# Tree Classification in Complex Forest Point Clouds Based on Deep Learning

Xinhuai Zou, Ming Cheng, *Member, IEEE*, Cheng Wang, *Senior Member, IEEE*, Yan Xia, and Jonathan Li, *Senior Member, IEEE*

***Abstract*— Recently, the classification of tree species using 3-D point clouds has drawn wide attention in surveys and forestry investigations. This letter proposes a new voxel-based deep learning method to classify tree species in 3-D point clouds collected from complex forest scenes. The proposed method includes three steps: 1) individual tree extraction based on the density of the point clouds; 2) low-level feature representation through voxel-based rasterization; and 3) classification of tree species by a deep learning model. Two data sets of 3-D forest point clouds acquired by terrestrial laser scanning systems are used to evaluate the proposed method. The method achieves an average classification accuracy of 93.1% and 95.6% on the two data sets. Furthermore, in comparative experiments, the proposed method exhibits performance superior to that of the other 3-D tree species classification methods.**

***Index Terms*— Deep learning, point clouds, rasterization, terrestrial laser scanning (TLS), tree species classification.**

## I. Introduction

TREE species classification in forests has become increasingly important for forestry investigation, forestry planning, environmental protection, and statistics pertaining to forest resources [1]. Every year, governments and companies conduct forestry investigations, which consume a large amount of manpower and financial resources.

Terrestrial laser scanning (TLS) systems efficiently collect 3-D point cloud data from the surroundings with highly detailed representation of objects. The data are used in surveys, map drawings, and object measurements. This technology has the potential to reduce manpower consumption in forestry investigations. In this letter, we focus on the classification of tree species in plantation forests using 3-D point clouds.

Manuscript received June 19, 2017; revised October 3, 2017 and October 16, 2017; accepted October 17, 2017. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFC1401001 and in part by the National Natural Science Foundation of China under Grant 61371144 and Grant U1605254. *(Corresponding author: Ming Cheng.)*

X. Zou, M. Cheng, C. Wang, and Y. Xia are with the Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Information Science and Engineering, Xiamen University, Xiamen 361005, China (e-mail: xhzou@yahoo.com; chm99@xmu.edu.cn; cwang@xmu.edu.cn; xiayan@stu.xmu.edu.cn).

J. Li is with the Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Information Science and Engineering, Xiamen University, Xiamen 361005, China, and also with the Department of Geography and Environmental Management, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: junli@xmu.edu.cn).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LGRS.2017.2764938

Recently, much work [2]–[7] has been done on individual tree extraction and classification of tree species from light detection and ranging (LiDAR) data. In [2]–[4], TLS was used; in [5]–[7], airborne laser scanning (ALS) was used. ALS systems obtain only the top structures of the objects. In contrast, data collected by TLS systems have higher sampling density and provide richer information.

In terms of an application scenario, most of the existing methods are aimed at urban areas. Puttonen *et al.* [8] proposed a method that uses fused mobile laser scanning and hyperspectral data to classify trees. Yao and Wei [9] combined airborne LiDAR data and imagery as features to detect trees in 3-D. In the case of plantation forests, trees are close to each other, causing mutual occlusion, and there is much noise (e.g., lush weeds and stones), which increases the difficulty in segmentation and classification. In terms of the classification algorithm, most of the existing methods do not take into account the rotation invariance of the point clouds. Li *et al.* [10] described the tree textures by deriving several LiDAR features in forests. In [11], a feature extraction method was proposed to characterize simple indoor objects. A method using waveform features and a deep learning model was presented in [12].

To classify trees, this letter presents a novel method, based on the rasterization and application of a deep learning model to 3-D point clouds. The method consists of individual tree extraction, feature extraction, and classification. In the first stage, individual trees are extracted based on center density, followed by data preprocessing including ground point removal, noise removal from occlusion [13], and main direction adjustment of the tree trunk [14]. In the second stage, the 3-D point cloud is projected onto 2-D images that contain the outlines of the trees. Considering the rotation invariance of the raw data, the projection is repeated every 10° to increase the richness of the training samples. The projected images represent the low-level features of the trees. A deep belief network (DBN) is then introduced to generate high-level features, which are used by a softmax classifier in the final classification step.

