PFG (2022) 90:103–121
https://doi.org/10.1007/s41064-022-00200-4

ORIGINAL ARTICLE

Classification of Tree Species and Standing Dead Trees with Lidar Point Clouds Using Two Deep Neural Networks: PointCNN and 3DmFV-Net

Maximilian Hell[1,2,3] · **Melanie Brandmeier**[1,2] · **Sebastian Briechle**[3] · **Peter Krzystek**[3]

Received: 5 October 2021 / Accepted: 18 February 2022 / Published online: 30 March 2022

Abstract

Knowledge about tree species distribution is important for forest management and for modeling and protecting biodiversity in forests. Methods based on images are inherently limited to the forest canopy. Airborne lidar data provide information about the trees' geometric structure, as well as trees beneath the upper canopy layer. In this paper, the potential of two deep learning architectures (PointCNN, 3DmFV-Net) for classification of four different tree classes is evaluated using a lidar dataset acquired at the Bavarian Forest National Park (BFNP) in a leaf-on situation with a maximum point density of about 80 pts/m^2. Especially in the case of BFNP, dead wood plays a key role in forest biodiversity. Thus, the presented approaches are applied to the combined classification of living and dead trees. A total of 2721 single trees were delineated in advance using a normalized cut segmentation. The trees were manually labeled into four tree classes (*coniferous*, *deciduous*, standing *dead tree* with crown, and *snag*). Moreover, a multispectral orthophoto provided additional features, namely the Normalized Difference Vegetation Index. PointCNN with 3D points, laser intensity, and multispectral features resulted in a test accuracy of up to 87.0%. This highlights the potential of deep learning on point clouds in forestry. In contrast, 3DmFV-Net achieved a test accuracy of 73.2% for the same dataset using only the 3D coordinates of the laser points. The results show that the data fusion of lidar and multispectral data is invaluable for differentiation of the tree classes. Classification accuracy increases by up to 16.3% points when adding features generated from the multispectral orthophoto.

Keywords Classification · Tree species · Dead wood · Lidar · Point clouds · Deep neural networks

Zusammenfassung

Klassifizierung von Baumarten und abgestorbenen Bäumen anhand von Lidar-Punktwolken mit zwei tiefen neuronalen Netzen: PointCNN und 3DmFV-Net. Das Wissen über Baummerkmale wie Standort und Baumarten ist wichtig für die Waldbewirtschaftung und den Schutz der biologischen Vielfalt in Wäldern. Die auf Bildern basierenden Methoden sind von Natur aus auf das Kronendach des Waldes beschränkt. Luftgestützte Lidar-Daten liefern Informationen über die geometrische Struktur der Bäume, sowie über die Bäume, welche unter der oberen Baumkronenschicht stehen. In dieser Arbeit wird das Potenzial von zwei Deep-Learning-Architekturen (PointCNN, 3DmFV-Net) für die Klassifizierung von vier verschiedenen Baumklassen anhand eines Lidar-Datensatzes evaluiert, der im Nationalpark Bayerischer Wald (BFNP) im belaubten Zustand mit einer maximalen Punktdichte von ca. 80 Punkten/m^2 aufgenommen wurde. Gerade im BFNP spielt das Totholz eine wichtige Rolle für die Biodiversität im Wald. Daher werden die vorgestellten Ansätze auf die kombinierte Klassifizierung von lebenden und toten Bäumen angewandt. Insgesamt wurden 2.721 Einzelbäume vorab mit einer Normalized Cut Segmentierung deliniert. Anschließend wurden die Bäume manuell in vier Baumklassen eingeteilt (Nadelbäume,

✉ Maximilian Hell
maximilian.hell@fhws.de

Melanie Brandmeier
melanie.brandmeier@fhws.de

Sebastian Briechle
sebastian.briechle@hm.edu

Peter Krzystek
peter.krzystek@hm.edu

[1] Esri Deutschland GmbH, Kranzberg, Germany

[2] Faculty of Plastics Engineering and Surveying, FHWS University of Applied Sciences Würzburg-Schweinfurt, Würzburg, Germany

[3] Faculty of Geoinformatics, HM University of Applied Sciences München, Munich, Germany

Laubbäume, stehende tote Bäume mit Krone und Nadelbäume). Darüber hinaus lieferte ein multispektrales Orthofoto zusätzliche Merkmale wie den Normalized Difference Vegetation Index (NDVI). Bei der Verwendung von PointCNN mit 3D-Punkten, Laserintensität und multispektralen Merkmalen wurde eine Testgenauigkeit von bis zu 87,0% erreicht, was das Potenzial dieses Deep-Learning-Ansatzes für die Forstwirtschaft zeigt. Im Gegensatz dazu erreichte das 3DmFV-Net eine Testgenauigkeit von 73,2% für denselben Datensatz, indem es nur die 3D-Koordinaten der Laserpunkte verwendete. Die Ergebnisse zeigen, dass die Datenfusion von Lidar- und Multispektraldaten entscheidend für die Differenzierung der Baumklassen ist. Die Klassifizierungsgenauigkeit steigt um bis zu 16,3 Prozentpunkte, wenn aus dem multispektralen Orthofoto generierte Merkmale hinzugefügt werden.

1 Introduction

The world's forests cover 31% of the total land area (FAO 2020), which makes them one of the most important ecological systems on earth. Trees are fundamental for life on earth because of their ability to absorb carbon-dioxide and produce oxygen in return. They affect regional climates and the hydrological balance. Additionally, they provide a habitat to a plethora of living creatures and serve humans as a renewable resource for construction material and energy. Thus, they are fundamental to biodiversity and human life. Another important part of this ecosystem is dead wood, which is part of the natural life cycle. When trees die, they decompose and form a new type of habitat for insects, fungi, and birds (BMEL 2015). This process recoups nutrients for the growth of new trees. However, an increase in tree loss due to climate change and related extreme weather conditions, such as storms, droughts, and their respective consequences, have been observed. Thus, a good understanding of forest structures and tree stands is required for successful forest management, conservation, and planning. Knowledge of tree species, health, and location within a forest area is of particular interest. Traditionally, this information is obtained in situ using measurements from sample plots and extrapolated to larger scales (McRoberts and Tomppo 2007).

The capture of information relevant to forestry was accelerated through the emergence of remotely sensed data from space or airborne platforms, such as satellites, airplanes, helicopters, or remotely piloted aircrafts. These can be equipped with multi- or hyperspectral sensors to capture high-resolution images. In recent years, data acquisition has extended to using lidar that penetrates the tree canopy and provides information from the lower layers of the forest. Additionally, the laser points provide geometric and radiometric information (Korpela et al. 2010). In a preprocessing step, the single trees are then segmented using the captured point clouds to assess further information at the single/object level.

Conventionally, features are developed and engineered to describe different traits, such as tree health and species. Usually, a statistic classification or machine learning (ML) model is trained to then infer labels on unlabeled data. Fassnacht et al. (2016) showed that well-known parametric and non-parametric approaches—such as maximum likelihood, support vector machines, or random forest—are often used to classify trees, whilst neural networks are not commonly employed. However, recent developments in the field of computer vision and deep learning have attracted more attention to the development of models that learn from three-dimensional data (Guo et al. 2021). These approaches can alleviate the time needed to develop hand-crafted features from the point clouds, because they extract information and build representations automatically. However, they need much more training data compared to traditional ML methods.

1.1 Deep Learning on Point Clouds

Deep learning on images has become very successful with the rise of convolutional neural networks (CNNs). These networks use the regular grid structure of images and can derive information through the neighborhood relationships of pixels using convolution operators. However, the application of this approach on point clouds reveals itself as a challenge, because point clouds are inherently irregular and unordered. In recent years, an increasing number of publications have dealt with deep learning on point clouds. These publications provide various approaches to address problems such as 3D point cloud segmentation, point cloud registration, 3D reconstruction, or object classification. In this paper, we focus on classification algorithms for tree classes at the tree level extracted from lidar data. Griffiths and Boehm (2019) provide a very good overview of the techniques for applying deep learning on three-dimensional sensed data. Guo et al. (2021) also present an extensive overview of deep learning models applied specifically to point clouds.

In the field of object classification, there are different approaches for handling and processing data. Initial attempts, such as the multi-view convolutional neural network (MVCNN) (Su et al. 2015), projected the 3D data on two-dimensional planes prior to processing. Projecting the objects onto several planes in space generates multiple images. These projections of the point cloud are then handled like usual images and can be processed by the well-known 2D convolutions. Other approaches were developed to discretize the point cloud directly within three-dimensional space. A 3D grid structure is generated

by voxelizing the space and summarizing multiple points into a spatial unit. The 3D convolution can be applied to the voxels to further extract global descriptors. A notable example of this kind is VoxNet (Maturana and Scherer 2015). However, this subdivision of space does not scale well for denser point clouds.

To avoid the possible loss of information due to the discretization of the point cloud, newer networks have been developed that act directly on the points. A notable example is PointNet (Qi et al. 2017). This network learns pointwise features through several multilayer perceptrons (MLPs). Because this network does not capture local information and the features are learned independently, Qi et al. (2017) proposed its successor PointNet++. This network applies PointNet learning layers in a hierarchical fashion to smaller groups of points to learn their neighborhood structures. PointNet and PointNet++ are widely considered as milestones in deep learning on point clouds (e.g., in Jiao and Yin 2020; Wang et al. 2020). Both networks can be categorized as point-based MLP methods.

Another category of deep learning networks that act directly on points are convolution-based approaches. These networks often consider the neighborhood of points and then use convolution to act on them. For the convolution, either continuous or discrete representations of the points are used. This can be realized through an operator to bring them into a latent or canonical space.

A relatively new network architecture is PointCNN, proposed by Li et al. (2018), which is evaluated in the present study. We compare its performance to 3D-modified Fisher Vectors Net (3DmFV-Net), proposed by Ben-Shabat et al. (2018). 3DmFV-Net network uses a discrete three-dimensional grid together with a continuous function to transform the point cloud into another representation and subsequently apply 3D convolutions. Therefore, this method has been coined as a hybrid approach.

Both networks are further explained in Sect. 3. Some point cloud deep learning networks have already been successfully applied to forest related tree classifications or segmentation. Such studies are presented in the following section.

1.2 Related Work

Research on automated tree species classification from lidar is mostly conducted in the field of ML or other statistical methods. However, conventional ML often requires engineered features to be derived from the raw data, namely the point clouds, to then learn the classifier. Using deep learning, the need for specially developed features is reduced or even eliminated, because the networks automatically extract features from the learned representations of the raw input data (LeCun et al. 2015). However, studies using deep learning models on segmented individual trees from lidar data to classify their species have not yet been widely explored. Most significantly, this applies to cases in which approaches act on the lidar points themselves instead of discretizing them into two- or three-dimensional representations, namely pixels or voxels. Studies employing deep neural networks (DNNs) to classify individual trees on an object level in combination with aerial lidar data are presented in the following.

Hamraz et al. (2019) used two airborne lidar acquisitions captured in Robinson Forest in leaf-on and leaf-off conditions to classify single trees into coniferous and deciduous species. For both seasons, they generated digital surface models from the upper canopy layer and an image stack compiled by nadir and side views of the single trees. In total, 124 coniferous and 2404 deciduous trees were available for training. These were then fed into a two-dimensional CNN model. Overstory trees reached classification accuracies of 92.1% for coniferous and 87.2% for deciduous trees. In contrast, understory trees were not as strongly identified, with accuracies of 69.0% and 92.1% for coniferous and deciduous trees, respectively.

