HAL open science

Segmentation of structural parts of rosebush plants with 3d point-based deep learning methods

Kaya Turgut, Helin Dutagaci, Gilles Galopin, David Rousseau

▶ **To cite this version:**

Kaya Turgut, Helin Dutagaci, Gilles Galopin, David Rousseau. Segmentation of structural parts of rosebush plants with 3d point-based deep learning methods. Plant Methods, 2022, 18 (20), pp.00857-3. 10.1186/s13007-022-00857-3 . hal-03144153v1

HAL Id: hal-03144153

https://institut-agro-rennes-angers.hal.science/hal-03144153v1

Submitted on 17 Feb 2021 (v1), last revised 28 Mar 2022 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

arXiv:2012.11489v1 [cs.CV] 21 Dec 2020

SEGMENTATION OF STRUCTURAL PARTS OF ROSEBUSH PLANTS WITH 3D POINT-BASED DEEP LEARNING METHODS

A PREPRINT

Kaya Turgut
Department of Electrical-Electronics Engineering
Eskisehir Osmangazi University
Eskisehir, Turkey

Helin Dutagaci
Department of Electrical-Electronics Engineering
Eskisehir Osmangazi University
Eskisehir, Turkey
hdutagaci@ogu.edu.tr

Gilles Galopin
INRAe, UMR1345
Institut de Recherche en Horticulture et Semences
Angers, France

David Rousseau
Laboratoire Angevin de Recherche en Ingénierie des Systèmes (LARIS)
Université d'Angers
Angers, France
david.rousseau@univ-angers.fr

December 22, 2020

ABSTRACT

Segmentation of structural parts of 3D models of plants is an important step for plant phenotyping, especially for monitoring architectural and morphological traits. This work introduces a benchmark for assessing the performance of 3D point-based deep learning methods on organ segmentation of 3D plant models, specifically rosebush models. Six recent deep learning architectures that segment 3D point clouds into semantic parts were adapted and compared. The methods were tested on the ROSE-X data set, containing fully annotated 3D models of real rosebush plants. The contribution of incorporating synthetic 3D models generated through Lindenmayer systems into training data was also investigated.

Keywords Plant part segmentation · Point cloud · Virtual plant · Deep learning · Semantic segmentation · Phenotyping

1 Introduction

Automatic plant phenotyping based on computer vision techniques has become essential to enable high throughput experiments in botanical and agricultural research [1]. Several measurements related to plant phenotyping require segmentation of plant parts, such as branches and individual leaves. A large body of research has been conducted in recent decades for organ segmentation of plants through 2D images and 3D reconstructions using machine learning approaches [2–13].

Deep learning techniques have become popular and commonplace for 2D image-based plant segmentation and phenotyping [14–18]. Despite this trend, the application of deep learning on full 3D point clouds of plants has not emerged yet. After the publication of the PointNet architecture [19], there has been an explosion of research on 3D point cloud processing through deep learning [20–22]. However, their assessment are limited to a few data sets of artificial objects such as tables and chairs, indoor scenes of buildings and LiDAR scans of outdoor environments [20].

The main factor that impedes the 3D deep learning techniques from being applied to plant phenotyping is the lack of large annotated 3D plant data sets [23]. Even moderate size annotated data sets of full plant models are not available. As opposed to the speed of acquiring and annotating 2D images, the procedures for 3D model reconstruction and annotation of real plants are time-demanding and error-prone. Incorporation of synthetic plants through generative models such as Lindenmayer systems (L-systems) [24, 25] into training data are effective with 2D plant phenotyping [26]. The same strategy of creating synthetic 3D plant models can be followed to supply sufficient training data to machine learning frameworks [23].

Exploration of the performance of recent deep learning techniques on 3D plant phenotyping is imperative. Deep learning approaches have the capability of simultaneously extracting relevant information from the data at various scales and learning to design classifiers that model the variability in the data. They have been proven to outperform machine learning methods that rely on hand-crafted features. However, the recently developed 3D point-based deep learning architectures have not previously been analyzed for their suitability for organ segmentation of full 3D plant models.