The framework for tree classification in 3-D scenes proposed in this letter demonstrates excellent performance in experiments. The rasterization, multiview projection, and deep learning techniques used in this letter can be extended to the general classification of 3-D objects.

The rest of this letter is arranged as follows. Section II gives the details of the proposed method. Section III shows and

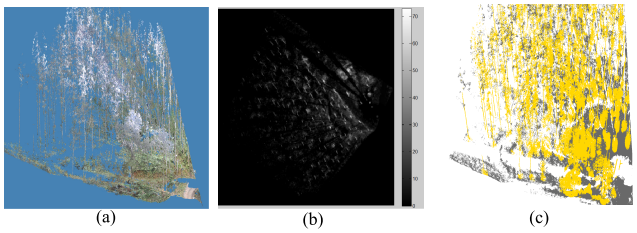



Fig. 1. Processing procedure of individual tree extraction. (a) Raw point cloud. (b) Density map of the point cloud. (c) Results of tree extraction.

discusses the experimental results. The conclusion is presented in Section IV.

## II. Methods

First, we introduce the method for individual tree extraction in Section II-A, and then describe the preprocessing for individual tree point clouds in Section II-B. Voxel-based rasterization and deep learning-based classification are presented in Sections II-C and II-D, respectively.

### A. Individual Tree Extraction

The observation that a tree trunk usually has the largest point density in a horizontal projection image can be used to determine the position of the trunk.

First, the forest point cloud is divided into grid blocks, and according to its $x$ and $y$ coordinates, each 3-D point in the point cloud is assigned to a grid block. Then, all the points are projected onto the horizontal plane, and the point density is calculated by counting the number of points in each grid block. Blocks with high densities are preserved by setting a threshold, and the central points of those blocks are considered the candidate tree centers. The neighborhood with radius $r$ on the horizontal plane for each candidate tree center is considered the scope of the tree. The value of $r$ is determined by the average distance between one tree and other trees. Finally, the individual tree is extracted by backprojecting the points in the neighborhood of the tree center onto the original point cloud. Fig. 1, which shows the raw point cloud, density map of the point cloud, and tree extraction results, illustrates the procedure for individual tree extraction.

### B. Preprocessing for Individual Tree Point Cloud

Because of the complexity of the forest environment, an individual tree point cloud must be preprocessed, which mainly includes two steps.

*Step 1—Noise Removal:* Noise includes the ground points and points of branches and leaves of other trees. Voxel-based upward-growing filtering [15] is used to remove ground points. This method partitions point cloud data into an octree structure with a voxel size. For each voxel, the octree structure expands to its nine neighboring upward voxels; then, the growing scheme expands until it reaches the top boundary. If the elevation of the top voxel is lower than the predefined threshold, the cluster of these voxels is referred to as the ground, and the points in these voxels are removed from the tree point cloud. Distance-based clustering is used to remove the noise generated by the branches and leaves of other trees.

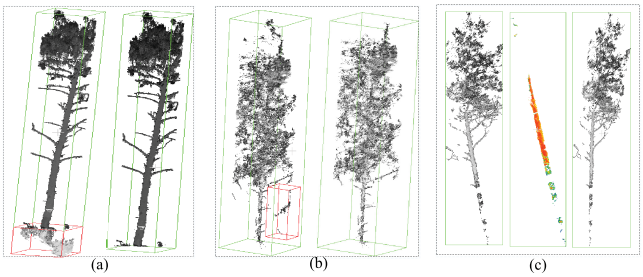



Fig. 2. Preprocessing for individual tree point cloud. (a) Ground point removal. (b) Noise removal. (c) Trunk direction adjustment.