By combining lidar data with high-resolution multispectral satellite imagery, Hartling et al. (2019) classified eight different tree species—of which seven were deciduous and one coniferous—in an urban environment. They generated intensity images from the lidar data and used these in combination with the panchromatic and infrared bands from WorldView 2 and 3 images. Training a DenseNet (Huang et al. 2017) classifier on the image patches of single trees—a total of 1552 tree samples for training—yielded an overall accuracy (OA) of 82.9% and an average accuracy of 80.9% for all eight classes.

Mustafić and Schardt (2019) used airborne laser scanning (ALS) data from a forest near Burgau, Austria, with a relatively low point density (15 pts/m^2). They examined three tree classes: pine, spruce, and deciduous trees (consisting of beech, birch, oak, ash, and alder). Each of these classes contains exactly 500 trees. The local maxima from a Canopy Height Model (CHM) were used as positions for tree candidates. Using different radii around those points, the single trees were approximated from the point cloud, and the neural network found the most fitting size. They further reduced the single trees into two dimensions by projecting the trees onto a plane and encoding the orientation of the points as color. Additionally, a set of one-dimensional representations (histograms and percentiles) was used as input for the DNN. Their proposed method of using multiple CNNs reached a mean OA of 74%, which was higher than their comparative experiments using transfer learning with InceptionV3 (Szegedy et al. 2016) and VGG16 (Simonyan and Zisserman 2015).

The study conducted by Sun et al. (2019a) in a wetland ecosystem in south China aimed at classifying 18 different

tree species. The authors delineated single trees using the CHM generated from the lidar point cloud. Image patches for each tree were extracted and used for training three CNNs. To increase the number of training samples, the image patches were randomly rotated, mirrored, and flipped. This data augmentation resulted in 5664 samples for the training process. VGG16, AlexNet (Krizhevsky et al. 2012), and ResNet50 (He et al. 2016) were adjusted to take image patches of size 64×64 as input and then trained on their dataset. The best result was obtained using VGG16, with an OA of 73.55% and a F_1-score of 100% in two classes. On the same study site, Sun et al. (2019b) classified six tree species using a similar approach. Single trees were identified in the lidar CHM and used to crop multispectral images into patches of 64×64 pixels. By training the ResNet50 model on these patches, an OA of 89.8% was achieved.

Li et al. (2020) generated aerial view intensity images from ALS data of urban settings. These were combined with high-resolution multispectral WorldView 2 images, resulting in an 11-band composite. Segmented single trees (maple, locust, pine, and spruce) were classified. From this, 1052 individually identified trees were used for training, and through data augmentation (i.e. rotation and mirroring), a total of 6312 images were generated. Three DNNs—DenseNet40, ResNet18, and a 7-layer CNN—were compared in their classification performance, resulting in a maximum OA of 88.9% using the ResNet.

Individual trees comprising three species with additional standing dead trees were successfully classified by Briechle et al. (2020). In this study, the point cloud was not discretized into a lower dimensional space, namely two-dimensional images. Instead, the point-based PointNet++ (Qi et al. 2017) was applied to the point clouds of the single-segmented trees. In the learning process point clouds of 464 trees were used to train the classifier. An OA of 90.2% was reached, with the additional integration of features derived from the echo width of the laser sensor and features calculated from a multispectral orthophoto.

Mäyrä et al. (2021) used a three-dimensional CNN on hyperspectral images (a total of 250 bands) with trees delineated from a lidar CHM. Here, 2291 trees were used to train their proposed network from scratch. Their proposed network managed to differentiate four tree species (birch, pine, aspen, and spruce) with an OA of 87.0%. Like in most studies, the lidar point cloud was solely used for tree segmentation, whilst the classification was conducted with optical imagery.

In a recent study, Seidel et al. (2021) projected the point clouds of individual trees onto multiple planes to form a two-dimensional representation. They generated 8,100 images from point clouds of 690 different trees, of which 6720 were used in the training process. They trained a multilayered CNN on these images to learn to differentiate between seven different tree species. The OA in this experiment reached 86.0%. Interestingly, PointNet (Qi et al. 2017) was used on the raw point clouds, but could not reach competitive results, because most trees were falsely predicted.

Recently, Liu et al. (2021) published a custom DNN LayerNet designed for tree species classification. They used ALS data from a forested area in Saihanba National Forest Park (China) to classify birch and larch trees. With a combined segmentation approach, single trees were delineated from the point cloud. Their proposed LayerNet was then trained and evaluated on a set of 1200 trees. Unlike the other presented studies, no multispectral data were used in addition to the point clouds. The authors reported an OA of 88.8%, which was slightly better than the OA of 86.7% achieved using PointNet.

Briechle et al. (2021) developed a DNN model using a multi-view-based approach for classifying trees, called Silvi-Net. As a baseline method, the point-based network PointNet++ Qi et al. (2017) was used. Both networks were applied on the point clouds of pre-segmented trees from two study sites. In the Chernobyl Exclusion Zone study area, four tree classes were differentiated (the same as in Briechle et al. 2020). With PointNet++, an OA of 84.8% was achieved. In contrast, Silvi-Net achieved an OA of 96.1% using lidar and multispectral data. For the second study area, Bavarian Forest National Park (BFNP), OAs of 89.3% and 91.5% were achieved with PointNet++ and Silvi-Net, respectively. Notably, this study also used the same dataset as described in Sect. 2.

Deep learning approaches for tree species classification from lidar point clouds show promising results. However, most studies use CNN approaches based on images produced from the point clouds. A few studies exist which apply Deep Neural Networks (DNNs) that directly exploit the three dimensional geometry of the point clouds. Moreover, a challenge in this field is the lack of access to large labeled datasets for training.

1.3 Objectives

Until recently, lidar was mostly used for individual tree segmentation and not as input data for deep learning models. Only a few studies have employed raw point cloud data and their underlying geometry in the applied deep learning networks. However, lidar data contain more information about forest structure compared to optical imagery. The laser beam penetrates the upper canopy and captures information about the understory vegetation and smaller trees. In this paper, we therefore demonstrate the accuracy performance of two DNNs to classify tree classes (i.e., species and standing dead trees) by learning on three-dimensional point clouds. Both networks use different approaches to handle the underlying geometric information provided by pre-segmented

trees. Moreover, the effects on the classification accuracy are tested if further features are used, such as the intensity of the laser beam and multispectral information. Finally, the results are compared with a study that used the same data but relied on an image based approach.

2 Materials

2.1 Study Area

The study area for the present research is in the BFNP near the German–Czech border (see the two red transects in Fig. 1). Since its establishment in 1970, the national park has been governed by the guiding principle of 'letting nature be nature' (Nationalparkverwaltung Bayerischer Wald 2021), which applies to its whole area of 24,250 hectares. The tree species that mainly populate the forest are Norway spruce (*Picea albies*), European beech (*Fagus sylvatica*), silver fir (*Abies alba*), and larch (*Larix*). After severe storms and windthrow in the years 1983 and 1984, the forest struggled with a bark beetle infestation (Bibelriether 1989, as cited in Müller et al. 2008). This mass propagation—called a gradation—was not contained, as the policy was to avoid human intervention under almost any circumstances. Long periods of drought and mild climate favored a second gradation in the early 1990s, which still continues to date (Heurich et al. 2001, as cited in Müller et al. 2008). Consequently, more than 5000 ha of the spruce stand had already died by 2007

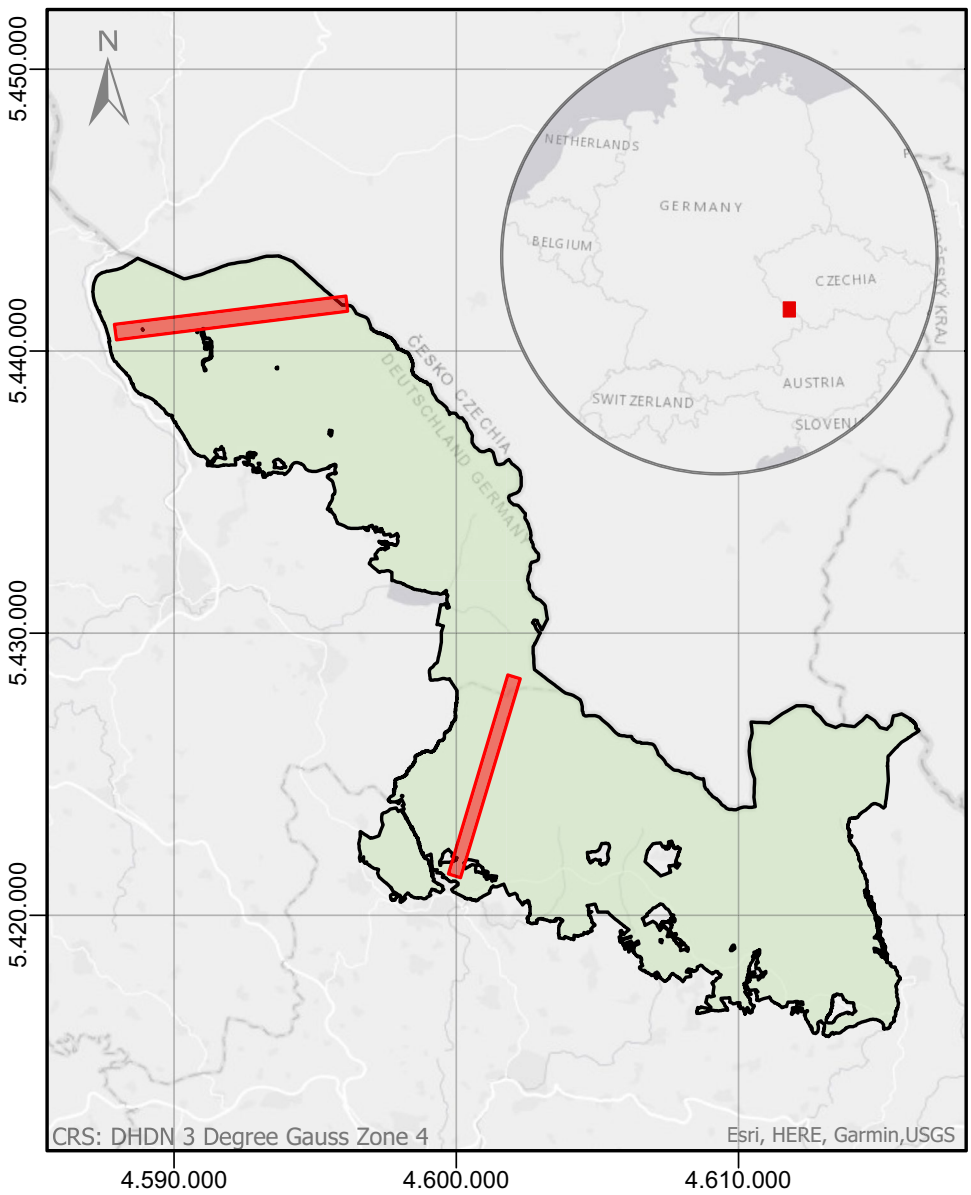


Fig. 1 Overview of the study area located in Bavarian Forest National Park. On 18.08.2016, two transects were flown with a lidar scanner attached to a helicopter, as presented in Sect. 2.2 The two transects of the lidar acquisition are marked in red

(Lausch et al. 2011). These natural stresses resulted in large amounts of standing dead wood.

