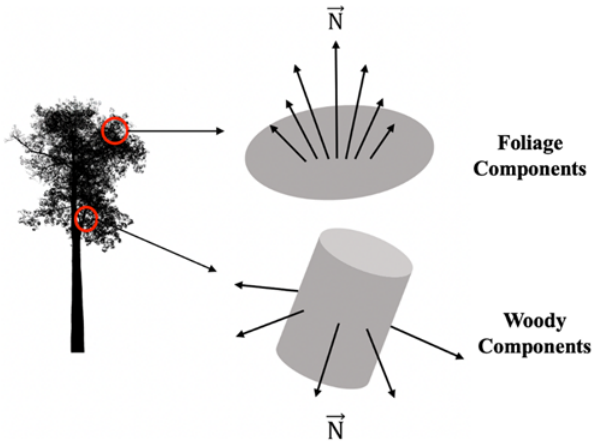

Fig. 3. Normal vector difference between woody and foliage components.

Finally, the MDC values can be calculated as follows:

$$\mathrm{MDC} = \frac{1}{1 + e^{\alpha_1 - \alpha_2} \cdot e^{\alpha_3 - \alpha_2}}. \tag{3}$$

It can be seen from the above formula that MDC characteristics are related to the three eigenvalues (λ_1–λ_3) of point cloud. The ordered eigenvalues of C_{cov} are the distribution morphological indices to represent the spatial distribution patterns of P. When $\lambda_1 = \lambda_2 = \lambda_3$, the MDC value is 0.5, indicating that all points in the set P are randomly distributed in the space; if $\lambda_1 \geq \lambda_2 = \lambda_3$, the MDC value is greater than 0.5, denoting that the points in set P are linearly distributed; if $\lambda_1 \approx \lambda_2 > \lambda_3$, the MDC value is less than 0.5, denoting that the points in set P are distributed in a plane shape. Fig. 2 shows the MDC between woody and foliage components. The red dots indicate the points we are interested in. The large circles represent the neighborhood with a radius of r at the center of the point. The other small circles represent other points in the neighborhood. The point clouds of the foliage component neighborhoods are distributed in a planar pattern, and the point clouds of the woody component neighborhoods are distributed in a linear pattern.

2) NDO Block: NDO is a parameter that represents the change in the direction of a normal vector in a neighborhood.

The normal vector of the point cloud is a very important geometric feature for the disordered 3-D point cloud. The point cloud normal vector represents the direction perpendicular to the point cloud tangent plane. If the normal vector direction of adjacent points is almost the same, it indicates that the surface of the point has not changed significantly. If the normal vector direction of the central point and the surrounding point changes greatly, it indicates that the structure of the region is significantly different. Depending on the composition of woody and foliage components, most foliage components point clouds are nearly planar, while woody components point clouds are close to cylindrical. Therefore, most foliage components' point clouds have normal vectors with similar directions, while woody components' point cloud normal vector directions are significantly different. This is shown in Fig. 3. The difference in the foliage components point normal vector is smaller than the normal vector difference in the nonleaf point in the neighborhood. Therefore, an NDO is proposed to assist in the separation of tree points, and the NDO between each point and other points in the neighborhood is calculated.

For the point p_i in set P, NDO in the neighbors is computed as follows:

$$\mathrm{NDO} = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{n}(p) - \hat{n}(p_i) \right) \tag{4}$$

where $\hat{n}(p)$ is the normal vector of point p in the neighbors and $\hat{n}(p_i)$ is the normal vector of point p_i.

3) Adaptive Neighborhood Radius: As mentioned above, MDC and NDO are used to assist in the task of separating woody and foliage components. However, the calculation of both parameters requires the construction of the neighborhood for the calculation and the determination of the neighborhood radius r. MDC features represent how the shapes of the neighborhood points within all points are distributed within a single tree point cloud. Therefore, radius r needs to be calculated for each point. NDO features characterize changes in normal vectors within different neighborhoods in a single sample. This feature needs to be compared with each other within the sample to determine the category, so it is necessary to ensure that the neighbor radius r in a single sample is equal. In previous studies, the neighborhood radius values would be chosen manually based on experience. However, experiments have shown that the choice of r greatly affects the accuracy of the final point cloud segmentation. Also, for different r's, the two eigenvalues (MDC and NDO) have significant variations. However, in the face of different tree species, different point cloud densities, and large sample sizes, it is impractical to artificially control the appropriate r values. Therefore, an adaptive approach to determining the neighborhood radius r is considered. Since the scale of r values required for the two features is different, two adaptive methods were used to determine the r values separately in this study.

1) *Determination of r_{MDC}:* Demantké et al. [25] proposed to determine the adaptive threshold using the principle of minimum entropy. Later, some scholars introduced

this idea into forestry research [48]. Since the point cloud neighborhood eigenvalues contain a large amount of point cloud information, it is common to learn that the traditional method is worth combining point cloud features. Three indicators have been proposed to characterize point cloud neighborhood geometry: linear–$\alpha_{1\text{-D}}$, planar–$\alpha_{2\text{-D}}$, and scatter–$\alpha_{3\text{-D}}$. $\alpha_{1\text{-D}}, \alpha_{2\text{-D}}$, and $\alpha_{3\text{-D}}$ are computed as follows:

$$\alpha_{1\text{-D}} = \frac{\sqrt{\lambda_1} - \sqrt{\lambda_2}}{\mu}, \alpha_{2\text{-D}} = \frac{\sqrt{\lambda_2} - \sqrt{\lambda_3}}{\mu}, \alpha_{3\text{-D}} = \frac{\sqrt{\lambda_3}}{\mu} \tag{5}$$

where μ is the is the normalization coefficient. The purpose is to ensure that $\alpha_{1\text{-D}}, \alpha_{2\text{-D}}$, and $\alpha_{3\text{-D}}$ are within the range of [0, 1]. λ_1–λ_3 are the three eigenvalues of the covariance matrix of set P.

Let us define the entropy function E_f with respect to $\alpha_{1\text{-D}}, \alpha_{2\text{-D}}$, and $\alpha_{3\text{-D}}$. E_f can be computed as follows:

$$E_f = -\alpha_{1\text{-D}} \ln(\alpha_{1\text{-D}}) - \alpha_{2\text{-D}} \ln(\alpha_{2\text{-D}}) - \alpha_{3\text{-D}} \ln(\alpha_{3\text{-D}}). \tag{6}$$

Woody components are defined as positive samples, foliage components are defined as negative samples, linear feature ($\alpha_{1\text{-D}}$) is defined as positive samples, and plane feature ($\alpha_{2\text{-D}}$) is defined as negative samples. The scattering feature ($\alpha_{3\text{-D}}$) in the tree point cloud is ignored. The smaller E_f value is, the more linear feature ($\alpha_{1\text{-D}}$) is reflected than the other two features ($\alpha_{2\text{-D}}$ and $\alpha_{3\text{-D}}$). Since the corresponding E_f for any r value is calculated, we adopt the method of minimum entropy to determine the optimal neighborhood radius r. Therefore, within the range $[r_{\min}, r_{\max}]$, the most appropriate method for determining the r value of MDC is as follows:

$$r_{\text{MDC}} = \text{argmin}_{r \in [r\min, r\max]} E_f. \tag{7}$$

After the optimal r_{MDC} value is calculated, the MDC characteristics of a single point are further calculated.

2) *Determination of r_{NDO}:* Unlike the r_{MDC} values used in MDC features, NDO needs to be compared with the normal vector changes in the same sample. Therefore, the value of r_{NDO} needs to be determined that is one in each sample. Here, an algorithm often used for image thresholding segmentation—Otsu [49] solves this problem. Otsu can classify an object into two categories based on the size of the difference between the categories. The larger the difference between the categories, the larger the difference between the two categories, and the smaller the probability of misclassification.

With the above two adaptive threshold determination methods, the MDC and NDO features of the point cloud are calculated for subsequent neural network separation of the woody components and foliage components of the tree point cloud.

B. SPTNet

SPTNet is proposed for the separation of woody and foliage components of tree point clouds. The whole network is an encoder–decoder structure. The encoder section contains five SpConv blocks and five transformer blocks. SpConv blocks are used to extract local features, and transformer blocks are used to aggregate global features. The decoder part contains four corresponding SpConv blocks to reorganize the new aggregation features. The seven dimensions vectors of XYZ, MDC, and NDO are the input of SPTNet. For data organization, voxel-based approaches facilitate convolution and improve computational efficiency. Unlike traditional CNNs, a more efficient and advanced combination of SpConv and transformer is used for feature aggregation. The model will be introduced sequentially in terms of network structure, model training, and accuracy evaluation.

1) Structure of SPTNet: Fig. 4 shows the basic architecture of SPTNet. Benefiting the excellent performance of the Unet network in 2-D and 3-D tasks, our backbone network is inspired by the UNet architecture. The whole network is an encoder–decoder structure. To extract the global and partial features of the tree point cloud more efficiently and completely, we designed a new network.

The encoder section contains five SpConv blocks and five transformer blocks. SpConv block is used to capture and extract the local information within a specific neighborhood of the point cloud. Due to the limited size of the convolution kernel, it is difficult to obtain global features. Therefore, a transformer block is designed to extract the global features from the convolved feature map. There are six transformer blocks used in the stack, which is consistent with the operations used in other works. Finally, the feature map after convolution is combined with the feature map after transformer by skip connection to get the aggregated feature map.