First, a $K$-d tree is constructed for an individual tree point cloud; then, the vertex point of the $K$-d tree is regarded as the starting point. If the distance between the starting point and the next point is less than the threshold, the two points are considered to be of the same class and marked. The starting point is changed when the previous point is traversed. The class with the greatest number of points is deemed to be the tree; the other classes are removed as noise. Fig. 2(a) and (b) shows the results of ground point removal and noise removal, respectively.

*Step 2—Main Direction Adjustment of the Tree Trunk:* As the slope of the forest ground changes considerably, many tree trunks, after extraction, are not perpendicular to the coordinate system. The variety of tree directions may negatively affect the classification results; thus, it is necessary to adjust the main direction of the trunk so that it is perpendicular to the $xy$ plane. The longest straight line in the tree is fit by searching the tree points; according to the angle between the fit line and the $xy$ plane, the original tree point cloud is rotated so that it is vertical. The process is shown in Fig. 2(c).

### C. Voxel-Based Rasterization

After individual tree extracting and preprocessing, to generate low-level features, the 3-D point cloud of an individual tree is projected onto 2-D images. The 3-D trees are projected at a specific angle of rotation, similar to taking photographs from different positions. Because the direction of the trunk has been adjusted, to obtain a projection image, we fix the $z$-axis of the tree coordinates and rotate the $xy$ plane around the $z$-axis at a specific angle $\alpha$. The coordinates are estimated as follows:

$$\begin{aligned} x' &= x * \cos\alpha - y * \sin\alpha \\ y' &= x * \sin\alpha + y * \cos\alpha \\ z' &= z \end{aligned} \tag{1}$$

where $x'$, $y'$, $z'$ are the transformed coordinates, $x$, $y$, $z$ are the original coordinates, and $\alpha$ is the angle of rotation.

The rasterization method divides the grid blocks in the sample space, which is followed by counting of the number of points in each grid, then performing grid accumulation in the $xz$ plane along the $y$-axis, and finally obtaining a picture similar to a gray image. As shown in Fig. 3, the original tree is in the red dashed box, and the results of the profile projection per 30° rotation are shown in the blue dashed box.

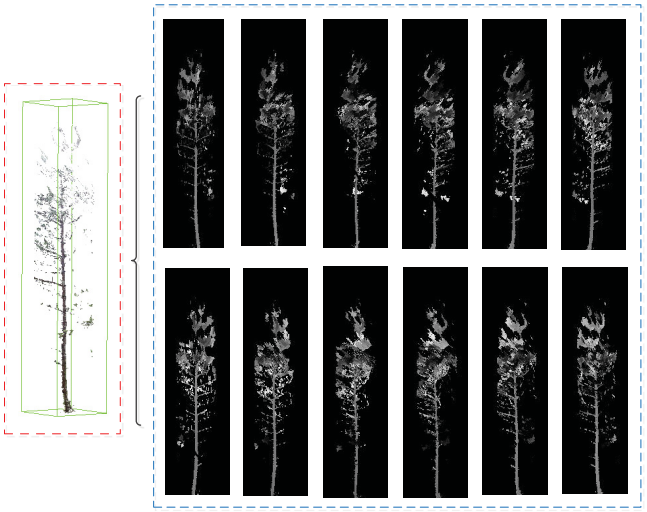


Fig. 3. Rasterization and projection of individual tree.

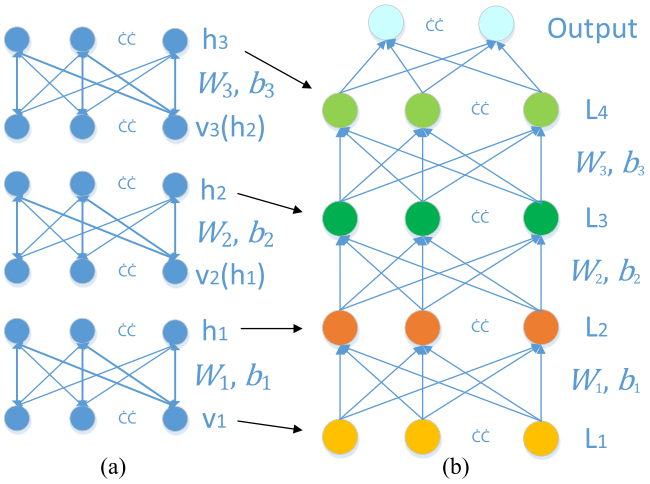



Fig. 4. Structure of DBN model. (a) RBM layers. (b) Fully connected layers.