The tree species classified in this study comprise *deciduous* and *coniferous* trees, as well as standing dead trees with crown, hereafter referred to as *dead tree*. The fourth class is called *snag*, which also refers to a standing dead tree but without a crown and missing most of its branches (Yao et al. 2012).

2.2 Data Acquisition

On 18th August 2016 , during the leaf-on season, full-waveform lidar data were acquired in the study area using a Riegl LMS-Q680i scanner carried by a helicopter. Two transects were flown with a 50% overlap (see Fig. 1). A flight speed of 25 m/s at an altitude of 400 m above ground resulted in a maximum point density of 80 points/m^2. Note that laser reflectance values were normalized w.r.t. travelling distance (Amiri et al. 2019). The summarized key data of the flight are shown in Table 1.

During the same period in June 2016, multispectral aerial photographs were captured using a Leica DMC III camera. True orthophotos were generated using a digital surface model calculated from the lidar data. Further details are shown in Table 2.

2.3 Data Pre-processing

2.3.1 Segmentation of Lidar Point Cloud

Individual trees were delineated using Normalized Cut segmentation (Shi and Malik 2000) and the software package Treefinder (see Krzystek et al. 2020). The quality of the segmentation was assessed through a visual inspection. An example of the segmented single trees is shown in Fig. 2a. Two-dimensional enclosing polygons were calculated from the point clouds of the delineated trees. The class assignment was done manually by visual interpretation of the individual tree point clouds and the corresponding representation in the aerial photograph. Incorrect segments were not taken into account in this process. The classes *dead tree* and *snag* were differentiated by subjective perception (Briechle et al. 2021). Shape differences between the four tree classes are exemplarily shown in Fig. 2b.

Table 1 Sensor equipment for the lidar data capture (after Amiri et al. 2019)

Platform	Helicopter D-HALL/AS350
Lidar sensor	Riegl LMS-Q680i
Flight date	18th August 2016
Laser wavelength (nm)	1550
Flight altitude above ground (m)	400
Flight speed (m/s)	25
Max. point density (points/m^2)	80
Radiometric information	Reflectance

Table 2 Sensor equipment for the multispectral data capture (after Briechle et al. 2021)

Camera	Leica DMC III
Focal length (mm)	92
Flight date	23rd June 2016
Bands	B, G, R, NIR
Flight altitude (m (NHN))	2918
Flight speed (m/s)	N/A
End/side lap (%)	75/60
GSD of orthomosaics (cm)	20

To achieve a constant number of points per object, each tree was up-/down-sampled to 1024 points. Samples with fewer points were upsampled by insertion of random copies of points. Trees with more points were reduced to this number by randomly choosing 1024 points out of the point set. Next, the 3D points of each tree were normalized to have a mean of zero, i.e., centered around the origin, and uniformly scaled to fit into a unit sphere. This led to coordinates in the range of $[-1, 1]$ for all three dimensions.

2.3.2 Generation of Multispectral Features

The Normalized Difference Vegetation Index (NDVI) (Rouse et al. 1974) (cf. Eq. 1) was calculated from the color infrared images

$$\mathrm{NDVI} = \frac{\mathrm{NIR} - \mathrm{R}}{\mathrm{NIR} + \mathrm{R}}. \tag{1}$$

All pixels of each tree were extracted using the enclosing polygons generated from the segmented single trees. Because the trees differ in size, each enclosing polygon can overlap with a different amount of pixels. It is favorable to have a constant number of feature inputs for deep learning. Thus, the pixel values are aggregated in 12 object-based descriptive statistics. In detail, the minimum (min) and maximum (max) values, the range (max − min), mean, standard deviation, mode, skewness, and kurtosis comprised eight features. Additionally, the 25th, 50th, 75th, and 90th percentiles (perc25, perc50, …) were computed and normalized. The five most important features for a classification were then identified using a random forest-based feature extraction, which was performed by Briechle et al. (2021). The five final features were min, mean, perc25, skewness, and range. These final features were also used to make a one-to-one comparison with the results of Briechle et al. (2021). In the following, these features will be referred as to *MS*.

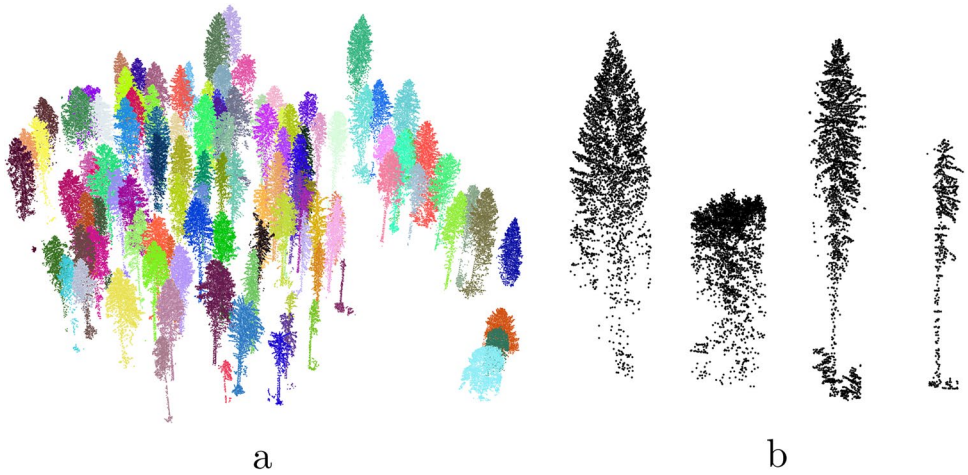


Fig. 2 **a** Single trees segmented by normalized cut. Different colors represent identified individual trees. **b** Point cloud examples of the four tree classes from left to right: *coniferous*, *deciduous*, *dead tree*, and *snag*

Finally, surface normals for all points were estimated using the `estimate_normals` function from the open source library Open3D provided by Zhou et al. (2018) (see also Briechle et al. 2021). The default settings were used, meaning that the 30 nearest neighbors are used to perform the normals estimation.

2.3.3 Splitting the Dataset

The dataset containing the segmented trees was divided into three subsets (*train*, *validation*, *test*), comprising 51%, 22%, and 27% of the data, respectively. Note that the number of samples for each class was balanced in the *train* dataset to avoid bias effects (see Table 3 for details).

Table 3 Number of samples in the dataset

Class label	Training	Validation	Test	Total
Coniferous	345	149	259	753
Deciduous	345	149	202	696
Dead tree	345	149	139	633
Snag	345	149	145	639
Total	1380	596	745	2721
%	50.7	21.9	27.4	100

2.3.4 Standardization of Laser Intensities

The laser intensities with a value ranged from 0 to 3694 were transformed using a power transformation. LeCun et al. (1998) showed that normalizing the input data is advantageous for a faster convergence. They suggest that all input data should have a mean close to zero and almost the same covariance. We used the two-parameter Box–Cox transformation, as proposed by Box and Cox (1964), which is defined by a pair of transformation parameters $\lambda = (\lambda_1, \lambda_2)$. The function (Eq. 2) holds for $y > -\lambda_2$. Because the values of the intensities were greater than or equal to zero, the parameter λ_2 was set to 1 and λ_1 set to 0

$$\mathrm{fbc}(y,\lambda) = y^{(\lambda)} = \begin{cases} \frac{(y+\lambda_2)^{\lambda_1}-1}{\lambda_1} & (\lambda_1 \neq 0), \\ \log\left(y+\lambda_2\right) & (\lambda_1 = 0). \end{cases} \tag{2}$$

After a subsequent standardization using Equation 3, the data values were centered around zero and had a standard deviation of 1

$$z\text{–score}(y) = y_z = \frac{y-\bar{y}}{s_y}; \tag{3}$$

Figure 3a, b show the effect of the power transformation and the subsequent standardization.

3 Methodology

3.1 3D-Modified Fisher Vectors-Net (3DmFV-Net)

3DmFV-Net is a deep learning network proposed by Ben-Shabat et al. (2018) for object classification from three-dimensional data, namely point clouds. The main idea of the network is to represent the point cloud in another form and then convolve over it. The representation is made up of two components. First, a Gaussian Mixture Model (GMM) is applied to fill the space. The means of the single Gaussians, which have a uniform standard deviation in their dimensions, are placed on a three-dimensional regular grid. The second component is the representation of the data in the form of 3D-modified Fisher Vectors (3DmFV).

The proposed approach is denoted as a hybrid model, because it comprises a discrete grid that combines the more continuous character of the Gaussians with the 3DmFVs.

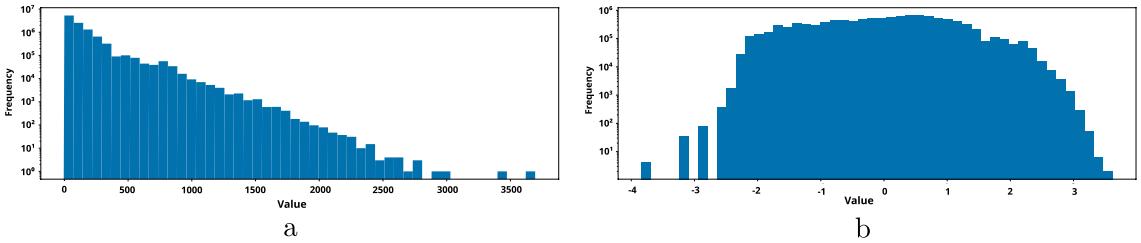


Fig. 3 Histograms of the laser intensities for all trees. The distribution of the laser intensities is transformed to approximate a normal distribution. This is achieved by applying a power transformation to the power law distributed intensities, thereby boosting classification performance. Note that the frequency scale is logarithmic. **a** Before any transformation or normalization. **b** After power transformation and subsequent standardization

This combination can be viewed as 'soft voxels', because they have fixed locations but no discrete borders.

3.1.1 Fisher Vectors

A set of points $X = \{p_t \in \mathbb{R}^D, t = 1, \dots, T\}$ is given to denote the point cloud of a segmented tree. Each of these points lie in three-dimensional space $D = 3, p_t = [x, y, z]'$. The set contains a total of T samples. A GMM is given with K components, in which each is described by its mean, or expected value, μ, its covariance matrix Σ, and mixture weight α. Let then $\lambda = \{(\alpha_k, \mu_k, \Sigma_k), k = 1, \dots, K\}$ be the set of parameters of the GMM.