The decoder section contains four corresponding SpConv blocks to reduce the dimensionality from the high-dimensional features aggregated by the encoder by inverse SpConv. The probability output of tree woody and foliage components is obtained by linear layer. The structures of the sparse convolutional block and transformer block in the network are described separately in the following.

2) SpConv Block: Images are generally stored in pixels by rows and columns. For images of $W \times H$ size, it is equivalent to dividing the image into grids. In turn, these grids can perform some convolution, pooling, and other operations. Corresponding to the 3-D space, the point cloud is composed of points, and usually, each point consists of three coordinates: X, Y, and Z. The three coordinates correspond to the grayscale values in a 2-D image. However, compared to 2-D images, 3-D point clouds are distributed in 3-D space while being nonuniformly distributed. Voxelization divides the point cloud into a grid of uniformly spaced voxels and then generates a many-to-one mapping between the 3-D points and their respective voxels.

To facilitate convolution, a standard voxelization operation on the tree point cloud is performed before inputting it into the neural network. The voxelization processes of a regular grid. Using the XYZ-coordinates of the point cloud, voxels of the same size are generated. Each voxel is represented by the average of the points within the voxel to represent the point cloud features in that voxel. Voxel coordinates are generated

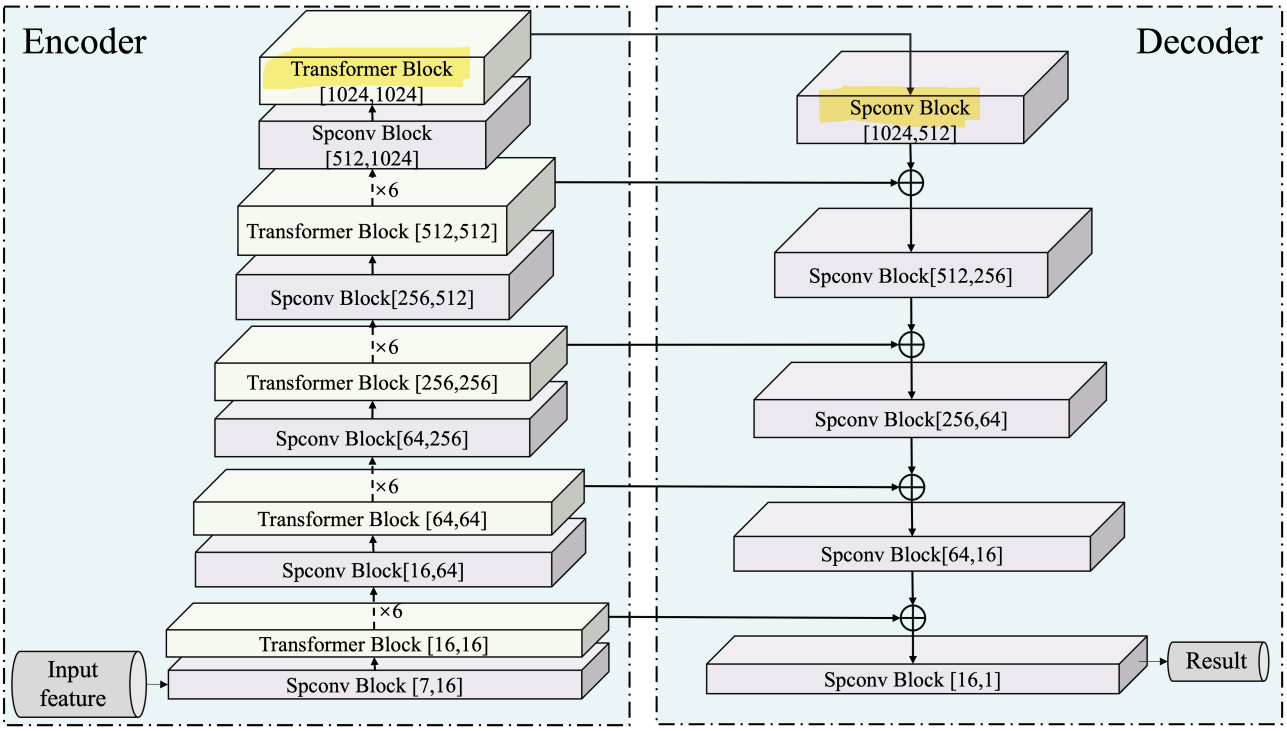

Fig. 4. Basic architecture of SPTNet.

using the spatial position of voxels. To prevent data leakage, MDC and NDO are used to form other total 4-D features and combine voxel coordinates with geometric features in the form of a hash table.

Because of voxelization, the characterizations of our data source are sparse due to the specificity of the tree structure. In contrast, the convolutional implementation in traditional work is optimized for data on dense grids and cannot effectively work with sparse data. Traditional convolution makes data smooth and destroys the sparsity of the data. The number of points on the grid grows exponentially as the dimensionality increases. In this case, it becomes increasingly important to exploit data sparsity as much as possible to reduce the computational resources required for data processing. Therefore, the SpConv block is designed, which combines a combination of submanifold SpConv (SubMConv) [50] and SpConv [51]. It is used to extract feature information more accurately and to improve efficiency. Where SpConv is similar to ordinary convolution, as long as there are values in the convolution kernel, convolution operations are performed. SubMConv determines whether the center of the convolution kernel has a value. The SubMConv convolution operation is performed only if the center of the convolution kernel has a value.

Fig. 5(a) shows the structure of the SpConv block. Three SubMConv layers are used for feature extraction and a normal SpConv layer for downsampling. The skip connection is used to merge the feature maps obtained from each convolution. This approach avoids vanishing gradient and network degradation problems. Also, it maximizes the use of each convolution result. The convolution kernel size is set to 3, together with a padding of 1 to complete the SubMConv. It can extract the feature information within the neighborhood of the point cloud in a detailed way and ensure the sparsity of the data.

3) Transformer Block: The transformer structure emerged in 2017 and immediately made a splash in the NLP field. Due to the advent of ViT, the transformer is becoming more and more widely used in 2-D images. Transformer allows modeling the dependencies of input–output sequences without considering their distances in the sequence. Compared to CNN, the number of operations required to compute the correlation between two locations does not increase with distance. Thus, reliable global features can be obtained with lower computational complexity. Rather than simply stacking blocks, the transformer block is designed to obtain a reliable global feature by aggregating the convolved local features. Since a convolution kernel of size 3 is used to obtain the local features of the point cloud more finely. However, a smaller convolution kernel will result in a smaller field of view, and it is not easy to obtain reliable global features. Therefore, the transformer block is designed to aggregate the global features after convolution.

Fig. 5(b) shows the structure of the transformer block. Since the global features are acquired for a single tree sample, multiheaded self-attention is still used in the transformer block. The feature maps obtained by SpConv block input into different linear layers to obtain Q, K, and V for computing self-attention weight. Multihead attention is similar to multiple convolution kernels in convolution, where different attention heads can acquire different features. After combining the feature maps acquired by multiple heads, the feature maps after multihead attention are obtained. To maximize the use of features, the skip connection is used continually. The feature map after convolution is combined with the feature map

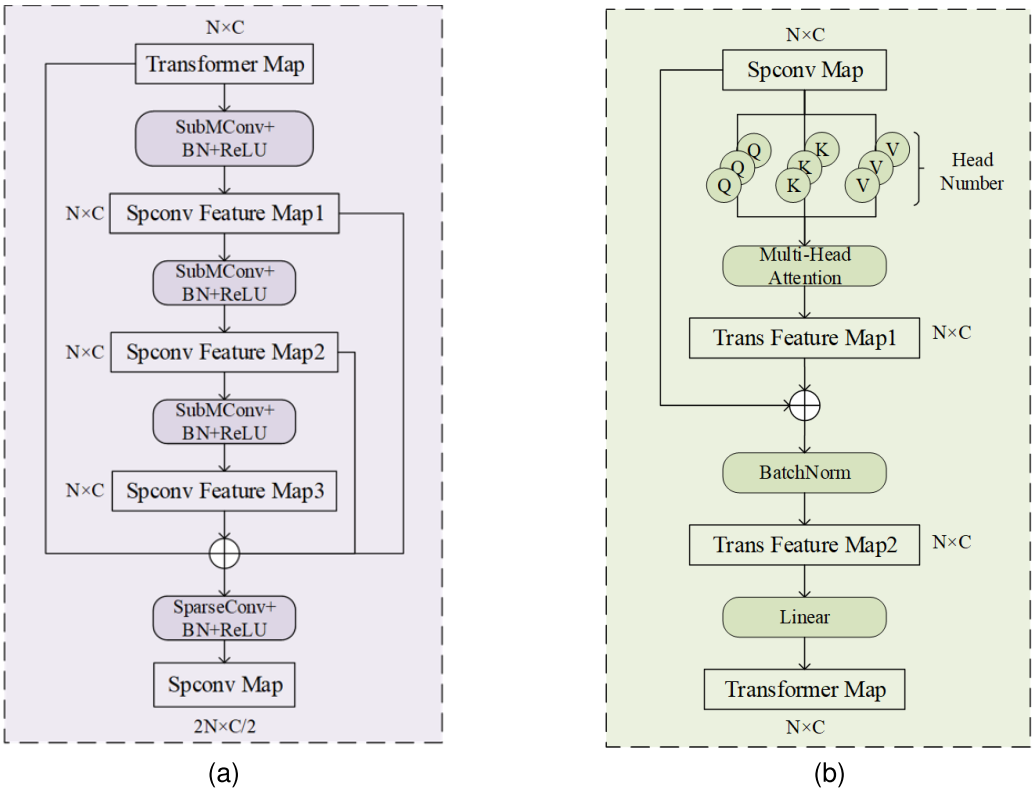

Fig. 5. Structures of blocks. (a) SpConv block. (b) Transformer block.