### *D. Classification Using Deep Learning*

Recently, deep learning techniques have been shown to be superior to other algorithms in image processing, especially in the field of target classification [16], [17]. Most of the deep learning models require a huge number of training samples. However, in the case of a forest, there is an insufficient number of samples. In this letter, we choose the DBN model [18], which achieves better convergence with a small-scale sample. Compared with other deep networks, DBN has a special step in model training: restricted Boltzmann machine (RBM) greedy layer-wise pretraining. The benefit is that the parameter initialization of the network is not random, but, in advance, adjusts the network parameters in a state that converges relatively easily.

The DBN model includes RBM and fully connected layers. As shown in Fig. 4, a DBN model consists of a three-layer RBM [see Fig. 4(a)] and three fully connected layers [see Fig. 4(b)]. Denote $v \in [0,1]^n$ as the visible units representing the rasterization feature of a tree, where $n$ is the number of visible units. Denote $h \in \{0,1\}^m$ as the hidden units, where $m$ is the number of hidden units in the hidden layers. The energy function of an RBM layer is defined as follows:

$$E(v,h;\theta) = -\sum_{i=1}^{n}\sum_{j=1}^{m} W_{ij} v_i h_j - \sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j \quad (2)$$

where $\theta = \{W_{ij}, a_i, b_j\}$ are the model parameters; $v_i$ is the $i$th element of $v$; $h_j$ is the $j$th element of $h$; $W_{ij}$ is the weight coefficient between the visible unit $i$ and the hidden unit $j$; $a_i$ and $b_j$ are the weight coefficients between the network and offset units, respectively. The activation function over the visible and hidden units is expressed as follows:

$$P(h_j = 1 \mid v) = g\left(\sum_{i=1}^{n} W_{ij} v_i + b_j\right) \quad (3)$$

$$P(v_i = 1 \mid h) = g\left(\sum_{j=1}^{m} W_{ij} h_i + c_i\right) \quad (4)$$

where $g(x) = 1/(1+e^{-x})$ is the logistic function.

The solution process for training model is divided into two stages: solution of the RBM and solution of the fully connected layers.

1) *Stage One:* The training tree samples are projected onto multiview images as input, and a greedy layer-wise pretraining [19] method is used to initialize the RBM parameters $W_i$ and $b_i$, $i = 1, 2, 3$, with input images [see Fig. 4(a)]. Then, a combination of Gibbs sampling and the contrast divergence method are used to adjust the values of the units.
2) *Stage Two:* This step uses the RBM parameters to initialize the fully connected layers [see Fig. 4(b)]. Also, this stage permits the network parameters to forward communication, and then uses a stochastic gradient descent to adjust the parameters.

To classify the tree species, a softmax layer is added at the top of network, which is defined as follows:

$$P(y = c) = \frac{\exp\left(\sum_d w_{id} x_d\right)}{\sum_j \exp\left(\sum_d w_{jd} x_d\right)} \quad (5)$$

where $c$ is a predicted label, $w_{id}$ and $w_{jd}$ are the weights, and $x_d$ are the units of the last layer.

In the training stage, pretreated trees are rasterized into multiview images, and a DBN model is trained (see the red dashed box in Fig. 5). The number of projection planes is decided by the angle of rotation at every turn, and the angle setting is discussed in Section III. In the test stage (see the green dashed box in Fig. 5), after rasterizing each pretreated tree into images, each projection image is taken as input to obtain the results of the deep model, and finally voted according to the results of all the projection images from the same tree. Each image that participates in the vote uses the same weight and confirms the tree species by obtaining the class that won most of the votes, defined as follows:

$$\text{Class(tree)} = \arg\max_{c_j}(\text{sum}_i(y_i = c_j)) \quad (6)$$

where $c_j$ stands for tree species with the $j$th class, and $y_i$ stands for the $i$th projection image.