The likelihood of an observed point p_t being generated by the whole GMM is a weighted sum of single Gaussians u_k

$$u_\lambda(p_t) = \sum_{k=1}^{K} w_k u_k(p_t) \tag{4}$$

with the condition that the sum of the weights equals one $\sum w_k = 1$ to be a valid probability density function. As each component is described by a weight α_k, these are normalized using the softmax function

$$w_k = \frac{\exp(\alpha_k)}{\sum_{j=1}^{K} \exp(\alpha_j)}; \quad k = 1, \dots, K. \tag{5}$$

The posterior probability of a point p_t to be associated with the kth component is given by the soft assignment

$$\gamma_t(k) = \frac{w_k u_k(p_t)}{u_\lambda(p_t)}; \tag{6}$$

Sánchez et al. (2013) described the Fisher Vector $\mathscr{G}_\lambda^X$ formally as the normalized gradient of the log-likelihood. The normalization term L_λ is given by the Cholesky decomposition of the inverse Fisher Information Matrix (FIM)

$$\mathscr{G}_\lambda^X = \sum_{t=1}^{T} L_\lambda \nabla_\lambda \log u_\lambda(p_t). \tag{7}$$

For real discrete data, the soft assignment is sharply peaked, so the FIM is diagonal and the gradient statistics can be computed from three gradients $\mathscr{G}_{\alpha_k}^X$, $\mathscr{G}_{\mu_k}^X$, and $\mathscr{G}_{\sigma_k}^X$.

The Fisher Vector is then defined as the concatenation of the summed gradients with respect to the parameters λ

$$\begin{aligned} \mathscr{G}_{FV_\lambda}^X = \Big(& \mathscr{G}_{\alpha_1}^X, \dots, \mathscr{G}_{\alpha_k}^X, \\ & \mathscr{G}_{\mu_1}^{X\prime}, \dots, \mathscr{G}_{\mu_k}^{X\prime}, \\ & \mathscr{G}_{\sigma_1}^{X\prime}, \dots, \mathscr{G}_{\sigma_k}^{X\prime} \Big). \end{aligned} \tag{8}$$

To not depend on the number of samples in each given set of points, the Fisher Vector is normalized by the number of sampled points

$$\mathscr{G}_{FV_\lambda}^X \leftarrow \frac{1}{T} \mathscr{G}_{FV_\lambda}^X. \tag{9}$$

Ben-Shabat et al. (2018) expanded this concept to the three-dimensional space and added different symmetric functions. They used the sum as a possible feature generation function but also the min and max functions with respect to the parameters α, μ, σ. They then constructed the 3DmFV as follows:

$$\mathscr{G}_{3DmFV_\lambda}^X = \begin{bmatrix} \sum_{t=1}^{T} g_\lambda^X \big|_{\lambda=\alpha,\mu,\sigma} \\ \max_t \left(g_\lambda^X\right) \big|_{\lambda=\alpha,\mu,\sigma} \\ \min_t \left(g_\lambda^X\right) \big|_{\lambda=\mu,\sigma} \end{bmatrix}. \tag{10}$$

This vector is calculated for each of the K components of the GMM. In the case of $D = 3$, both μ and σ are size 3 vectors, whilst α is a scalar. This generates a 3DmFV of size $D(3 + 3) + 2 = 20$.

Subsequently, the 3DmFV is normalized twice, in accordance with Perronnin et al. (2010). These two normalization steps are a signed power normalization $f_p(z) = \text{sign}(z)|z|^{0.5}$ followed by a ℓ^2-normalization.

3.1.2 Gaussian Mixture Model

All tree segments are fitted into a sphere centered at the origin and a radius of one (see Sect. 2). The single GMM components are placed on a uniform 3D grid in the range of $[-1, 1]$ in each dimension with a size of $m \times m \times m$, with $m \geq 3$ giving a total of $K = m^3$ components. Each is assigned a mixture weight of $\alpha_k = K^{-1}$ and a uniform standard deviation in each direction of $\sigma_k = m^{-1} \Rightarrow \Sigma_k = \sigma_k I$.

3.1.3 Inception Module

The main building block of the 3DmFV-Net is an adapted version of the Inception module. Szegedy et al. (2015) introduced the Inception module as a 'network within a network'. The used structure is shown in Fig. 4.

The sub-network consists of four 3D-convolution layers, each followed by a batch normalization (Ioffe and Szegedy 2015) and the ReLU activation function $\sigma_{\text{ReLU}}(x) = \max(0, x)$. There are three parameters for constructing the module: Two kernel sizes, c_1 and c_2, for two of the convolutions and a value N that corresponds to one-third of the output channel size. The inputs for both convolutions with the given kernel sizes c_1, c_2 are zero padded to result in the same spatial size. Using even kernel sizes, the input is only padded to one side of the dimension. The output channel size N has to be an even number. After all four convolutions, the outputs are concatenated. There is no dimensionality reduction, due to the initial padding. To reduce the dimension size, max-pooling layers are employed (see Sect. 3.1.4). The structure of the used Inception module is shown in Fig. 4.

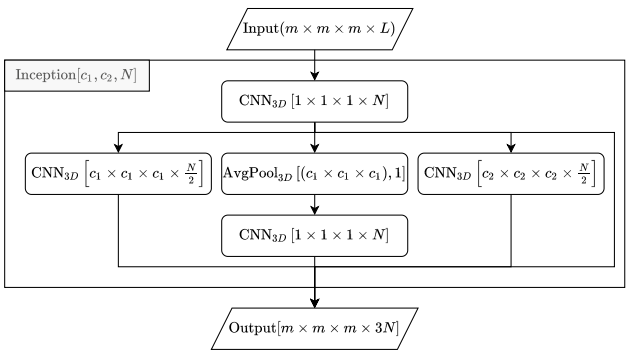


Fig. 4 Internal architecture of the Inception module as used in the 3DmFV-Net (per Ben-Shabat et al. 2018)

3.1.4 Model Architecture

The model architecture is dependent on the size of the chosen GMM. Each model comprises at least five Inception layers. After the third Inception module, a max-pooling layer is employed with $m \geq 5$. Only the model for $m = 16$ uses two more Inceptions as compared to the smaller GMMs. After the convolutional layers, the features are passed into a Fully Connected Network (FCN) to reduce the representation of the point cloud to the size the four classes. Table 4 shows the parameters of each layer for all the used GMM sizes.

Three more sizes for the GMM are then tested. They are chosen, so that they are more granular in the Z-dimension, in that more Gaussians are placed in this dimension. This is because trees, especially the two main species in the dataset (*Norway spruce, European beech*), usually grow more in height than in diameter in densely covered forests. For this approach, the sizes $(3 \times 3 \times 9)$, $(5 \times 5 \times 10)$ and $(8 \times 8 \times 12)$ are chosen. Using these values for the Z-dimension, the model structure, especially with regard to kernel sizes, does not have to be adjusted.

3.2 PointCNN

PointCNN was introduced by Li et al. (2018) and is designed, among other things, for object classification of point clouds. It is a convolutional-based approach for learning on irregular data. PointCNN transforms the geometry of point clouds into a latent feature space through its $\mathcal{X}$-Conv transformation.

A point cloud is formally described as a set of points lying within a metric space. Each point can contain none, one, or multiple features. In the case of single trees, they include the laser intensities, normals, and multispectral features as described in Sect. 2.

The input to PointCNN is a set of N points in a d-dimensional space

Table 4 Architecture of the 3DmFV-Net for different GMM sizes

Layer	m		
	3	5	8
Inception	2, 3, 64	3, 5, 64	3, 5, 64
Inception	2, 3, 128	3, 5, 128	3, 5, 128
Inception	2, 3, 256	3, 5, 256	3, 5, 256
MaxPool	–	3, 2	2, 2
Inception	2, 3, 256	2, 3, 256	3, 5, 256
Inception	2, 3, 512	2, 3, 512	3, 5, 512
MaxPool	–	–	2, 2
FC	1024		
FC	256		
FC	128		
FC	4		

$$\{p_i : p_i \in \mathbb{R}^d; i = 1, \ldots, N\}. \tag{11}$$

Each of these points has features associated with it

$$\{f_i : f \in \mathbb{R}^C; i = 1, \ldots, N\} \tag{12}$$

a position in a C-dimensional feature space or alternatively a total of C features. It is also possible that no features are associated with the points, i.e.. only the geometry of the point cloud is given. This would mean that $C = 0$ and result in the feature set being an empty set $\emptyset$. However, this has no influence on the learning procedure of the $\mathcal{X}$-Conv, as described further in Sect. 3.2.1. Together, they form the input point cloud $\mathbb{F}_1$

$$\begin{aligned}\mathbb{F}_1 = \{(p_{1,i}, f_{1,i}) : i = 1, 2, \ldots, N_1;\\ p_{1,i} \in \mathbb{R}^d; f_{1,i} \in \mathbb{R}^{C_1}\}.\end{aligned} \tag{13}$$

As this point cloud passes through the $\mathcal{X}$-Conv module, the number of channels (i.e., the dimensions of the feature space) may increase. The increased number of feature dimensions no longer represents tangible features, such as laser intensity, but a more abstract representation of small neighborhood patches and their interactions. These representations are learned by the PointCNN. The resulting point cloud is described for $C_2 \geq C_1$ as

$$\begin{aligned}\mathbb{F}_2 = \{(p_{1,i}, f_{2,i}) : i = 1, 2, \ldots, N_1;\\ p_{1,i} \in \mathbb{R}^d; f_{2,i} \in \mathbb{R}^{C_2}\}.\end{aligned} \tag{14}$$

A step usually taken when conducting deep learning on spatial data, such as images and point clouds, is the reduction of the spatial domain, thereby enriching the feature space. In the traditional convolutional layers, this happens automatically by choosing the right kernel sizes. Also, almost all convolutional neural networks employ a form of pooling (Goodfellow et al. 2016), which enables the reduction of the spatial size. In this case, the $\mathcal{X}$-Conv outputs a point cloud with the same spatial extent and, hence, needs to be downsampled (see Sect. 3.2.1).

3.2.1 $\mathcal{X}$-Conv

The input to the $\mathcal{X}$-Conv operator is an unordered set of points. Moreover, there are four parameters in three groups for the module. The first two parameters are channel sizes for the hidden C_H and output C_O layers. The third parameter is the size of the convolution kernel k, which is also the number of neighbors around each point in the k -Nearest Neighbor (k-NN) search. The final parameter, a dilation multiplier d, is introduced to broaden the neighborhood of the points.

Starting with a specific point p, the $k \cdot d$ nearest neighbors are identified by a k-NN search, such as 16 for $k = 8$ and $d = 2$. From these neighbors, k are randomly chosen, whilst the remaining ones are omitted. If the dilation parameter is one, all points are chosen. This set of points is denoted as P_k, with the corresponding features of these points in F_k.

In the next step, the neighborhood points P_k are translated into a local coordinate system with the central point p at the origin $(P_k - p)$. This eliminates dependence on absolute coordinates, because only the relative position of the points to each other is relevant.

Subsequently, the shifted points are passed through two different MLPs. The first, $\text{MLP}_\delta(\cdot)$, is applied to each point individually to lift them into a latent—and potentially canonical—feature space, which is a more abstract representation, similar to PointNet (Qi et al. 2017). The abstract neighborhood representation F_δ has its dimensionality given by the hidden channel parameter C_H. These features are then concatenated with the given features F_k, resulting in F_*, which later will be transformed. The second MLP, labeled as MLP$[k \times k]$ in Fig. 5, learns a transformation matrix of size $k \times k$ across the whole neighborhood. This transformation matrix $\mathcal{X}$ permutes and weights the features F_* by matrix multiplication.

Finally, these transformed features are convolved by a standard one-dimensional convolution. The kernel K has size k and is learned during the training of the neural network. The resulting features with output size C_O replace the previous features of the point cloud, thereby retraining the points' spatial positions. Mathematically, this result is described in Eq. 14. This can then be passed through further point sampling, pooling, or $\mathcal{X}$-Conv layers in the network.