TABLE I
QUANTITIES OF EACH TREE SPECIES DATASET

Dataset	Quantities	Country	Scanner
Spruce(Picea glauca and Picea abies) [52]	174	Canada	IIrisLR/Leica
Aspen(Populus tremuloides) [52]	162	Canada	IIrisHD/IIrisLR
Poplar(Populus deltoides and Populus angustifolia) [52]	80	Canada	IIrisHD/IIrisLR
Birch(Betula pendula) [52]	100	Finland	Leica
Pine(Pinus resinosa, Pinus contorta,and Pinus sylvestris) [52]	220	Canada/Finland	IIrisHD/Leica
Maple(Acer saccharum) [52]	169	Canada	IIrisHD
Larch	100	*Simulated	–
Tropical tree [53]	61	Cameroon	–
Total	1066	–	–

obtained by the attention mechanism to obtain the transformer map after the linear layer.

IV. RESULTS AND DISCUSSION

In this section, SPTNet is proved by extensive experiments. Section IV-A describes the dataset and data augmentation processing method used for the experiments. Section IV-B describes the model training. Section IV-C presents the segmentation results under different tree species with different datasets and compares the results under different methods and ablation experiments. Extensive experiments are conducted on seven tree species datasets and a large tropical tree dataset.

A. Dataset

1) Dataset Description: Fig. 6 shows the example of basic information from eight datasets. Specifically, the following tree datasets are included Spruce, Aspen, Poplar, Birch, Pine, Maple, Larch, and a large tropical tree dataset. The Larch dataset is a simulated dataset produced by simulation software, and the rest of the data are manually collected. The data were collected manually from Canada, Finland, and Cameroon, which were obtained through different sensor acquisitions. More specific information can be found in Table I.

The first six datasets [Fig. 6(a)–(f)] each contain six different tree species [52]. The dataset was derived from real scanned forest sample plot data. After the forest sample plot data were segmented by single wood, all the single wood data were manually labeled. The seventh [Fig. 6(g)] dataset is the point cloud data of trees, which is simulated by the software. Due to the huge volume and complex structure of point cloud data, manual annotation is laborious. Some fields are difficult to get data or have less available data. Therefore, to assist the data-driven type of neural networks, a combination of simulated and real data has been used in recent years to help network training. In this study, single tree models were developed using OnyxTree software. OnyxTree is a commercial

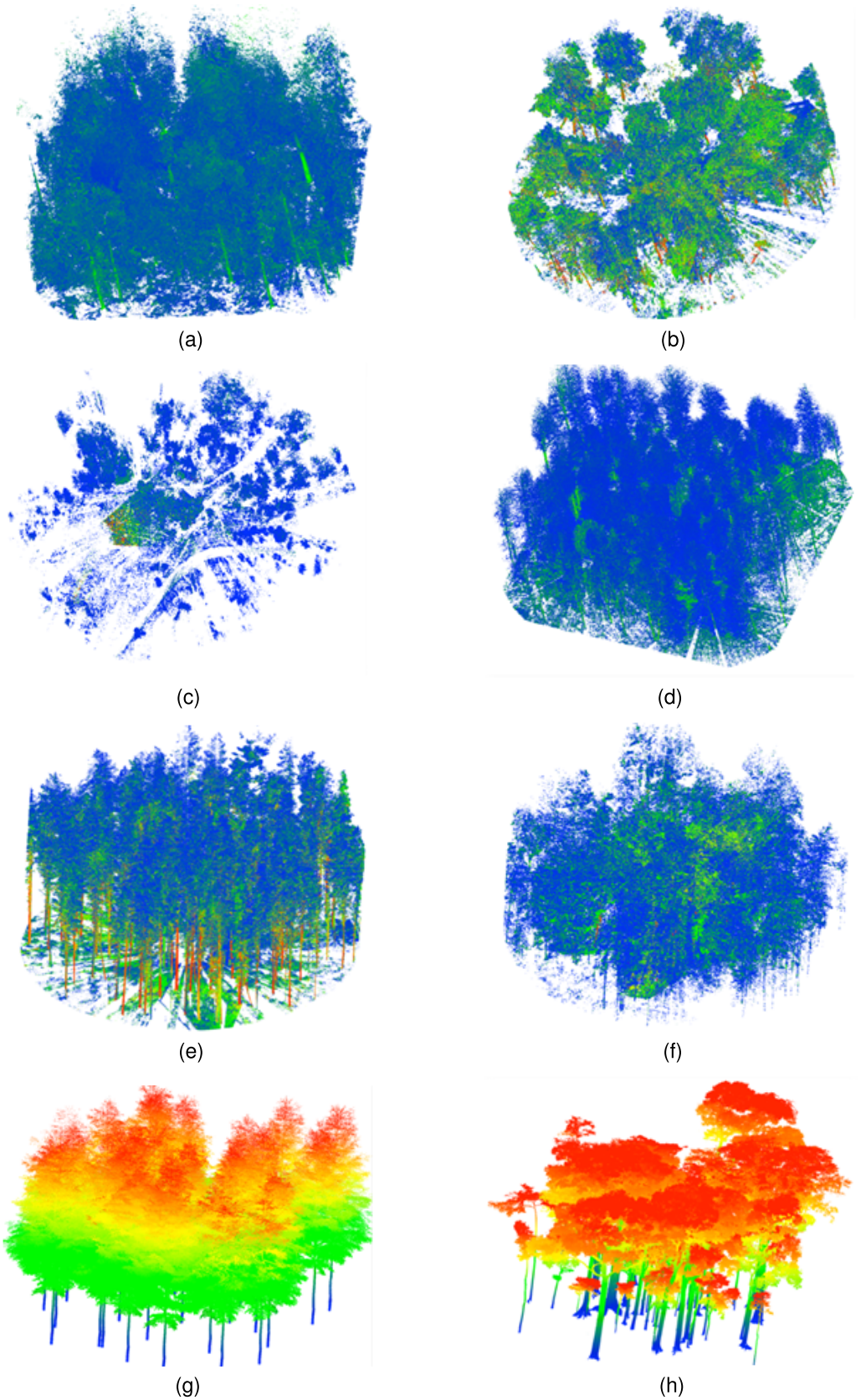

Fig. 6. Visualization of different tree species datasets. (a) Spruce. (b) Aspen. (c) Poplar. (d) Birch. (e) Pine. (f) Maple. (g) Larch (simulated). (h) Tropical tree.

pure computer simulation software dedicated to vegetation, which can simulate single tree models of various vegetation, including bamboo, flowers, grasses, palms, broadleaf trees, and conifers. Further point cloud data were produced by the

software package Helios++ [54]. During the point cloud simulation, the Riegl VZ-1000 LiDAR sensor was used to simulate the scanning of a quadrat. A total of five sites at the center and four corner points of the conventional quadrant were used to scan the already generated 3-D model of the trees. Finally, 100 simulated single trees were made, which include larch trees, considering the balanced sample size.

The last dataset [Fig. 6(h)] contains 61 large tropical trees. The points of woody components and foliage components have been manually marked. This LiDAR data based on tropical trees and their destructive references are copyrighted by LeWoS of the AMAP France laboratory. This dataset includes the eastern flank of 61 large tropical trees from Cameroon. Also, it has been used to estimate the biomass of large tropical trees and to calibrate anisotropic velocity models from the TLS point cloud. The dataset covers a total of 15 different species, which include Annickia chlorantha (Oliv.) Setten & Maas (3), Baphia leptobotrys Harms (3), Cylicodiscus gabunensis Harms (5), Duboscia macrocarpa Bocq. (2), Entandrophragma cylindricum (Sprague) Sprague (2), Eribroma oblongum (Mast.) Pierre ex A. Chev. (4), Erythrophleum suaveolens (Guill. & Perr.) Brenan (5), Macaranga barteri Müll.Arg. (2), Mansonia altissima (A. Chev.) A. Chev. (3), Pentachletra macrophylla Benth. (1), Petersianthus macrocarpus (P. Beauv.) Liben (6), Pterocarpus soyauxii Taub. (6), Pycnanthus angolensis (Welw.) Warb. (4), Terminalia superba Engl. & Diels (9), and Triplochiton scleroxylon K. Schum. (6). The number in parentheses indicates the number of single trees belonging to the species. Tree heights ranged from 8.7 to 53.6 m, with a mean of 33.7 ± 12.4 m. DBH ranged from 10.8 to 186.6 cm, with a mean of 58.4 ± 41.3 cm. See [53, Table 2] for a full description of the trees.

2) Data Augmentation: Data augmentation can achieve the effect of increasing the training sample with limited data, which in turn improves the robustness of the model. A large number of parameters are required in neural networks and getting a neural network to work requires a large amount of data to train. However, in many practical tasks, it is difficult to have enough data to complete these.