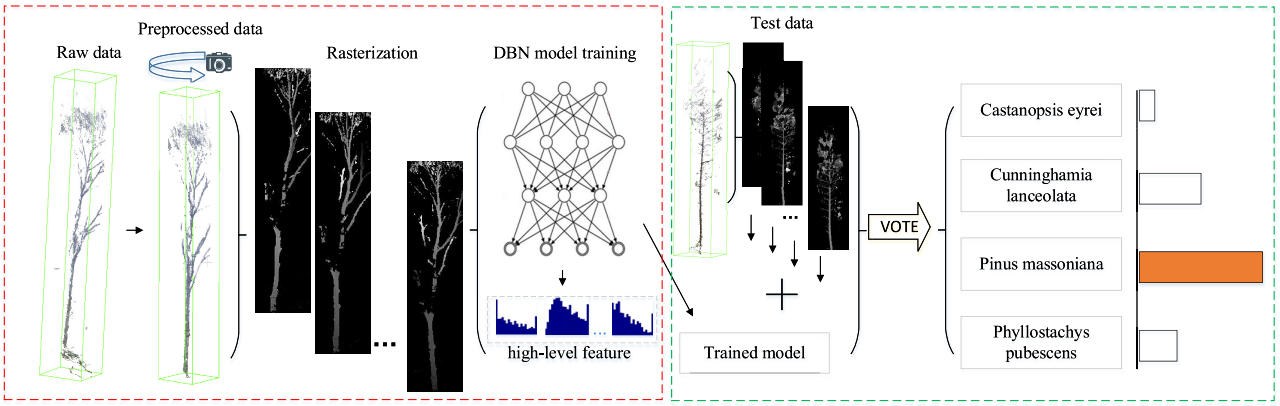



Fig. 5. Workflow of the proposed method. The training process is in the red dashed box, and the test process is in the green dashed box.


Fig. 6. Four tree species in data set 1.

TABLE I
OVERALL RESULTS OF TREE SPECIES CLASSIFICATION ACCURACY BY USING CONFUSION MATRIX

| | | Predicted class | | | | | Accuracy |
|---|---|---|---|---|---|---|---|
| | Species | C1 | C2 | C3 | C4 | Total | UA (%) |
| Actual class | C1 | 56 | 1 | 0 | 1 | 58 | 96.6 |
| | C2 | 0 | 67 | 5 | 2 | 74 | 90.5 |
| | C3 | 0 | 4 | 56 | 2 | 62 | 90.3 |
| | C4 | 0 | 1 | 1 | 52 | 54 | 90.3 |
| | Total | 56 | 73 | 62 | 57 | 248 | |
| | PA (%) | 100 | 91.8 | 90.3 | 91.2 | | |
| Accuracy | OA (%) = 93.1<br>Kappa coefficient = 0.9 | | | | | | |

Results from dataset one (shown in Fig. 6).
OA is the overall accuracy; UA is the user's accuracy;
PA is the producer's accuracy.

## III. RESULTS

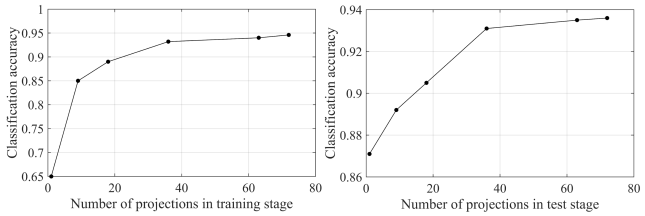
We used two data sets to evaluate our proposed method. Data set 1 was collected by the RIEGL VZ-1000 system (120 lines and 300 000 points per second) in Jiangle, China. The data set includes the following four tree species (see Fig. 6): C1: *Castanopsis eyrei*, C2: *Cunninghamia lanceolata*, C3: *Pinus massoniana*, and C4: *Phyllostachys pubescens*. Data set 2, which was collected by the RIEGL VMX-450 system (400 lines and 1.1 million points per second) in Xiamen, China, has also been used in [12]. The data set consists of the following eight tree species (see Fig. 8): T1: *Elaeocarpus apiculatus* mast, T2: sago *Cycas*, T3: *Trachycarpus fortunei*, T4: *Roystonea regia*, T5: *Bischofia polycarpa*, T6: *Delonix regia*, T7: *Euonymus japonicus*, and T8: *Mangifera indica*.