All components of the $\mathcal{X}$-Conv, such as both MLPs, the convolution Conv$(\cdot)$, and the matrix multiplication, are differentiable. Therefore, the whole $\mathcal{X}$-Conv is differentiable and suitable for training a DNN with backpropagation.

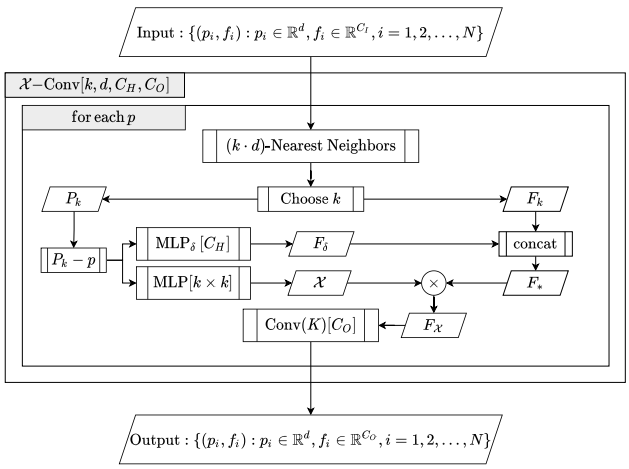


Fig. 5 Internal architecture of the $\mathcal{X}$-Conv module used in PointCNN

Finally, iterative farthest point sampling as described by Qi et al. (2017) is applied to reduce the input $x_1, x_2, \ldots, x_n$ to a smaller subset in an iterative manner. This ensures, together with the k-NN, that the receptive field of the following layers can grow.

3.2.2 Model Architecture

Li et al. (2018) provided a model zoo for different 2D and 3D benchmark datasets. The architectural design of the used model is based on the one used for the ModelNet40 (Wu et al. 2015) benchmark.

Figure 6a shows the general structure of PointCNN. The input to the model is a point cloud, which in this case is a single-segmented tree, comprising 1024 points in three-dimensional space. Further features can be assigned to each point. These include the laser intensity, normals, and multispectral features, as described in Sect. 2. Any combination of these, or none at all, can be used for the input. If none are chosen, the input represents the geometry of the point cloud. The first two $\mathcal{X}$-Conv layers are each followed by a downsampling layer to reduce the spatial dimensionality. Both subsequent convolution layers maintain the spatial size but increase the number of learned features. After four $\mathcal{X}$-Convs, the learned representation of a tree is further reduced via global mean pooling, which takes the mean of each feature across the whole object. Simply put, the whole point cloud is compressed into a single point with no location, which has 384 associated features. This embedding is then passed through three fully connected layers to produce an assignment probability for each of the four classes.

As the multispectral features (*MS*) are derived per tree, they describe the complete object rather than each point individually. Instead of assigning each point the same features, they are concatenated to the pooled features of the last $\mathcal{X}$-Conv layer. These learned features can be seen as descriptors for the whole object and are therefore in line with the *MS* features. Each point in the input can still have a combination of features attached to it. Here, the available choices are still the laser intensity and the normals. The adapted model structure is depicted in Fig. 6b. The last $\mathcal{X}$-Conv outputs five fewer features to compensate for the *MS* features. Moreover, the first fully connected layer keeps the same input size compared to the aforementioned model.

4 Experimental Setup

4.1 3DmFV-Net

The values of the 3DmFV are dependent on the absolute positions of the points in space rather than their relative positions to each other. Therefore, the point clouds of each single tree are centered around their center of gravity, i.e., the mean of their points. Furthermore, they are scaled to fit into a sphere with radius 1, as described in Sect. 2.3.1. Thus, the trees do not necessarily span the whole range of $[-1, 1]$ in their Z-dimension. The height of the trees in terms of normalized coordinates is not dependent on the actual height of the tree, but rather where the majority of the returning laser pulses were registered. Therefore, the point cloud of each individual tree was shifted and scaled, thereby spanning

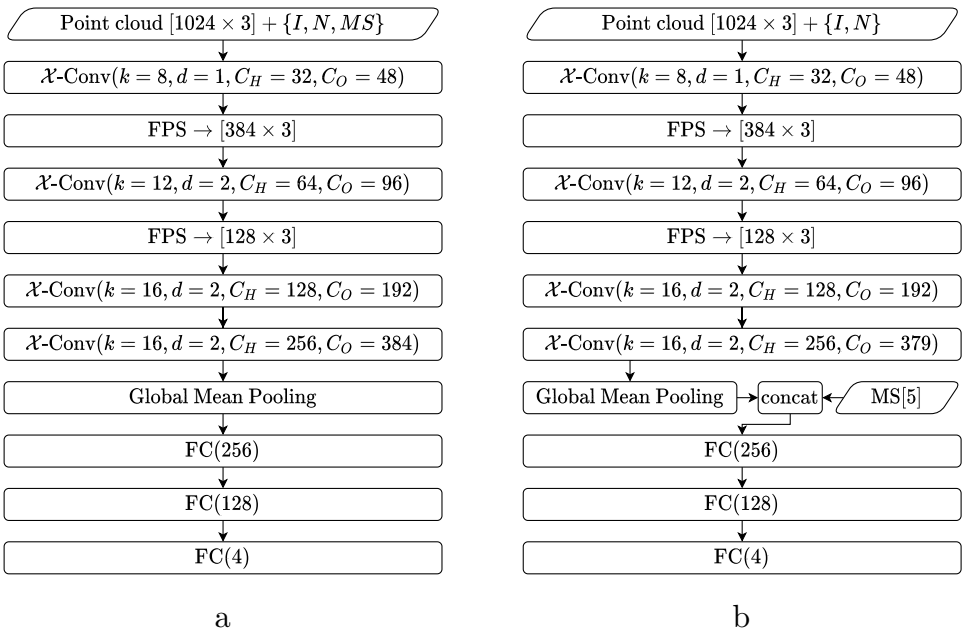


Fig. 6 **a** PointCNN model architecture with *MS* as a point feature. **b** PointCNN model architecture with *MS* as an object feature

from -1 to 1 in the Z-dimension. This may cause the values in the other two dimensions to not lie within the range $[-1, 1]$. The points that exceed this range are still captured in the 3DmFVs, because the GMM extends beyond it (see Sect. 3.1.2).

Moreover, the training set was augmented by inserting three more copies of each tree. These copies were rotated by 90°, 180°, and 270°, respectively. This behavior was as expected, because the 3DmFV are dependent on the absolute positions of the points. This means that the network is not rotation invariant with the exception that expected rotations of the objects are also learned.

The experiments involving the 3DmFV-Net differ mainly in the chosen grid size of the GMM. Ben-Shabat et al. (2018) suggest four different grid sizes and their according model architecture, as presented in Sect. 3.1.4. In this paper, these proposed sizes and their corresponding model architecture were employed. We simulated both uniform (denoted as $m \times m \times m$) and non-uniform grids (denoted as $m \times m \times n$). Parameter m is the base size of the grid and n is the changed grid size in the Z-dimension with the condition $m < n$. This is because trees were sampled more finely in their height and, thus, their characteristics in this dimension are captured better. For the base sizes of 3, 5, and 8, the number of sections for the third dimensions is set to 9, 10, and 12, respectively.

4.2 PointCNN

In contrast to the 3DmFV-Net, PointCNN is not dependent on the absolute positions of the point clouds. By design, this network is also capable of integrating more features than only the geometry (see also Sect. 5.2). The experiments for PointCNN used two slightly different architectural designs. First, a model which uses the geometry of the point clouds with none, or a combination of features (see Fig. 6a). In this approach, the *MS* features can be assigned to every point in each point cloud, although these features are describing the whole object. The second architecture is, for the most part, equivalent to the first one. Instead of using the *MS* features as point features, they are treated as object features and thus concatenated to the learned representation right before the FCN (see Fig. 6b). For both of these designs, the combinations of inputs are experimented with. All experiments use the geometry and none or up to three features. All configurations have in common that they always use the geometry GEOM of the point cloud. In total, 16 model configurations were tested.

4.3 Training of the Networks and Hyperparameters

Both networks were trained on the same dataset with exactly the same train/val/test split. The training was carried out on a workstation with 32 GB RAM and a NVIDIA Quadro RTX 4000 with 8 GB GPU memory. The number of samples per batch for PointCNN and 3DmFV-Net was 16 and 64 respectively. In each epoch, the samples for the mini-batches were shuffled. To prevent the model from overfitting, an early stopping was applied. If there is was no improvement in the validation loss within 20 steps, the learning process was stopped. The model parameters at the epoch with the lowest validation error were considered the best for a generalized model.

The hyperparameters used to train both networks are shown in Table 5.

4.4 Accuracy Metrics

To quantify the performance of our models, we used the manually assigned class labels (i.e., true values) and the classified labels to calculate confusion matrices along with values for OA and the F_1-score using recall and precision.

Table 5 Hyperparameters used in the learning process for PointCNN and 3DmFV

Hyperparameter	PointCNN	3DmFV
Batch size	16	64
Loss function	Cross Entropy Loss	
Optimizer	Adam (Kingma and Ba 2015)	
Base learning rate	0.001	
Learning rate scheduling	None	StepLR (step_size=20, $\gamma = 0.7$)
Number of epochs	Not limited	min. 10
Stopping criterion	Early stopping if no improvement in validation loss within 20 steps	
Batch norm momentum	–	0.1
Dropout rate	0.5	0.3

5 Experimental Results

5.1 Main Outcomes

In our experiments, we demonstrate the performance of two DNNs, 3DmFV-Net and PointCNN, to classify tree species and dead trees using lidar point clouds captured in BFNP. In terms of OA, PointCNN performs better, with an OA of 87.0%, in comparison to an OA of 75.1% for 3DmFV-Net (see Tables 6 and 7). Interestingly, if PointCNN is applied, additional features calculated from the lidar intensity and the normals do not significantly contribute to the classification results. Instead, the multispectral feature NDVI improves the results most notably. For both networks, we found apparent confusions between spruces and dead trees with crown.

5.2 3DmFV-Net

Table 6 summarizes the best results of our experiments using 3DmFV-Net. Relevant confusion matrices are provided in Fig. 7. All in all, the $m = 8$ grid performs best with an OA of 73.2%. This grid size also reaches the highest F_1-scores among all tested 3DmFV-Net configurations for the two 'living' classes, *coniferous* and *deciduous*. Although these scores are relatively high, the score for the class *dead tree* is the lowest overall at only 35.2%. For this class, the error matrix in Fig. 7c shows that only 51 of the 139 objects are detected correctly. False negatives mainly occur in *coniferous* and *snags* at 38 and 53, respectively. Only 8 *snags* and 60 *coniferous* are erroneously classified as *dead tree*.