In this study, seven single tree species datasets and one large tropical species dataset are used to prove the method. Compared with other well-established deep learning methods in other fields, most of them are learned by a large number of data-driven models. However, there is no large amount of data for us to train the model in our research. Hence, to improve the generalization ability of the model, we make full use of the available data for data augmentation operations. Data augmentation can help us prevent overfitting and avoid the model falling into local optimal solutions.

Traditional data augmentation methods include flipping, rotating, scaling, random cropping, panning, adding noise, and so on. These data augmentation methods are only considered from a model perspective, but not from a practical view. It is unreasonable to flip and randomly cut tree point clouds. Because there will be no trees with trunks on top and a canopy underneath, and there will be no trees that grow only halfway.

Hence, a more realistic approach is adopted to data augmentation.

1) A random sampling of the flattened data is performed, thus obtaining a sparse point cloud. In this way, the coordinates and density of the point cloud vary greatly, but the shape of the tree remains the same. This method can improve the robustness of the model.
2) The rotation angle of the tree is set to less than 45°. This method not only can achieve the effect of data enhancement but also is in line with the reality of the situation.

B. Model Training

1) Device and Hyperparameter Selection: The device was an NVIDIA GeForce RTX 3090 graphics card configured with 24 GB. After a lot of experimental parameters tuning and taking into account the memory limitations of the device itself, the following hyperparameter settings are used through the experiments: Voxel_size = 0.02 m, Learning_rate = $1e^{-5}$, and Batch_size = 4. Regarding the optimizer in the network, both SGD and Adam are tried. Compared with SGD and Adam, the Adam optimizer tends to fail to converge in several training sessions, and the convergence speed of Adam is significantly slower than that of SGD. Also, a large amount of data shows that the SGD optimizer is more suitable for tasks in the field of computer vision, Adam works better in the field of NLP or reinforcement learning [55]. In our network, the SGD optimizer is used. The parameters of Adam are set as follows: Momentum = 0.9 and Weight_decay = $1e^{-8}$. In the transformer block, there are eight heads of self-attention and reuse the transformer block six times within a block. This coincides with the earliest transformer article.

2) Loss Function and Model Convergence: Our study addressed the woody and foliage components separation problem for tree point clouds. It belongs to the tree point cloud segmentation task and is a binary classification task. Therefore, the binary cross-entropy loss (BCE-Loss) is better as the loss function for our task. Unlike the ordinary cross-entropy function, BCE-Loss is specifically designed to deal with dichotomous problems. Before using BCE-Loss, the output will be planned to 0–1 by the sigmoid activation function layer. BCE-Loss is calculated as follows:

$$L_{\mathrm{BCE}} = -f_{\mathrm{GT}} \cdot \log(f_{\mathrm{Pred}}) - (1 - f_{\mathrm{GT}}) \cdot \log(1 - f_{\mathrm{Pred}}) \quad (8)$$

where f_{GT} represents the positive sample ground-truth value and f_{Pred} represents the positive sample predicted value.

C. Segmentation Result and Accuracy Evaluation

1) Accuracy Evaluation Metric: To evaluate the accuracy of separating woody and foliage components of tree point clouds, there are several accuracy evaluation metrics. OA, foliage components sorting accuracy (FA), woody components sorting accuracy (WA), Kappa coefficient, coordinated mean of model accuracy and recall (F1_score), and the mIoU. The calculation equations are as follows.

1) *OA, FA, and WA:*

$$\mathrm{OA} = \frac{\mathrm{PV}_f + \mathrm{PV}_w}{\mathrm{GT}_f + \mathrm{GT}_w} \quad (9)$$

TABLE II
CONFUSION MATRIX

Classes	Foliage (GT)	Wood (GT)
Foliage (Pred)	TP=FA	FN
Wood (Pred)	FP	TN=WA

$$\mathrm{FA} = \frac{\mathrm{PV}_f}{\mathrm{GT}_f} \tag{10}$$

$$\mathrm{WA} = \frac{\mathrm{PV}_w}{\mathrm{GT}_w} \tag{11}$$

where PV_f and PV_w represent the correct woody and foliage components in the model prediction results, respectively, and GT_f and GT_w represent the true values of woody and foliage components in the point set, respectively.

2) *Kappa:* First, a confusion matrix is constructed. The details of the confusion matrix are shown in Table II, where GT denotes the ground truth and Pred denotes the predicted results of the model. The parameters in the confusion matrix are expressed in the form of probabilities, and the parameters are calculated, as shown in Table II.
The Kappa is calculated according to the following formula:

$$\mathrm{Kappa} = \frac{\mathrm{OA} - P_c}{1 - P_c} \tag{12}$$

$$P_c = \frac{(\mathrm{TP}+\mathrm{FN}) \cdot (\mathrm{TP}+\mathrm{FP}) + (\mathrm{FP}+\mathrm{TN}) \cdot (\mathrm{FN}+\mathrm{TN})}{\mathrm{TP}+\mathrm{FP}+\mathrm{TN}+\mathrm{FN}} \tag{13}$$

where P_c is the expected probability value.

3) *$F1$_score:* $F1$_score can be computed as follows:

$$F1\_\mathrm{score} = \frac{2 \cdot \mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}} \tag{14}$$

$$\mathrm{precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{15}$$

$$\mathrm{recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{16}$$

4) *mIoU:* The mIoU calculation method is as follows:

$$\mathrm{mIoU} = \frac{\sum_{i=1}^{n} \frac{TP}{FN+FP+TP}}{n}. \tag{17}$$

2) Segmentation Results: Among the single tree point cloud data in the eight datasets, 70% of each of these datasets are selected as the training set. The rest of the data are generally used for the validation set and the test set. Voxel coordinates are combined with two geometric features—MDC and NDO—to form an $N \times 8$ hash table as the input to the neural network. SPTNet trains under each of the eight datasets. In addition, to balance sample size and segmentation difficulty, SPTNet trained 50 epochs on the Pine dataset and 100 epochs on the other datasets. We experimented with increasing the number of training rounds, but the excessive number of training rounds did not lead to an improvement in accuracy, but rather to overfitting.

Our proposed model segments the tree point cloud. Table III shows in detail the quantitative evaluation metrics of our method. On the Spruce dataset, the mean OA is 92.20%, the mean value of FA is 99.18%, and the mean value of WA is 97.62%. The F1_score of this species is 0.923, and the mIoU is 93.70%. On the Aspen dataset, the mean OA is 90.80%. The mean value of FA is 98.67% and the mean value of WA is 89.50%. The F1_score of this species is 0.907, and the mIoU is 92.95%. On the Poplar dataset, the mean OA is 86.35%, the mean value of FA is 98.78%, and the mean value of WA is 88.80%. The F1_score of this species is 0.855, and the mIoU is 85.70%. On the Birch dataset, the mean OA is 98.05%, the mean value of FA is 99.74%, and the mean value of WA is 90.81%. The F1_score of this species is 0.912, and the mIoU is 96.92%. On the Pine dataset, the mean OA is 96.01%, the mean value of FA is 99.95%, and the mean value of WA is 99.64%. The F1_score of this species is 0.959, and the mIoU is 92.20%. On the Maple dataset, the mean OA is 97.34%, the mean value of FA is 99.51%, and the mean value of WA is 97.29%. The F1_score of this species is 0.973, and the mIoU is 94.82%. On the Larch dataset, the OA was 92.43%. The mean value of FA was 97.21% and the mean value of WA was 78.66%. The F1_score of this species was 0.791, and the mIoU was 78.21%. On the large tropical tree dataset, the OA was 94.69%. The mean value of FA was 97.10% and the mean value of WA was 91.09%. The F1_score of this species was 0.896 with the mIoU of 89.96%.

TABLE III
EVALUATION METRICS OF SPTNET

Dataset	OA(%)	FA(%)	WA(%)	$F1\_score$	mIoU(%)
Spruce	92.20	99.18	97.62	0.923	93.70
Aspen	90.80	98.67	89.50	0.907	92.95
Poplar	86.35	98.78	88.80	0.855	85.70
Birch	98.05	99.74	90.81	0.912	96.92
Pine	96.01	99.95	99.64	0.959	92.20
Maple	97.34	99.52	97.29	0.973	94.82
Larch	92.43	97.21	78.66	0.791	78.21
tropical	94.69	97.10	91.09	0.896	89.96

The visualization of the segmentation results of our proposed network for the eight datasets is shown in Fig. 7. In Fig. 7, our method can perform the tree point cloud dendritic separation task better in all datasets. Different tree species have different tree structures, and our method can be generalized to most tree species. For some trees with the irregular structure of single wood data [such as Fig. 7(c)], our method can be accurate in that the woody component and foliage component can be divided. Fig. 8 shows the detailed figure of the different tree species after segmentation. Our method gives better segmentation results on all tree species datasets. One of the main difficulties in the segmentation of tree point clouds lies in the intersection of woody and foliage components. Our method achieves good segmentation results both in wood-only locations and wood-foliage locations.