### A. Tree Classification

For data set 1, 340 tree samples (12 240 projection images if we rotate 10° each time) were used to train the DBN model. At the test stage, 248 tree samples (8928 projection images) were used to evaluate the method. All the trees were extracted from six forest point clouds. The area of each forest sample is about one hectare. In Table I, the results of the classification of the tree species are illustrated by providing a confusion matrix. The overall accuracy of the tree classification is 93.1%; the kappa coefficient is 0.9. The producer's accuracies and the user's accuracies for all test samples are greater than 90%. The similarities between C2 and C3 cause the main error in the classification.

The parameter values used in the experiments are as follows. For extracting an individual tree, we set the block size at 0.2 m. In the preprocessing step, we set the threshold at 3000 to remove the ground points and set the clustering distance at 0.4 m when eliminating the noise. As for the rasterization, we fixed the projection area at 3 m × 15 m, so that it would contain most of the information of the trees, and set the voxel size at 3 cm, generating images of 100 × 500 pixels.

Our method has rasterized trees in both the training and testing stages. Therefore, we discuss the impact that the number of projection images has on the experimental results in these two stages. In the training stage, we set the number of projection images from 1 to 72 by changing the rotation angle of the rasterization. Fig. 7(a) shows the effect that different numbers of projection images have on the accuracy. The richer the information about a sample as the number of projection images increases, the higher is the accuracy rate. Also, as shown in Fig. 7(a), the increase in accuracy slows when the number of projections reaches a certain quantity. However, more projection images would influence




(a) (b)

Fig. 7. Accuracy at different number of projection images. (a) Variation curve in the training stage. (b) Variation curve in the testing stage.

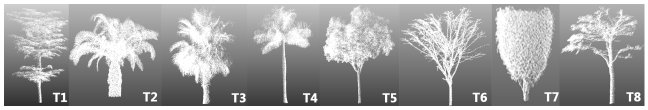



Fig. 8. Eight tree species in data set 2.

TABLE II
COMPARATIVE RESULTS OF CLASSIFICATION ACCURACY

| Method | OA (%) | Kappa coefficient | UA (%) | | PA(%) | |
|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max |
| Method in [12] | 85.3 | 0.8 | 78.2 | 91.1 | 81.3 | 90.7 |
| Proposed Method | 95.6 | 0.9 | 93.3 | 97.1 | 94.5 | 96.2 |

Results from dataset two (shown in Fig. 8).
OA is the overall accuracy; UA is the user's accuracy;
PA is the producer's accuracy.

the efficiency in model training. Therefore, we chose 36 as the number of projections in training to balance the accuracy and efficiency. Fig. 7(b) shows how accuracy is affected by the different number of projection images used to vote in the testing stage. The more the images participating in the voting, the higher is the accuracy. However, efficiency must also be considered. Finally, in testing, 36 projection images are used.

### *B. Comparison*

We used data set 2 (see Fig. 8), which was also used in [12], to perform comparative experiments. Table II exhibits the results of the comparison. We used the projection images of trees as low-level features, which contain more original information about the trees and are easier for the deep model to learn than the waveform features in [12]. Therefore, for accuracy and the kappa coefficient, our method achieves values of 95.6% and 0.9%, respectively, which are higher than the values achieved by the method proposed in [12].