The objective of this work is to address this lack of analysis and to provide a benchmark for application of 3D point-based deep learning methods to plant part segmentation. The target data set is the recently introduced ROSE-X data set, which includes eleven 3D models of real rosebush models obtained through X-ray imaging [27]. The models are fully annotated with three semantic labels: 1) Flower, 2) Leaf, and 3) Stem (branches and petioles). In order to augment the training data, 48 synthetic rosebush (Rosa x hybrida) models were generated using a simulator based on L-networks. As baseline methods, six recent 3D point-based deep learning architectures were adapted and evaluated for the segmentation of rosebush plants to their structural parts.

The contributions of this work include:

- a first attempt of applying various 3D point-based deep learning techniques to the problem of the segmentation of structural parts of full 3D models of real plants;
- incorporation of synthetic 3D plant models for augmenting the training data required by 3D deep learning methods;
- a benchmark for future developments of 3D point-based architectures targeting 3D plant phenotyping.

2 Related Works

Deep learning methods, in contrast to the use of hand-crafted features, have the advantage of being able to learn features from raw input data and model the within-class and between-class variations of the features simultaneously. Their application to image-based plant phenotyping and part-segmentation is vast [28]. Despite its advantages, deep learning techniques require large amount of annotated data to avoid overfitting. The bottleneck in machine learning driven computer vision becomes the annotation of images each time a new application on a new type of data is targeted for developing. A way to reduce this time consuming step is using synthetic data generated with their associated ground truth. This approach has been extensively used in plant phenotyping with 2D images [29–32, 32–36].

Virtual plant modeling have been used in agricultural and plant sciences to simulate plant behaviour and analyze interactions of the plants with their environment [37–39]. There exists many platforms for constructing virtual plant models such as L+C modelling language [40, 41] and L-Py framework [42], which are based on the formalism of L-systems [24]. Despite the availability of such platforms capable of generating synthetic plants with complex architectures, employing them as 3D training data in the form of point clouds for plant phenotyping is only considered very recently [23].

While 2D image-based processing proves to provide high-throughput phenotyping, advances in 3D data acquisition and modeling enable precise estimation of traits through full, occlusion-free 3D geometric information of plants [43, 44]. In most of the previous studies, local surface features were extracted to segment 3D point clouds acquired from plants. Such features included eigenvalues of local covariance matrix [45] or the second tensor [4], Fast Point Feature Histograms (FPFH) [6, 8, 46, 47], and surface curvature [7]. There are also methods that operate on volumetric 3D data of plants. Extraction of eigenvalues of the second-moments tensor of the 3D neighbourhood [48], a breath-first flood-fill algorithm with a 26-connected neighbourhood [10], extraction of multi-scale texture and edge features [27] are examples to volumetric approaches. In some studies, supervised learning techniques such as Support Vector Machines (SVM) [8, 45, 47] and Random Forests [27] were applied to classify 3D plant points to semantic classes. Markov Random Fields (MRF)-based smoothing [7, 47] over class labels or region growing [8, 46] were occasionally used to ensure consistency of point labels in local regions.

Apart from segmentation methods based on local features, graph-based approaches involving spectral embedding and clustering [9, 49] were also proposed. Another strategy is fitting geometric primitives such as ellipses, tubular structures, cylinders or rings to 3D data for semantic segmentation [3, 5, 50, 51].

Research on deep learning methods that directly consume 3D points clouds as input data exploded since the publication of the pioneering work of Qi et al. [19], introducing the PointNet. Guo et al. [20] provide a recent and comprehensive review on deep learning for point clouds. For semantic part segmentation application only, Guo et al. [20] compare 30 point-based architectures that have been developed since 2017. It is beyond the scope of this paper to mention all these architectures here. The benchmarks with which these architectures are commonly tested are data sets including indoor scenes (S3DIS [52], ScanNet [53]) or outdoor urban scenes (Semantic3D [54], Semantic KITTI [55, 56]).

Despite the fast progress in research on point-based 3D deep learning techniques, their application on plant sciences and agriculture is limited to very few studies. For example, Wu et al. [57] modified the PointNet architecture for separating foliage and woody components in terrestrial laser scanning data. In [58], PointNet was used to estimate the proper grasping pose of apples for autonomous harvesting. In some studies for organ segmentation of 3D plant models, Convolutional Neural Networks (CNN) were applied to 2D multi-view images and the inferences were back-projected to 3D for post-processing [59, 60]. In [61] a voxel-based convolutional neural network (VCNN) was designed for maize stem and leaf classification and segmentation. The point clouds were converted to volumetric models before processing. The authors briefly compared their method to PointNet and PointNet++ in terms of segmentation accuracy . To the best of our knowledge, this is the only work where the authors reported part segmentation results on 3D plant models using point-based deep learning architectures.