3) Performance Comparison: Our approach is expected to be of maximum generality, so performance on multiple datasets with different species of trees is even more important. The diversities of tree species and differences in wood and foliage component structure are considered and finally conducted experiments on three representative datasets: a

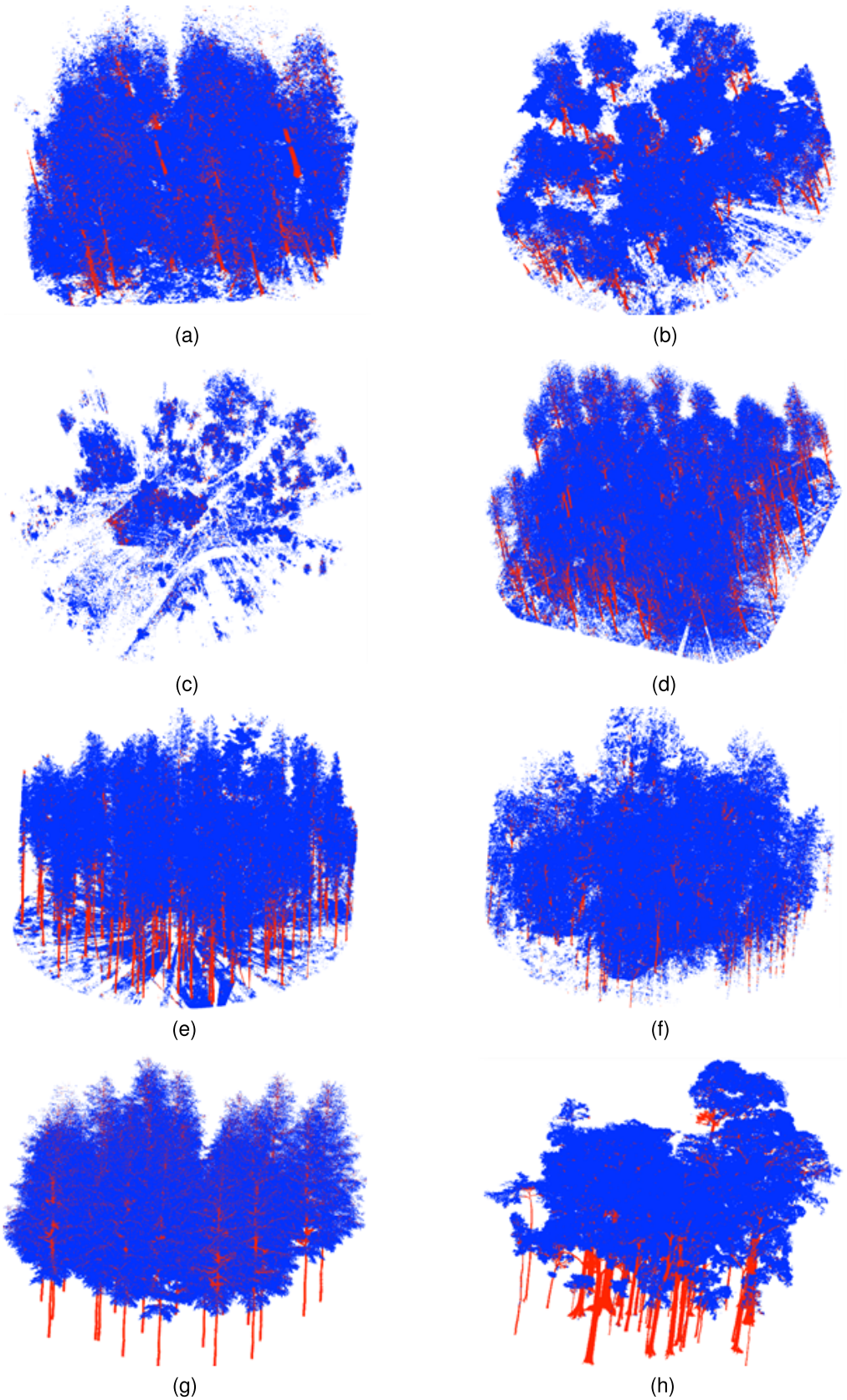

Fig. 7. Visualization of woody and foliage segmentation results on different species. (a) Spruce. (b) Aspen. (c) Poplar. (d) Birch. (e) Pine. (f) Maple. (g) Larch (simulated). (h) Tropical tree.

Tropical Multispecies dataset, a broadleaf forest (Birch), and a coniferous forest (Larch). A total of three existing branch and leaf separation methods were selected for comparison, LWCLF, LeWos, and FWCNN. Among them, FWCNN is

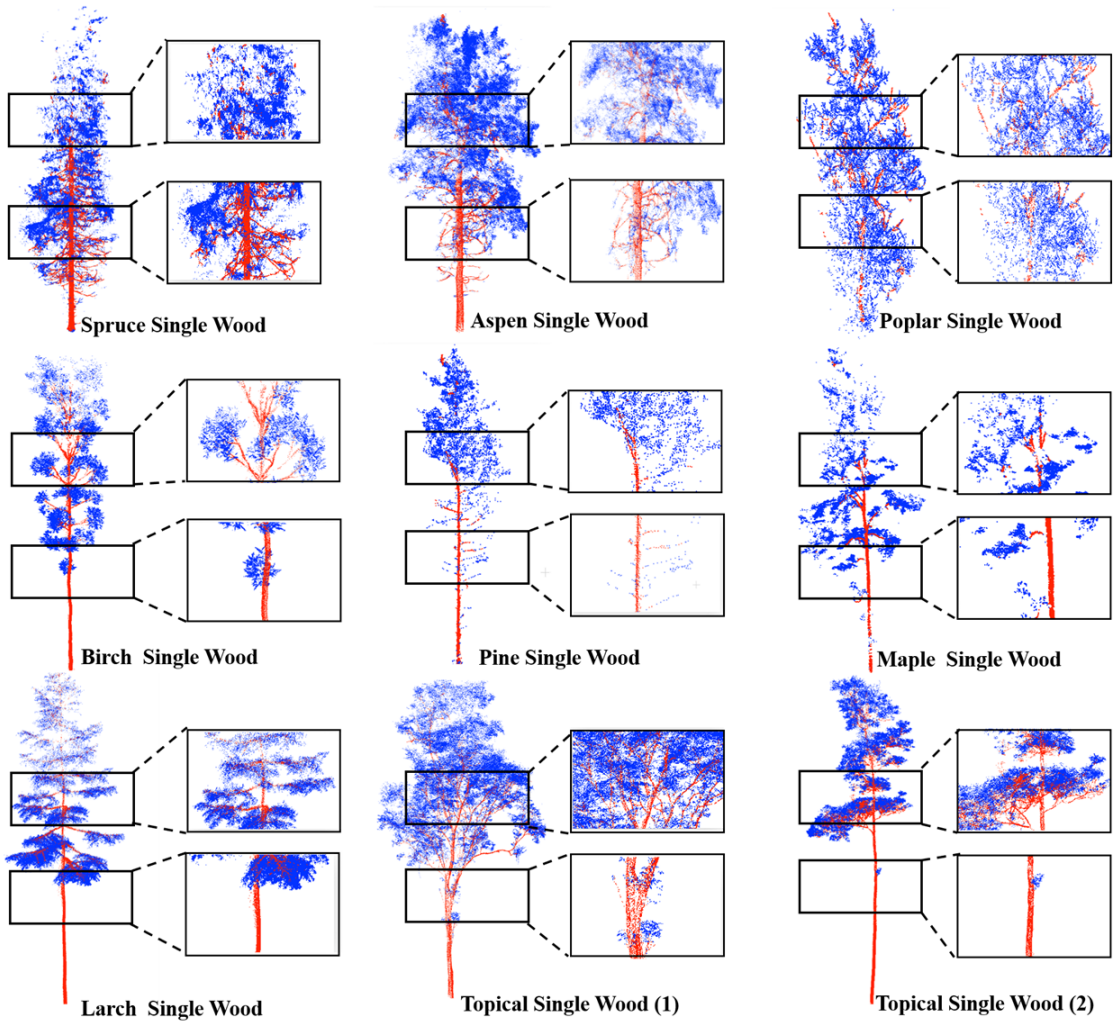

Fig. 8. Segmentation details of different tree species.

a wood and foliage component separation method based on PointNet. At the same time, FWCNN is based on the combination of physical mechanisms and deep learning, which is the most effective existing wood and foliage component separation method. LWCLF and LeWoS are two methods based on traditional feature extraction threshold segmentation. Table IV shows the segmentation accuracy of these methods on different datasets. As can be seen from the data, SPTNet shows the best segmentation results, especially in the Larch dataset, with all evaluation metrics leading the other methods across the board. Meanwhile, in the tropical multitree dataset, SPTNet exhibits 98.05% of the OA and 96.92% mIoU. However, it is worth noting that FWCNN similarly demonstrates competitive segmentation results, especially in wood segmentation accuracy in the Tropical Multispecies dataset and the Birch dataset. However, its overall accuracy and mIoU are slightly inferior to SPTNet, while LWCLF and LeWoS did not show better segmentation results in complex scenes and confusing scenes.

4) Ablation Experience: To test the effectiveness of our method and the contribution of each block, a series of ablation experiments is conducted.