5.3 PointCNN

The first experiment consists of learning a PointCNN model using only the geometry of the point clouds. As shown in Fig. 6b, an OA of 69.7% is reached if the model structure is applied. This is in the same order of magnitude as 3DmFV-Net. Furthermore, this configuration reports the highest F_1-score for the class *snag* achieved by PointCNN in this study. The confusion matrix (Fig. 8a) shows confusions mainly between *dead tree* and *coniferous*. When adding the unnormalized intensities as a feature to each point, the OA decreases significantly to 61.5%. There is an increase of falsely predicted dead trees across all classes compared to the network that uses only the geometry, as shown in Fig. 8b (GEOM_I). This leads to *coniferous* and *dead tree* having the lowest recall and precision, respectively. Standardizing the intensities before the learning process results in an OA of 69.5%. This is an increase compared to the unnormalized intensities but not significantly different compared to that which uses only the geometry. Using the power-transformed intensities I_s, an increase in accuracy can be observed. This input reaches an OA of 71.5%. If compared to the GEOM configuration, the confusion matrix (Fig. 8d) shows that this brings an improvement in true positives for *dead tree* and *coniferous*, as well as less misclassification between them.

Employing the estimated normals N of the point cloud in the model GEOM_N as the only feature besides the geometry GEOM shows no increase in classification performance. The OA is lower than that using only the GEOM. Significantly more *coniferous* trees are predicted as *dead tree*, which also leads to a smaller F_1-score in the *coniferous* class (see Fig. 8e).

The biggest change in performance can be observed when using the multispectral features MS. When using a copy of these object features as an additional feature to each point (MS_{feat}), an OA of 86.0% is reached. The recurring misclassifications between *coniferous* and *dead tree* have almost been eliminated (see Fig. 8f). *Deciduous* trees are detected almost perfectly with an F_1-score of 96.5%, the highest score in this study. As a result of these fewer misclassifications, *coniferous* and *dead trees* also show high F_1-scores of 91.0% and 72.3%, respectively.

The second possibility for using *MS* features is to provide them on an object basis. Instead of using a copy of the MS as a feature, these features are concatenated to those resulting from the $\mathcal{X}$-Conv layers. These are then passed through the FCN. This is the case for the second proposed model structure (as shown in Fig. 6b). Similar to the use of MS as point features, the combination of GEOM and MS_{FCN} reaches an OA of 85.6%. The F_1-scores are all in the same order of magnitude as in the MS_{feat} configuration. The comparison of their confusion matrices (see Fig. 8f, g) shows minor differences. Compared to GEOM_MS_{feat}, GEOM_MS_{FCN} shows fewer true positives for *coniferous* but more in all other classes. There are 17 *coniferous* trees classified as *deciduous*, producing lower F_1-scores compared to MS_{feat}. Although the OA is lower than for the GEOM_MS_{feat} configuration, all tests confirm

Table 6 Results of 3DmFV-Net with uniform and non-uniform grids

3DmFV-Net	OA	F_1-score			
		Coniferous	Deciduous	Snag	Dead tree
$3 \times 3 \times 3$	0.683	0.684	0.897	0.696	0.517
$5 \times 5 \times 5$	0.643	0.617	0.918	0.669	0.372
$8 \times 8 \times 8$	**0.732**	**0.752**	**0.928**	0.751	0.352
$3 \times 3 \times 9$	0.565	0.432	0.854	0.664	0.559
$5 \times 5 \times 10$	0.635	0.526	0.909	0.718	**0.628**
$8 \times 8 \times 12$	0.707	0.690	0.908	**0.768**	0.441

The best results in each column are underlined. Bold values are the best overall scores for any 3DmFV-Net configuration

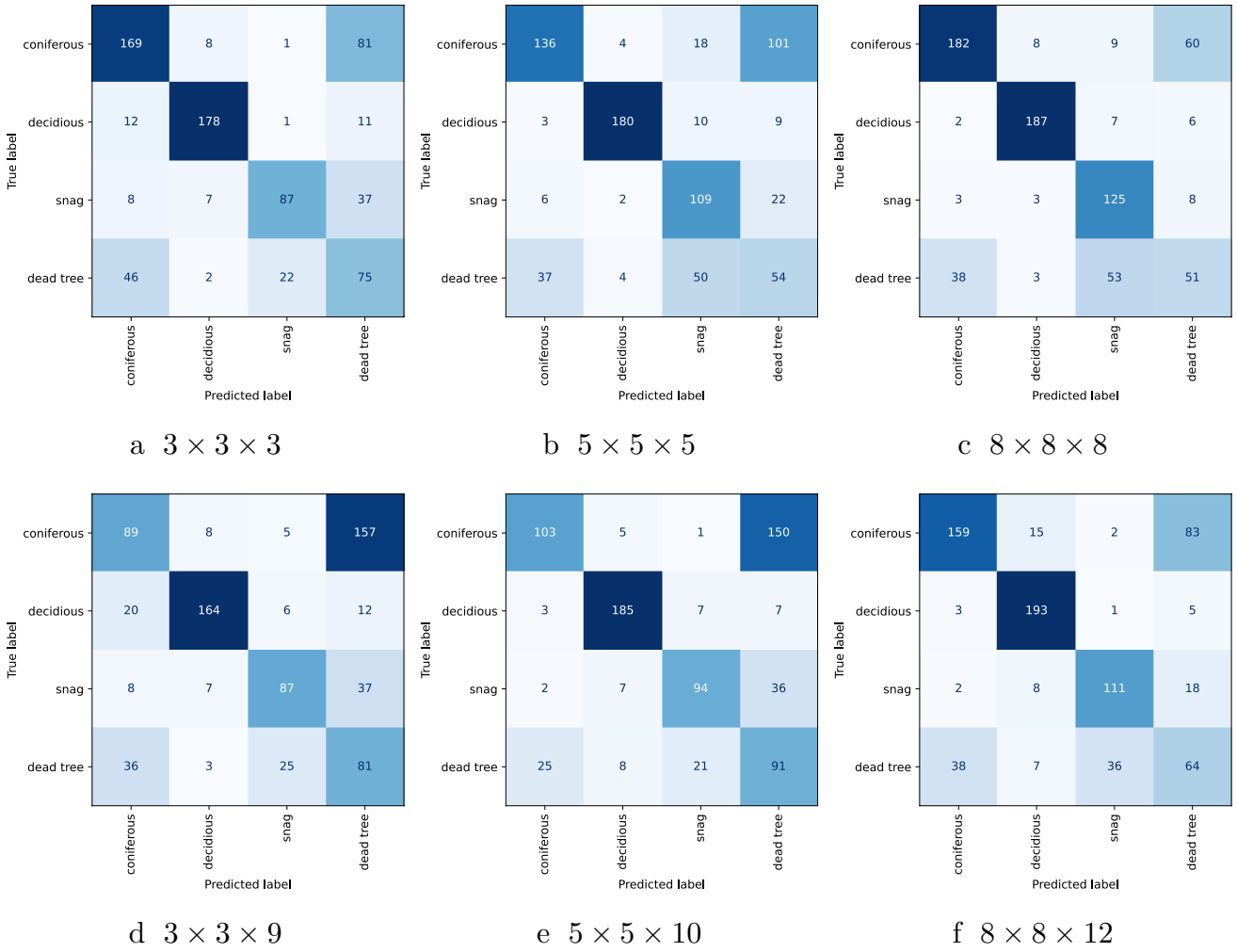


Fig. 7 Confusion matrices for experiments with 3DmFV-Net using uniform and non-uniform grids. Subfigures **a–f** refer to uniform and non-uniform grids

that using the MS as an object feature in combination with other features yields better classification results.

Further tests include input data with two types of features. Combining the laser intensities with the MS results in higher accuracy than using only the intensities as a single feature. The combination of power-transformed intensities I_s and MS_{FCN} produces the highest OA among all configurations at 87.0%. This input configuration also results in the highest F_1-score for *coniferous* (91.5%) and *dead tree* (76.5%) among the PointCNN experiments. The confusion matrix (see Fig. 8i) shows 236 correctly identified *coniferous* trees and little confusion in the other classes. This is especially true for *dead tree*. Moreover, the confusion between *coniferous* and *dead tree*, which is the most prominent confusion in the experiments, is relatively small (13 and 12 misclassifications, respectively). With a total of 44 misclassifications, the most falsely predicted trees result from the *snag* class and are assigned to *dead tree*. Compared to the best configuration, the combination of I_z and MS_{FCN} performs slightly worse in terms of OA (= 85.4%) and relevant F_1-scores (see also confusion matrices in Fig. 8h, i).

The remaining results, shown in Table 7 (GEOM_N_I_s, GEOM_N_I_s_MS_{feat}, GEOM_N_I_s_MS_{FCN}), demonstrate that normalizing the intensities generally shows an increase in accuracy, whilst the addition of the normals impairs the results. The biggest change in classification performance results from adding MS features; this produces a significant increase in OA and distinction between classes.

Table 7 Results of the PointCNN

PointCNN	OA	F_1-score			
		Coniferous	Deciduous	Snag	Dead tree
GEOM	0.697	0.657	0.894	**0.803**	0.431
GEOM_I	0.615	0.543	0.828	0.709	0.424
GEOM_I_z	0.695	0.675	0.898	0.741	0.477
GEOM_I_s	0.715	0.694	0.916	0.779	0.480
GEOM_N	0.678	0.612	0.874	0.795	0.434
GEOM_MS_{feat}	0.860	0.910	**0.965**	0.758	0.723
GEOM_MS_{FCN}	0.856	0.895	0.945	0.782	0.731
GEOM_I_MS_{feat}	0.769	0.775	0.866	0.683	0.695
GEOM_I_z_MS_{feat}	0.846	0.888	0.924	0.766	0.737
GEOM_I_s_MS_{feat}	0.851	0.893	0.934	0.763	0.749
GEOM_I_MS_{FCN}	0.773	0.819	0.892	0.651	0.678
GEOM_I_z_MS_{FCN}	0.854	0.897	0.946	0.763	0.727
GEOM_I_s_MS_{FCN}	**0.870**	**0.915**	0.953	0.775	**0.765**
GEOM_N_I_s	0.714	0.709	0.902	0.754	0.500
GEOM_N_I_s_MS_{feat}	0.855	0.887	0.950	0.797	0.728
GEOM_N_I_s_MS_{FCN}	0.854	0.899	0.945	0.766	0.735

The best result in each column is marked in bold text. MS_{feat} denotes cases in which each feature is assigned to a point. MS_{FCN} denotes cases in which object features enter the FCN at the end of the network. The laser intensities without any scaling are denoted as I. Experiments for which these values were standardized across the whole dataset are denoted as I_z. The third notation I_s is used when the values are power-transformed and subsequently standardized, as shown in Sect. 2.3.4

6 Discussion

6.1 3DmFV-Net

As shown in Sect. 3.1, the 3DmFV-Net uses a hybrid approach. In the experiments, the main difference is the choice of grid size for the GMM, as the design of the network only allows the use of the point clouds' geometry.

The smallest grid has a base length of three and performs well in detecting *deciduous* trees, but struggles to keep the other three classes apart from each other. When the base size increases to five, the OA decreases slightly. When the base size increases to eight, the OA and all F_1-scores reach the highest values in this set of experiments (see Table 7). This could suggests that a finer grid—thus a larger base size—will increase accuracy. However, the s with sizes of 10 and 12 show that this is not the case. Rather, a grid size of eight performs best in terms of OA and F_1-scores.

In conclusion, a grid of $8 \times 8 \times 8$ with data augmentation in the training set is optimal. However, this network could still be improved by more extensive hyperparameter tuning, the introduction of other symmetric functions of the 3DmFV, or more sophisticated data augmentation. The network could also be tuned by integrating other features in the network design, such as multispectral information, which guaranteed a major improvement in classification accuracy in the PointCNN.