## IV. CONCLUSION

In this letter, we proposed a novel rasterization-based method for the classification of tree species from TLS point clouds of complex forest scenes. Our method consists of individual tree extraction and noise removal, representing tree features by voxel-based rasterization and classifying tree species with a DBN model. Experiments show that high accuracy is achieved on both data sets. Rasterization is a powerful expression of 3-D object information. In the future, we will continue to consider more effective ways to express 3-D objects.

## REFERENCES

[1] J. Zhao, "Public finance infrastructure investment in the forestry investigation," *Forestry Econ.*, no. 12, pp. 94–99, Dec. 2010, doi: 10.13843/j.cnki.lyjj.2010.12.017.
[2] J. Vauhkonen *et al.*, "Classification of spruce and pine trees using active hyperspectral LiDAR," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 5, pp. 1138–1141, Sep. 2013.
[3] M. Rutzinger, A. K. Pratihast, S. O. Elberink, and G. Vosselman, "Detection and modelling of 3D trees from mobile laser scanning data," in *Proc. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. 38. 2010, pp. 520–525.
[4] B. Wu *et al.*, "A voxel-based method for automated identification and morphological parameters estimation of individual street trees from mobile laser scanning data," *Remote Sens.*, vol. 5, no. 2, pp. 584–611, Jan. 2013.
[5] B. Wu, B. Yu, Q. Wu, Y. Huang, Z. Chen, and J. Wu, "Individual tree crown delineation using localized contour tree method and airborne LiDAR data in coniferous forests," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 52, pp. 82–94, Oct. 2016.
[6] R. Dinuls, G. Erins, A. Lorencs, I. Mednieks, and J. Sinica-Sinavskis, "Tree species identification in mixed baltic forest using LiDAR and multispectral data," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 5, no. 2, pp. 594–603, Apr. 2012.
[7] B. St-Onge, F. A. Audet, and J. Bégin, "Characterizing the height structure and composition of a boreal forest using an individual tree crown approach applied to photogrammetric point clouds," *Forests*, vol. 6, no. 11, pp. 3899–3922, Oct. 2015.
[8] E. Puttonen, A. Jaakkola, P. Litkey, and J. Hyyppä, "Tree classification with fused mobile laser scanning and hyperspectral data," *Sensors*, vol. 11, no. 5, pp. 5158–5182, May 2011.
[9] W. Yao and Y. Wei, "Detection of 3-D individual trees in urban areas by combining airborne LiDAR data and imagery," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 6, pp. 1355–1359, Nov. 2013.
[10] J. Li, B. Hu, and T. L. Noland, "Classification of tree species based on structural features derived from high density LiDAR data," *Agricult. Forest Meteorol.*, vols. 171–172, pp. 104–114, Apr. 2013.
[11] Z. Wu *et al.*, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 1912–1920.
[12] H. Guan, Y. Yu, Z. Ji, J. Li, and Q. Zhang, "Deep learning-based tree classification using mobile LiDAR data," *Remote Sens. Lett.*, vol. 6, no. 11, pp. 864–873, Nov. 2015.
[13] H. Song and H.-Y. Feng, "A global clustering approach to point cloud simplification with a specified data reduction ratio," *Comput.-Aided Des.*, vol. 40, no. 3, pp. 281–292, Mar. 2008.
[14] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, Jun. 2007.
[15] Y. Yu, J. Li, H. Guan, and C. Wang, "Automated extraction of urban road facilities using mobile laser scanning data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2167–2181, Aug. 2015.
[16] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba, "Learning with hierarchical-deep models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1958–1971, Aug. 2013.
[17] Y. Yu, J. Li, H. Guan, F. Jia, and C. Wang, "Learning hierarchical features for automated extraction of road markings from 3-D mobile LiDAR point clouds," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 2, pp. 709–726, Feb. 2015.
[18] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.
[19] R. Salakhutdinov and G. Hinton, "An efficient learning procedure for deep Boltzmann machines," *Neural Comput.*, vol. 24, no. 8, pp. 1967–2006, Aug. 2012.