TABLE IV
QUANTITATIVE COMPARISON BETWEEN THE DIFFERENT METHODS

Method	Dataset	OA(%)	FA(%)	WA(%)	mIoU(%)
LWCLF (2018)	tropical	81.05	78.82	84.27	68.10
	Birch	78.96	84.13	77.69	84.11
	Larch	74.05	78.82	71.38	48.59
LeWoS (2020)	tropical	88.70	79.65	82.99	63.89
	Birch	86.72	91.16	89.72	87.34
	Larch	75.66	77.15	73.73	60.29
FWCNN (2020)	tropical	94.26	94.44	90.63	89.24
	Birch	96.25	98.79	**92.63**	95.89
	Larch	89.81	90.08	**86.80**	77.95
SPTNet(Ours)	tropical	**94.69**	**97.10**	**91.09**	**89.96**
	Birch	**98.05**	**99.74**	90.81	**96.92**
	Larch	**92.43**	**97.21**	78.66	**78.21**

a) Backbone: To verify the effectiveness of the SpConv in our network with the fusion of transformer blocks, the experiment is set to train with the same data but with multiple backbone replacements. A lot of backbones are used P to perform the ablation experiments, including PointNet++, DGCNN, Point Transformer, PointBert, and PointMLP. PointNet++ is a point-based convolution method; DGCNN is a graph convolution-based method, and Point

TABLE V
QUANTITATIVE COMPARISON BETWEEN THE DIFFERENT BACKBONE

Backbone	dataset	OA(%)	mIoU(%)
PointNet++ (2017)	Tropical	94.26	83.81
	Birch	96.25	88.77
	Larch	89.80	65.51
DGCNN (2019)	Tropical	80.27	78.23
	Birch	81.13	82.61
	Larch	86.47	57.03
Point Transformer (2021)	Tropical	92.33	88.02
	Birch	96.97	90.69
	Larch	90.36	67.72
PointBert (2022)	Tropical	94.12	83.97
	Birch	96.65	89.89
	Larch	90.20	67.57
PointMLP (2022)	Tropical	92.33	73.88
	Birch	89.38	81.24
	Larch	84.70	47.06
SPTNet(Ours)	Tropical	**94.69**	**89.96**
	Birch	**98.05**	**96.92**
	Larch	**92.43**	**78.21**

TABLE VI
EFFICIENCY COMPARISON BETWEEN THE DIFFERENT BACKBONE

Backbone	Train Speed *Converge Epoch	Model Weight parameter size	Inference speed
PointNet++ (2017)	16s*245	95.23M	10950ms
DGCNN (2019)	15s*161	–	10113ms
Point Transformer(2021)	45s*90	222.91M	25314ms
PointBert (2022)	17s*102	526.2M	60650ms
PointMLP (2022)	10s*198	192.28M	22109ms
SPTNet(Ours)	**8s*48**	**29.49M**	**3391ms**

Transformer and PointBert are transformer-based methods that perform state-of-the-art in each task. PointMLP is a method based on the pure MLP structure. Also, Table V shows the performance of the separation under the different backbones.

Similarly, one representative multispecies dataset, one broadleaf forest dataset (Birch), and one coniferous forest dataset (Larch) are selected for our experiments. As can be seen from the data in Table V, summarizing the three datasets comparing the mentioned methods, SPTNet achieves the optimal segmentation results for all of them. There is no lack of promising methods in our comparison, and Point Transformer and PointBert achieved more than 80% mIoU in both the Tropical Multispecies dataset and the Birch dataset, which suggests that the Transformer-based network performs well for the task of branch and leaf separation. However, to our surprise, PointNet++ surpassed our expectations, achieving 94.26% OA and 83.81% mIoU, because simple MLP-based networks will show relatively low results when it comes to complex problems. Even in the more difficult to segment coniferous Larch dataset, 89.80% OA and 65.15 mIoU were achieved. It is the neighborhood focus of PointNet++ that plays a better role. This indirectly shows the importance of in-neighborhood features for point cloud segmentation. More detailed comparison data can be viewed in Table V.

At the same time, we focused on comparing the training speed of the network under the different backbones. Table VI shows quantitatively the comparison between the efficiencies. Comparing these methods, Pointnet++, DGCNN, and PointBert all take about 15 s to train an epoch, Point Transformer takes 45 s, PointMLP takes 10 s, and our method takes only 8 s. It guaranteed the same Batch_size for comparison, PointNet++ requires 245 epochs to converge, DGCNN requires 161 epochs, and Point Transformer requires 90 epochs. PointBert requires 102 epochs to converge, and PointMLP requires 198 epochs to converge, while our method requires only 48 epochs to converge. Inference speed is something we are keen to consider. The size of the input data as well as differences in equipment also affect inference time, so purely absolute time has no comparative value. Thus, the efficiency of the models was compared in two aspects, the size of the weights of the models and the inference time required on the same dataset. After reviewing the data, we found that model size and inference speed are generally positively related and that models are readily available. The weight model sizes obtained after training different models are compared in Table VI. The weight files output by this method are much smaller than those of other comparisons and even less than 1/3 of those of the lightweight PointNet. The reason is that voxelization as well as SpConv work together to cause the network to reason efficiently. Meanwhile, the inference speed is an important indicator of being able to evaluate efficiency. This test uses a tree with about 50 000 points as a sample, and SPTNet takes about 3391 ms to reason about it. Other methods, such as PointNet++ and DGCNN, require more than 10 000 ms to predict, with the PointBert method requiring more than 60 000 ms. A simple analysis of the reason is that SPTNet is a voxel-based network, which mostly reduces the memory consumption as compared to the point-based approach. Meanwhile, the properties of SpConv result in it being able to complete inference faster, and the lightweight transformer improves the performance without taking up more computing time. In summary, our method has both a fast training speed and an inference speed. Table VII shows the comparison of the segmentation results after replacing the backbone in the network. Both PointNet++ and PointBert are good at separating where woody and foliage components are not interlaced. However, there are few sorting abilities at the intersection of woody and foliage components. Our method allows for finer segmentation in more difficult point cloud conditions. Also, our method yields good segmentation results.

b) Block: Furthermore, to verify which block plays a role in our proposed network, the SpConv block and the transformer block are removed in the network separately and perform the separation task. Table VIII shows the quantitative comparison after removing the different blocks.

As can be seen in Table VIII, the SpConv block has a greater impact on the accuracy of the network. After dropping the SpConv block, a larger decrease in accuracy occurs. The reason for this is that SpConv is significantly better than pure transformer methods in terms of local feature extraction. Also, the judgment of whether it is a woody component or a foliage component depends more on the geometric features in the neighborhood. The network with the transformer block removed still maintains a good level of separation but is still slightly lower than the combination

TABLE VII
COMPARISON OF THE SEGMENTATION RESULTS OF DIFFERENT BACKBONE

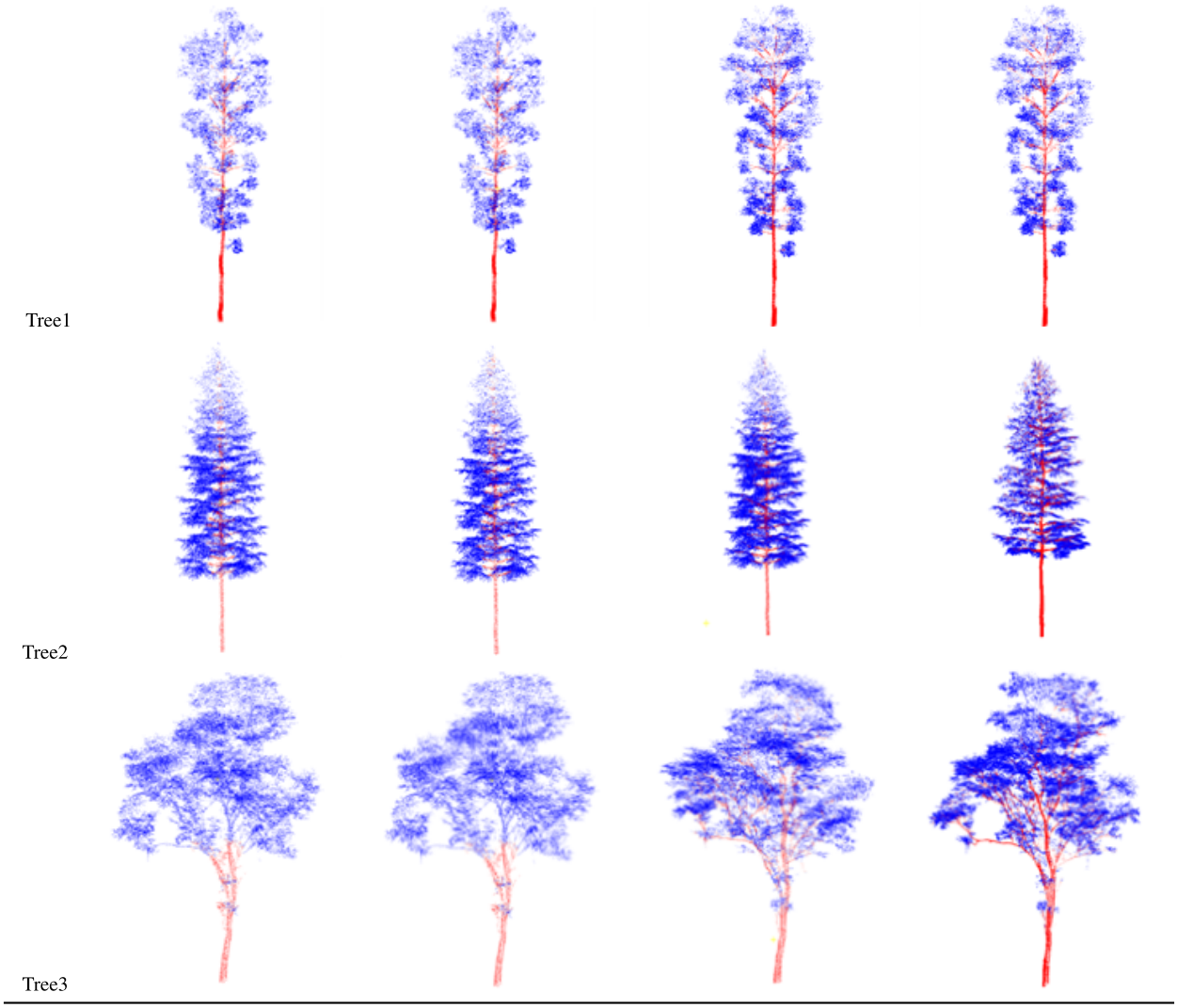

TABLE VIII
QUANTITATIVE COMPARISON AFTER REMOVING THE DIFFERENT BLOCKS

Module	Dataset	OA(%)	mIoU(%)
No sparse block	tropical	90.36	88.02
No transformer block	tropical	93.98	87.42
Completely	tropical	**94.69**	**89.96**

TABLE IX
QUANTITATIVE COMPARISON OF DIFFERENT FEATURES INPUT

Feature	Dataset	OA(%)	mIoU(%)
Only Coord	tropical	91.68	83.74
Coord+MDC	tropical	92.89	85.04
Coord+NDO	tropical	93.56	86.52
Coord+MDC+NDO	tropical	94.69	89.96

of SpConv and transformer. The reason is that the transformer can better aggregate global features, which makes the network learning field of feeling expanded and the performance of the network.