6.2 PointCNN

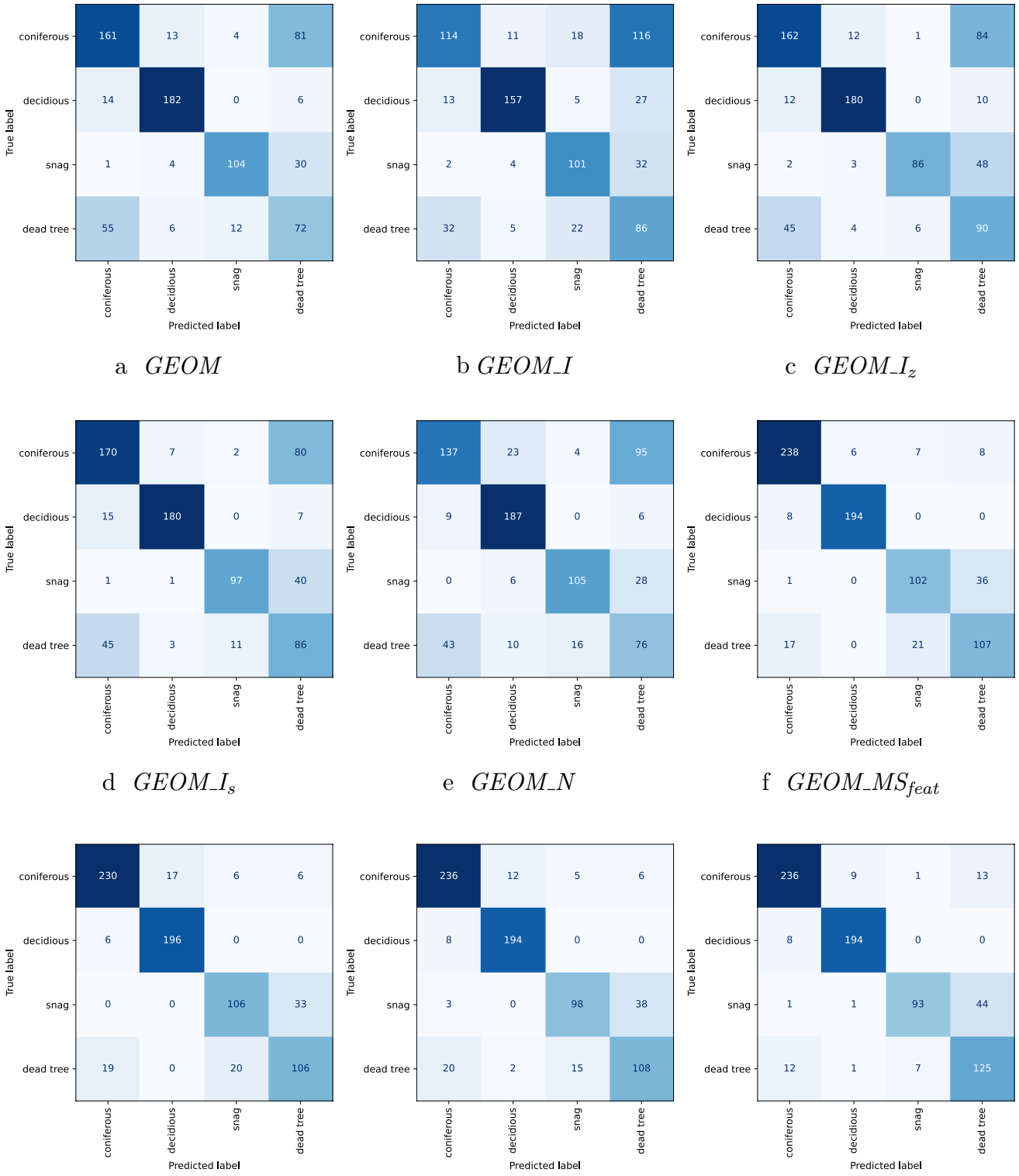
When using the different inputs (e.g., laser intensities, normals, and multispectral features), some different effects on classification can be observed. The addition of the intensities as features diminishes classification performance significantly (OA of 61.5% compared to 69.7% with only the geometry; see Table 7). This is influenced by the power law distribution of the values. As expected, after power-transforming and standardizing, the intensities no longer reduce classification accuracy. Interestingly, the OA also does not increase in a significant way. This indicates that the network cannot derive any meaningful information from the laser intensities. The sole standardization of the intensities seems to be sufficient.

The normals of the points are another feature that can be used as an input in combination with the geometry of the point cloud. The results of the experiments show that the normals usually do not improve the classification significantly. This is likely due to the rough and irregular surfaces of each tree's point cloud. The best performing feature combination GEOM_I_s_MS_{FCN} does not include the normals. Future experiments can therefore leave these out and reduce computational effort.

The third feature is a composite of the descriptive statistics derived from the multispectral orthophotos. Adding these to the network input, either as point-based features or object-based features, improves the classification results significantly. The OA reaches 86.0% and a nearly perfect F_1-score of 96.5% for *deciduous* trees in combination with the geometry (see GEOM_MS_{feat} in Table 7). These results are achieved by inserting a copy of the object feature to the feature ensemble of each point carrying redundant information. However, the results of the other experiments show that using the MS features in the FCN generally performs slightly better compared to the feature approach. The best-performing input configuration GEOM_I_s_MS_{FCN} also uses this model architecture. The sole use of the MS in combination with the geometry already performs nearly as well as the best model. Unexpectedly, using intensities only marginally increases classification accuracy, whilst the use of normals results in a small decrease.

6.3 Misclassifications

Both networks, PointCNN and 3DmFV-Net, show similar patterns in their classification results—especially visible in the confusion matrices—when not using the MS features. Generally, the *deciduous* class was classified very



a *GEOM* b *GEOM_I* c $GEOM\_I_z$

d $GEOM\_I_s$ e *GEOM_N* f $GEOM\_MS_{feat}$

g $GEOM\_MS_{FCN}$ h $GEOM\_I_z\_MS_{FCN}$ i $GEOM\_I_s\_MS_{FCN}$

Fig. 8 Confusion matrices of selected PointCNN configurations

consistently with high F_1-scores (> 80%), and thus was well distinguished from the other classes. Most misclassifications occur through falsely assigning *coniferous trees* to the *dead tree* class, and similarly vice versa, independent of the used network (cf. Figs. 7 and 8). These false negatives were most apparent in the 3DmFV-Net configurations but also in the PointCNN when the geometry was combined with the unnormalized laser intensities (see GEOM_I in Fig. 8b).

The BFNP is infested with bark beetles, which mainly kill spruces, the main species in the *coniferous* class. These dying spruces constituted the majority—if not all—of the samples in the *dead tree* class. Because they still have their crown, they are geometrically very similar to their living conspecifics. This often leads to misclassification between these two classes. The MS features, derived from the NDVI, mostly eliminates these erroneous classifications, however. Hence, the difference between living and dead trees is best described through these features. In other words, multispectral information captured by an optical sensor is necessary to gain better discrimination between dead and living spruces.

Another pattern that occurred in the aforementioned classifications and their error matrices was confusion between *dead tree* and *snags*. When a tree is dying, such as because of the bark beetle infestation, it continuously loses its tree crown and branches. It is considered a *snag* when the tree has lost a substantial amount of its branches. The distinction between *dead tree* and *snag* in the present study was based on the subjective perceptions of three research assistants (Briechle et al. 2021). This may have been the reason for the observed difficulties in distinguishing the two classes in the two deep learning networks. For example, false positives were assigned to both classes. Furthermore, as the confusion matrices show, this effect could not be mitigated completely by the strategic use of the input features or grid sizes. However, some model configurations show better distinction in one area. The 3DmFV-Net $8 \times 8 \times 8$ configuration (see Table 7) exhibited significantly less mislabeling from *dead tree* to *snags* compared to the other configurations of this network. Similarly, the PointCNN configuration GEOM_I_z showed fewer misclassifications from *snags* to *dead tree*. For future studies, either further features should be crafted to generate a measure for distinction, or a more objective way of labeling the training data should be developed. For future applications, it is also possible to use a threshold on the certainties of the classification. Tree samples resulting in low certainty values for each class could then be flagged and pushed to manual assessment.

6.4 Comparison to Related Work

We compare our results with the study of Briechle et al. (2021) that was not only conducted in the same study area but also used the exact same dataset. Therefore, a one-to-one comparison can be made here. PointNet++ was used as a baseline method for comparison to their proposed network. Similar to PointCNN, this DNN is also point-based. Considering only the geometric information from the point clouds of the single trees, PointNet++ reached an OA of 85.7% with F_1-scores of higher or equal to 72%. PointCNN reached an OA of 69.7% with low F_1-scores in the *coniferous* and *dead tree* class. 3DmFV-Net could also not perform that well with a maximum OA of 73.2% using solely geometric information. When further features were taken into account, i.e., laser intensity and multispectral features, PointCNN gains a rather large performance increase reaching 87.0% (+ 18.3% points) in classification accuracy. Such an increase in performance does not occur with PointNet++, which can gain 3.6% points for an OA of 89.3%. PointNet++ additionally shows higher F_1-scores throughout. However, both networks are in the same order of magnitude. This suggests that PointNet++ performs better on pure geometry data than PointCNN in this kind of real-world application. Furthermore, PointCNN seems to rely on further initial features to reach results on par with PointNet++.

In their work, Briechle et al. (2021) also proposed their own DNN. Silvi-Net uses a multi-view-based approach projecting the point clouds of individual trees onto 12 image planes. Moreover, image patches from the aerial orthophoto are created using the enclosing polygons of the trees. Employing only geometric features from the point clouds, this network was able to reach an OA of 84.7%, which is significantly higher than the 69.7% achieved by PointCNN and 73.2% of 3DmFV-Net. Similar to PointCNN, Silvi-Net experiences a decrease (−1.2% points) in classification accuracy when using the additional—not power transformed—laser intensities in the generated images. Interestingly, the same misclassification patterns, as discussed in Sect. 6.3, can also be observed with Silvi-Net. However, the network configuration using geometry, laser intensities, and multispectral images attained the highest classification performance. The OA was 91.5% that is higher than both PointCNN (87.0%) and the used baseline PointNet++ (89.3%). This suggests that multi-view approaches could perform better than point-based DNNs in tree species classification.

7 Conclusions

In this paper, the two DNNs PointCNN and 3DmFV-Net were investigated for the application to tree species classification using airborne lidar data. We demonstrated that both architectures are capable of classifying both living and dead trees simultaneously. 3DmFV-Net is limited to the geometry of the single trees, whereas PointCNN offers integration of further features by design. The geometric information from the point clouds is sufficient for the distinction between

coniferous and *deciduous* trees. The synergetic use of normalized intensities and multispectral information improved classification accuracy significantly. This effect could not be seen with unnormalized intensities, or the standardized ones (cf. Table 7). Including the point normals as additional features did not lead to improved results. However, the features derived from a multispectral orthophoto proved to be crucial for differentiating between dead trees and their living counterparts. PointCNN even reached an OA of 87.0% that can be considered operational in practical applications. Both networks are possibly sensitive to the point count of each tree and to the point density of the lidar capture. In further studies, the impact of the point density on the accuracy performance could be evaluated. To this end, a second dataset would also be valuable to show the transferability of this study to other forest areas and tree classes.