TABLE X
QUANTITATIVE COMPARISON OF DIFFERENT MASKED RATES

Masked rate	Dataset	OA(%)	mIoU(%)
Complete Data	tropical	94.69	89.96
Masked 20%	tropical	91.07	90.65
Masked 40%	tropical	92.24	82.05
Masked 60%	tropical	94.06	77.52
Masked 80%	tropical	92.01	54.52

c) Point features: To verify whether the two physical features added would assist in the classification results, the corresponding ablation experiments are conducted. In addition, SPTNet is trained by only point coordinates, coordinates and MDC, coordinates and NDO, and all features. All parts remain the same except for the input. Table IX shows the quantitative comparison (OA and mIoU) of the separation results for different input features.

Table IX shows that the addition of the MDC and NDO features has a positive impact on the segmentation results.

TABLE XI
VISUALIZATION OF DIFFERENT MASKED RATES' SEGMENTATION RESULTS

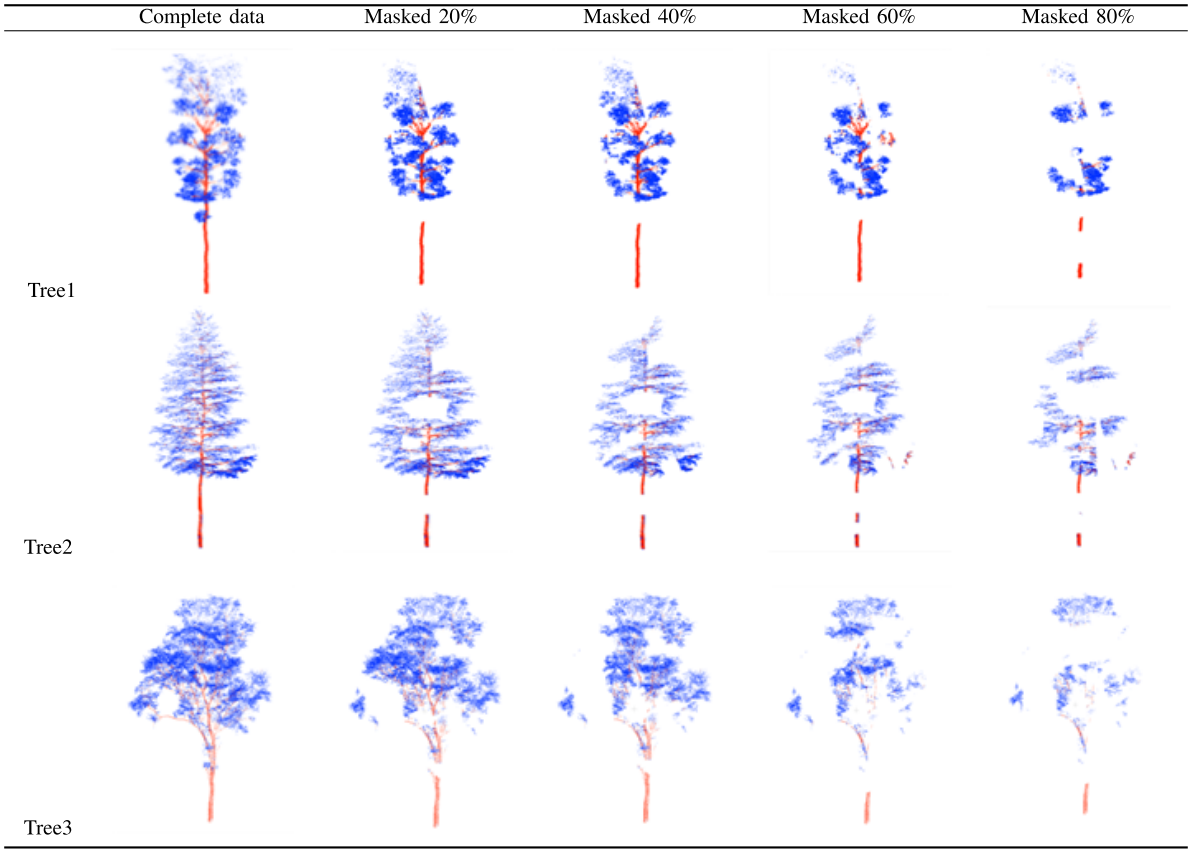

TABLE XII
COMPARISON OF RESULTS FOR ADAPTIVE RADIUS AND CONSTANT RADIUS

Neighborhood radius	OA(%)	mIoU(%)
0.1	**95.51**	87.64
0.2	93.98	86.79
0.3	93.16	84.02
0.4	94.22	83.47
0.5	92.10	82.53
Adaptive radius	94.69	**89.96**

NDO features have a greater impact on the results than MDC. Also, at the same time, the speed efficiency has slowed down.

d) Robustness testing: Actually, the point cloud data obtained in real cases are usually incomplete. Both TLS and airborne laser scanning are subject to terrain occlusion. Different percentages of masking on individual tree sample point clouds are performed to verify the robustness of our method. Also, the tree point cloud samples are masked at 20%, 40%, 60%, and 80% randomly. Other conditions were kept constant for the test. Table X shows quantitatively the tree point cloud woody and foliage components segmentation for different masked rates. The large tropical tree dataset still has 91.07% OA and 90.65% mIoU after 20% masking. Also, it has 92.24% OA and 82.05% mIoU after 40% masked. The large tropical tree dataset has 94.06% OA and 77.52% mIoU after 60% masked. Also, it has only 92.01% OA and 54.52% mIoU after 80% masked. According to Table X, our method can show good segmentation accuracy for all masked rates less than 60%. This further demonstrates the robustness of our method. Too large a proportion of occlusion leads to a loss of information in most neighborhoods and a significant loss of information at the edges. This is probably the main reason for the degradation of segmentation accuracy. Table XI shows the segmentation visualization results with different masked rates. Our network still has good segmentation results at large masked rates.

Furthermore, in order to verify the effectiveness of the adaptive radius method corresponding to the MDC and NDO features, a comparison test is set up. To verify the effectiveness of adaptive radii, multiple fixed radii were used for comparison tests. A total of three radius sizes were selected from 0.1 to 0.5 at intervals of 0.2 for comparison with the adaptive radius. Table XII shows that the use of adaptive radius can slightly improve the accuracy of wood and foliage component separation. Utilizing an adaptive approach will consume more time, and meanwhile, a fixed radius value that performs relatively well can be found after several manual tests. However, the adaptive approach does not have any redundant testing, and besides, human a priori knowledge cannot accurately localize the optimal fixation radius at once. It is a difficult tradeoff but goes for a more automated approach with acceptable time consumption.

V. Conclusion

In this article, we proposed a fully automatic method for tree point cloud separation between woody and foliage components called SPTNet. First, two blocks based on geometric features are proposed to assist in segmentation. Different adaptive algorithms are used for each of the two physical features to reduce human influence. Second, SPTNet was proposed as a network based on a combination of SpConv and transformer to accomplish the segmentation task, which is proved to be effective by experiments on multiple datasets with multiple tree species. Our method reached 94.69% for OA and 89.96% for mIoU on a large tropical multispecies dataset. Also, it takes only 8 s to train an epoch and 48 epochs to converge. Compared with other traditional methods (LeWoS and LWCLF) and deep learning methods (FWCNN), our method achieves state-of-the-art in terms of accuracy and efficiency. Nevertheless, the method is sensitive to the voxelization resolution parameter. Different voxel resolutions can lead to fluctuations in the results. Smaller voxel resolutions inevitably require larger computational resources.

In the future, there is a plan to extend the method to the sample site scale. The task of tree separation is accomplished in the presence of multiple trees within the scene, even with ground points or other ground noise. In addition, SPTNet would be tested on more complex and difficult data, such as airborne LiDAR data to validate its generalization. Also, it is hoped that the method will overcome the problem of tree shading or even infer tree trends based on known incomplete data. Furthermore, SPTNet would be extended to more domains, such as semantic segmentation of urban scenes.