Acknowledgements The authors would like to thank Prof. Dr. Marco Heurich, Head of the Department of Visitor Management and National Park Monitoring, for providing the remote sensing data in the study area BFNP (lidar data and multispectral images).

Funding Open Access funding enabled and organized by Projekt DEAL. The authors would like to thank Esri Deutschland GmbH for funding the master's thesis leading to this paper.

Declarations

Conflict of interest The authors declare that there is no conflict of interest.

Code availability We used publicly available open source code, and in addition have developed our own code. For the open source code, we have included the corresponding links in the text. All codes used for the experiments can be found at https://www.github.com/hellwue/TreeSpeciesClassification. The used data cannot be published, as the authors do not hold the rights for it.

References

Amiri N, Krzystek P, Heurich M, Skidmore A (2019) Classification of tree species as well as standing dead trees using triple wavelength ALS in a temperate forest. Remote Sens 11(22):2614. https://doi.org/10.3390/rs11222614

Ben-Shabat Y, Lindenbaum M, Fischer A (2018) 3DmFV: three-dimensional point cloud classification in real-time using convolutional neural networks. IEEE Robot Autom Lett 3(4):3145–3152. https://doi.org/10.1109/LRA.2018.2850061

Bibelriether H (1989) Windwürfe und Borkenkäfer im Nationalpark Bayerischer Wald. Nationalpark 6:24–27

BMEL (2015) The forests in Germany—selected results of the third national forest inventory. Bundesministerium für Ernährung und Landwirtschaft (BMEL)

Box GEP, Cox DR (1964) An Analysis of Transformations. J R Stat Soc Ser B (Methodol) 26(2):211–243. https://doi.org/10.1111/j.2517-6161.1964.tb00553.x

Briechle S, Krzystek P, Vosselman G (2021) Silvi-Net—a dual-CNN approach for combined classification of tree species and standing dead trees from remote sensing data. Int J Appl Earth Observ Geoinform 98(102):292. https://doi.org/10.1016/j.jag.2020.102292

Briechle S, Krzystek P, Vosselman G (2020) Classification of tree species and standing dead trees by fusing Uav-based lidar data and multispectral imagery in the 3D deep neural network Pointnet++. In: ISPRS annals of the photogrammetry, remote sensing and spatial information sciences, vol V-2-2020. Copernicus GmbH, pp 203–210. https://doi.org/10.5194/isprs-annals-V-2-2020-203-2020

FAO (2020) Global forest resources assessment 2020: key findings. FAO, Rome, Italy. https://doi.org/10.4060/ca8753en

Fassnacht FE, Latifi H, Stereńczak K et al (2016) Review of studies on tree species classification from remotely sensed data. Remote Sens Environ 186:64–87. https://doi.org/10.1016/j.rse.2016.08.013

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge

Griffiths D, Boehm J (2019) A review on deep learning techniques for 3D sensed data classification. Remote Sens 11(12):1499. https://doi.org/10.3390/rs11121499

Guo Y, Wang H, Hu Q et al (2021) Deep learning for 3D point clouds: a survey. IEEE Trans Pattern Anal Mach Intell 43(12):4338–4364. https://doi.org/10.1109/TPAMI.2020.3005434

Hamraz H, Jacobs NB, Contreras MA et al (2019) Deep learning for conifer/deciduous classification of airborne LiDAR 3D point clouds representing individual trees. ISPRS J Photogramm Remote Sens 158:219–230. https://doi.org/10.1016/j.isprsjprs.2019.10.011

Hartling S, Sagan V, Sidike P et al (2019) Urban tree species classification using a WorldView-2/3 and LiDAR data fusion approach and deep learning. Sensors 19(6):1284. https://doi.org/10.3390/s19061284

Heurich M, Reinelt A, Fahse L (2001) Die Buchdruckermassenvermehrung Im Nationalpark Bayerischer Wald. Waldentwicklung im bergwald nach windwurf und borkenkäferbefall 14:9–48

He K, Zhang X, Ren S et al (2016) Deep Residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), pp 770–778. https://doi.org/10.1109/CVPR.2016.90

Huang G, Liu Z, Van Der Maaten L et al (2017) Densely connected convolutional networks. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), pp 2261–2269. https://doi.org/10.1109/CVPR.2017.243

Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Bach F, Blei D (eds) Proceedings of the 32nd international conference on machine learning, proceedings of machine learning research, vol 37. PMLR, Lille, France, pp 448–456. https://proceedings.mlr.press/v37/ioffe15.html

Jiao Y, Yin Z (2020) A two-phase cross-modality fusion network for robust 3D object detection. Sensors 20(21):6043. https://doi.org/10.3390/s20216043

Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. arXiv:1412.6980 [cs]

Korpela I, Ørka H, Maltamo M et al (2010) Tree species classification using airborne LiDAR—effects of stand and tree parameters, downsizing of training set, intensity normalization, and sensor type. Silva Fennica 44(2). https://doi.org/10.14214/sf.156

Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L et al (eds) Advances in neural information processing systems, vol 25. Curran Associates Inc, Red Hook

Krzystek P, Serebryanyk A, Schnörr C et al (2020) Large-scale mapping of tree species and dead trees in Šumava National Park and Bavarian Forest National Park using lidar and multispectral imagery. Remote Sens 12(4):661. https://doi.org/10.3390/rs12040661

Lausch A, Fahse L, Heurich M (2011) Factors affecting the spatio-temporal dispersion of *Ips typographus* (L.) in Bavarian Forest National Park: a long-term quantitative landscape-level analysis. Forest Ecol Manag 261(2):233–245. https://doi.org/10.1016/j.foreco.2010.10.012

LeCun Y, Bottou L, Orr GB et al (1998) Efficient BackProp. Neural networks: tricks of the trade, this book is an outgrowth of a 1996 NIPS workshop. Springer, Berlin, pp 9–50

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539

Li Y, Bu R, Sun M et al (2018) PointCNN: convolution on χ-transformed points. arXiv:1801.07791 [cs]

Li H, Hu B, Li Q et al (2020) CNN-based tree species classification using airborne lidar data and high-resolution satellite image. In: IGARSS 2020—2020 IEEE international geoscience and remote sensing symposium. IEEE, Waikoloa, HI, USA, pp 2679–2682. https://doi.org/10.1109/IGARSS39084.2020.9324011

Liu M, Han Z, Chen Y et al (2021) Tree species classification of LiDAR data based on 3D deep learning. Measurement 177(109):301. https://doi.org/10.1016/j.measurement.2021.109301

Maturana D, Scherer S (2015) VoxNet: a 3D convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, Hamburg, Germany, pp 922–928. https://doi.org/10.1109/IROS.2015.7353481

Mäyrä J, Keski-Saari S, Kivinen S et al (2021) Tree species classification from airborne hyperspectral and LiDAR data using 3D convolutional neural networks. Remote Sens Environ 256(112):322. https://doi.org/10.1016/j.rse.2021.112322

McRoberts RE, Tomppo EO (2007) Remote sensing support for national forest inventories. Remote Sens Environ 110(4):412–419. https://doi.org/10.1016/j.rse.2006.09.034

Müller M, Mayer M, Job H (2008) Totholz und Borkenkäfer im Nationalpark Bayerischer Wald aus touristischer Perspektive. In: Die Destination Nationalpark Bayerischer Wald als regionaler Wirtschaftsfaktor. Hubert Job, pp 100–116

Mustafić S, Schardt M (2019) Deep Learning-basierte Baumartenklassifizierung auf Basis von ALS-Daten. In: Dreiländertagung der DGPF, der OVG und der SGPF, Wien, Austria, 2019, 20.-22. Februar, pp 527–536

Nationalparkverwaltung Bayerischer Wald (2021) Der Nationalpark Bayerischer Wald im Porträt. https://www.nationalpark-bayerischer-wald.bayern.de/ueber_uns/steckbrief/index.htm

Perronnin F, Sánchez J, Mensink T (2010) Improving the Fisher Kernel for large-scale image classification. In: Daniilidis K, Maragos P, Paragios N (eds) Computer vision—ECCV 2010, lecture notes in computer science. . Springer, Berlin, pp 143–156. https://doi.org/10.1007/978-3-642-15561-1_11

Qi CR, Su H, Kaichun M, Guibas LJ (2017) PointNet: deep learning on point sets for 3D classification and segmentation. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), pp 77–85. https://doi.org/10.1109/CVPR.2017.16

Qi CR, Yi L, Su H et al (2017) PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Guyon I, Luxburg UV, Bengio S et al (eds) Advances in neural information processing systems, vol 30. Curran Associates Inc, Red Hook

Rouse JW Jr, Haas RH, Schell JA et al (1974) Monitoring vegetation systems in the great plains with ERTS. NASA special publication, vol 351, p 309

Sánchez J, Perronnin F, Mensink T et al (2013) Image classification with the fisher vector: theory and practice. Int J Comput Vis 105(3):222–245. https://doi.org/10.1007/s11263-013-0636-x

Seidel D, Annighöfer P, Thielman A et al (2021) Predicting tree species from 3D laser scanning point clouds using deep learning. Front Plant Sci. https://doi.org/10.3389/fpls.2021.635440

Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905. https://doi.org/10.1109/34.868688

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 [cs]

Su H, Maji S, Kalogerakis E et al (2015) Multi-view convolutional neural networks for 3D shape recognition. In: 2015 IEEE international conference on computer vision (ICCV), pp 945–953. https://doi.org/10.1109/ICCV.2015.114

Sun Y, Huang J, Ao Z et al (2019a) Deep learning approaches for the mapping of tree species diversity in a tropical wetland using airborne LiDAR and high-spatial-resolution remote sensing images. Forests 10(11):1047. https://doi.org/10.3390/f10111047

Sun Y, Xin Q, Huang J et al (2019b) Characterizing tree species of a tropical wetland in Southern China at the individual tree level based on convolutional neural network. IEEE J Sel Top Appl Earth Observ Remote Sens 12(11):4415–4425. https://doi.org/10.1109/JSTARS.2019.2950721

Szegedy C, Liu W, Jia Y et al (2015) Going deeper with convolutions. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), pp 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Szegedy C, Vanhoucke V, Ioffe S et al (2016) Rethinking the inception architecture for computer vision. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), pp 2818–2826. https://doi.org/10.1109/CVPR.2016.308

Wang L, Zhao D, Wu T et al (2020) Drosophila-inspired 3D moving object detection based on point clouds. Inf Sci 534:154–171. https://doi.org/10.1016/j.ins.2020.05.006

Wu Z, Song S, Khosla A et al (2015) 3D ShapeNets: a deep representation for volumetric shapes. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, Boston, MA, USA, pp 1912–1920. https://doi.org/10.1109/CVPR.2015.7298801

Yao W, Krzystek P, Heurich M (2012) Identifying standing dead trees in forest areas based on 3D single tree detection from full waveform lidar data. ISPRS Ann Photogramm Remote Sens Spat Inf Sci I-7:359–364. https://doi.org/10.5194/isprsannals-I-7-359-2012

Zhou QY, Park J, Koltun V (2018) Open3D: a modern library for 3D data processing. arXiv:1801.09847 [cs]