References

[1] X. Chen, X. Zhang, Y. Zhang, T. Booth, and X. He, "Changes of carbon stocks in bamboo stands in China during 100 years," *Forest Ecol. Manage.*, vol. 258, no. 7, pp. 1489–1496, Sep. 2009.

[2] W. Li, Z. Niu, S. Gao, N. Huang, and H. Chen, "Correlating the horizontal and vertical distribution of LiDAR point clouds with components of biomass in a *Picea crassifolia* forest," *Forests*, vol. 5, no. 8, pp. 1910–1930, Aug. 2014.

[3] S. Xia, C. Wang, F. Pan, X. Xi, H. Zeng, and H. Liu, "Detecting stems in dense and homogeneous forest using single-scan TLS," *Forests*, vol. 6, no. 12, pp. 3923–3945, Oct. 2015.

[4] T.-M. Yen, Y.-J. Ji, and J.-S. Lee, "Estimating biomass production and carbon storage for a fast-growing Makino bamboo (*Phyllostachys makinoi*) plant based on the diameter distribution model," *Forest Ecol. Manage.*, vol. 260, no. 3, pp. 339–344, Jun. 2010.

[5] G. Yan et al., "Review of indirect optical measurements of leaf area index: Recent advances, challenges, and perspectives," *Agricult. Forest Meteorol.*, vol. 265, pp. 390–411, Feb. 2019.

[6] G. Zheng, L. Ma, W. He, J. U. H. Eitel, L. M. Moskal, and Z. Zhang, "Assessing the contribution of woody materials to forest angular gap fraction and effective leaf area index using terrestrial laser scanning data," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1475–1487, Mar. 2016.

[7] L. Ma, G. Zheng, J. U. H. Eitel, T. S. Magney, and L. M. Moskal, "Determining woody-to-total area ratio using terrestrial laser scanning (TLS)," *Agricult. Forest Meteorol.*, vols. 228–229, pp. 217–228, Nov. 2016.

[8] X. Zhao, Q. Guo, Y. Su, and B. Xue, "Improved progressive TIN densification filtering algorithm for airborne LiDAR data in forested areas," *ISPRS J. Photogramm. Remote Sens.*, vol. 117, pp. 79–91, Jul. 2016.

[9] W. Zhang et al., "An easy-to-use airborne LiDAR data filtering method based on cloth simulation," *Remote Sens.*, vol. 8, no. 6, p. 501, Jun. 2016.

[10] C. Vega et al., "PTrees: A point-based approach to forest tree extraction from LiDAR data," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 33, pp. 98–108, Dec. 2014.

[11] X. Xu, F. Iuricich, K. Calders, J. Armston, and L. De Floriani, "Topology-based individual tree segmentation for automated processing of terrestrial laser scanning point clouds," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 116, Feb. 2023, Art. no. 103145.

[12] H. Qin, W. Zhou, Y. Yao, and W. Wang, "Individual tree segmentation and tree species classification in subtropical broadleaf forests using UAV-based LiDAR, hyperspectral, and ultrahigh-resolution RGB data," *Remote Sens. Environ.*, vol. 280, Oct. 2022, Art. no. 113143.

[13] D. Wang, S. Momo Takoudjou, and E. Casella, "LeWoS: A universal leaf-wood classification method to facilitate the 3D modelling of large tropical trees using terrestrial LiDAR," *Methods Ecol. Evol.*, vol. 11, no. 3, pp. 376–389, Mar. 2020.

[14] B. Wu, G. Zheng, and Y. Chen, "An improved convolution neural network-based model for classifying foliage and woody components from terrestrial laser scanning data," *Remote Sens.*, vol. 12, no. 6, p. 1010, Mar. 2020.

[15] M. B. Vicari, M. Disney, P. Wilkes, A. Burt, K. Calders, and W. Woodgate, "Leaf and wood classification framework for terrestrial LiDAR point clouds," *Methods Ecol. Evol.*, vol. 10, no. 5, pp. 680–694, May 2019.

[16] X. Zhu et al., "Foliar and woody materials discriminated using terrestrial LiDAR in a mixed natural forest," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 64, pp. 43–50, Feb. 2018.

[17] L. Ma, G. Zheng, J. U. H. Eitel, L. M. Moskal, W. He, and H. Huang, "Improved salient feature-based approach for automatically separating photosynthetic and nonphotosynthetic components within terrestrial LiDAR point cloud data of forest canopies," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 2, pp. 679–696, Feb. 2016.

[18] S. Tao, Q. Guo, S. Xu, Y. Su, Y. Li, and F. Wu, "A geometric method for wood-leaf separation using terrestrial and simulated LiDAR data," *Photogramm. Eng. Remote Sens.*, vol. 81, no. 10, pp. 767–776, Oct. 2015.

[19] E. S. Douglas et al., "Finding leaves in the forest: The dual-wavelength Echidna LiDAR," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 4, pp. 776–780, Apr. 2015.

[20] Z. Li et al., "Separating leaves from trunks and branches with dual-wavelength terrestrial LiDAR scanning," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2013, pp. 3383–3386.

[21] H. Wei, G. Zhou, and J. Zhou, "Comparison of single and multi-scale method for leaf and wood points classification from terrestrial laser scanning data," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 3, pp. 217–223, Apr. 2018.

[22] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on X-transformed points," in *Proc. Neural Inf. Process. Syst.*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:53399839

[23] E. Ayrey and D. Hayes, "The use of three-dimensional convolutional neural networks to interpret LiDAR for forest inventory," *Remote Sens.*, vol. 10, no. 4, p. 649, Apr. 2018.

[24] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA. Red Hook, NY, USA: Curran Associates, 2017, pp. 5105–5114.

[25] J. Demantké, C. Mallet, N. David, and B. Vallet, "Dimensionality based scale selection in 3D LiDAR point clouds," in *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. XXXVIII-5/W12, pp. 97–102, 2011. [Online]. Available: https://isprs-archives.copernicus.org/articles/XXXVIII-5-W12/97/2011/

[26] T. Yun, F. An, W. Li, Y. Sun, L. Cao, and L. Xue, "A novel approach for retrieving tree leaf area from ground-based LiDAR," *Remote Sens.*, vol. 8, no. 11, p. 942, Nov. 2016.

[27] R. Ferrara, S. G. P. Virdis, A. Ventura, T. Ghisu, P. Duce, and G. Pellizzaro, "An automated approach for wood-leaf separation from terrestrial LiDAR point clouds using the density based clustering algorithm DBSCAN," *Agricult. Forest Meteorol.*, vol. 262, pp. 434–444, Nov. 2018.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, vol. 9351, 2015, pp. 234–241.

[29] D. Wang, M. Hollaus, and N. Pfeifer, "Feasibility of machine learning methods for separating wood and leaf points from terrestrial laser scanning data," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 2, pp. 157–164, Sep. 2017.

[30] J. Mao et al., "Voxel transformer for 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3164–3173.

[31] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4490–4499.

[32] A. Hamdi, S. Giancola, and B. Ghanem, "MVTN: Multi-view transformation network for 3D shape recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 1–11.

[33] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 945–953.

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

[35] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 6411–6420.

[36] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 922–928.

[37] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4866–4875.

[38] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual MLP framework," 2022, *arXiv:2202.07123*.

[39] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[40] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 10012–10022.

[41] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-BERT: Pre-training 3D point cloud transformers with masked point modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 19313–19322.

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

[43] Z. Dai et al., "CoAtNet: Marrying convolution and attention for all data sizes," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 3965–3977.

[44] M. Wortsman, G. Ilharco, S. Gadre, R. Roelofs, R. Gontijo-Lopes, and A. S. Morcos, "Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time," in *Proc. 39th Int. Conf. Mach. Learn.*, Baltimore, MD, USA, 2022, pp. 23965–23998.

[45] J. Zhang et al., "MiniViT: Compressing vision transformers with weight multiplexing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12135–12144.

[46] K. Turgut, H. Dutagaci, G. Galopin, and D. Rousseau, "Segmentation of structural parts of rosebush plants with 3D point-based deep learning methods," *Plant Methods*, vol. 18, no. 1, p. 20, Dec. 2022.

[47] Z. Ao et al., "Automatic segmentation of stem and leaf components and individual maize plants in field terrestrial LiDAR data using convolutional neural networks," *Crop J.*, vol. 10, no. 5, pp. 1239–1250, Oct. 2022.

[48] J. Zhou, H. Wei, G. Zhou, and L. Song, "Separating leaf and wood points in terrestrial laser scanning data using multiple optimal scales," *Sensors*, vol. 19, no. 8, p. 1852, Apr. 2019.

[49] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.

[50] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," 2017, *arXiv:1706.01307*.

[51] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, "Sparse convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 806–814.

[52] Z. Xi, C. Hopkinson, S. B. Rood, and D. R. Peddle, "See the forest and the trees: Effective machine and deep learning algorithms for wood filtering and tree species classification from terrestrial laser scanning," *ISPRS J. Photogramm. Remote Sens.*, vol. 168, pp. 1–16, Oct. 2020.

[53] S. Momo Takoudjou et al., "Using terrestrial laser scanning data to estimate large tropical trees biomass and calibrate allometric models: A comparison with traditional destructive approach," *Methods Ecol. Evol.*, vol. 9, no. 4, pp. 905–916, Apr. 2018.

[54] L. Winiwarter et al., "Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning," *Remote Sens. Environ.*, vol. 269, Feb. 2022, Art. no. 112772.

[55] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.