In order to address the scarcity of application of deep learning-based approaches on 3D plant phenotyping, this work provides a detailed analysis of performances of various 3D point-based architectures on the segmentation of 3D plant models.

3 Material and Method

We adapt and test six recent point-based deep learning segmentation methods for the specific problem of segmentation of rosebush plants to their structural parts and discuss their strengths and shortcomings. The methods we considered in this work are: 1) PointNet [19], 2) PointNet++ [62], 3) Dynamig Graph CNN (DGCNN) [63], 4) PointCNN [64], 5) ShellNet [65], and 6) RIConv [66]. We employed the recently introduced ROSE-X data set, which includes eleven 3D models of real rosebush models obtained through X-ray imaging [27] to train and evaluate the networks. The data set also contains the ground truth labels; each point in the plant shoot is categorized one of the three classes: 1) Flower, 2) Leaf, and 3) Stem (branches and petioles).

In order to explore the contribution of using synthetic data for training, we created a data set consisting of 48 synthetic rosebush (Rosa x hybrida) models. The models were generated by a simulator developed by Favre et al. [67]. The simulator was implemented with L-Studio software [41] based on L-systems. The networks trained by the synthetic data set were then re-trained using transfer learning [68] with some models from the ROSE-X data set. The results were compared with those of the networks trained only with real rosebush models.

3.1 Data sets

In this study, we utilized two sets of 3D models of rosebush plants. The first set is the ROSE-X data set, which is composed of 11 fully annotated 3D models of real rosebush plants acquired through X-ray scanning. The second is the set of synthetic rosebush models which were generated using the L-studio-based simulator developed by Favre et al. [67]. The details of the data sets are provided in the following subsections. The ROSE-X data set is open to public use at [69].

3.1.1 ROSE-X data set

The models in the ROSE-X data set were acquired from real rosebush plants using 3D Siemens X-ray imaging system. The voxel data was annotated fully with manual supervision and then converted to 3D point clouds. The details of the procedure for annotation and the data structure can be found in [27]. Each point in a model belongs to one of three organ classes: Leaf, stem, and flower. The petioles between leaflets were also labeled as stem, since they have branch-like structures and their inclusion to the architecture of branches is important for further analysis. We reserved 3 models for training the networks and used the rest of the 8 models for testing. The distribution of points to the three classes for these models is given in Table 1.

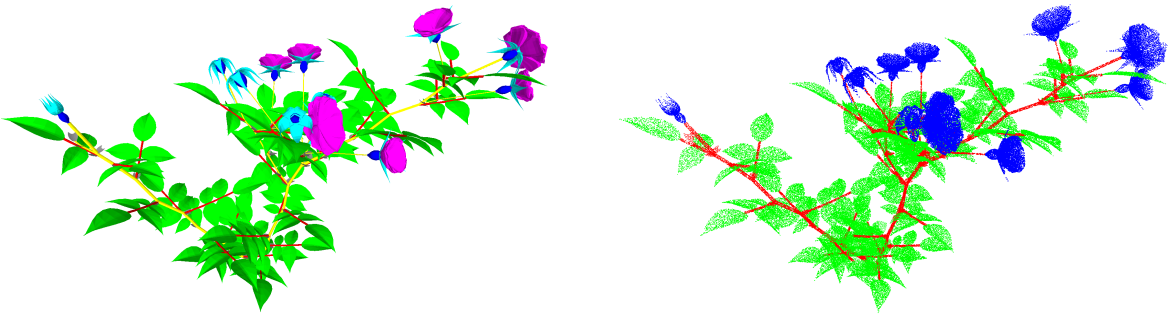
Table 1: Distribution of classes in 3D rosebush point clouds (%).

	Leaf	Stem	Flower
ROSE-X training models	84.56	10.63	4.81
ROSE-X test models	81.00	11.51	7.49
Synthetic models for training	65.80	17.95	16.25
Synthetic models for validation	66.84	17.14	16.02

3.1.2 Synthetic Rosebush Models

To create synthetic rosebush (Rosa x hybrida) models, we used a simulation procedure developed by Favre et al. [67]. The procedure is implemented with the L-studio software [41], which is based on L-systems [24, 25]. It simulates plant growth in 3D in stress-free environments. The simulation model uses the rules of growing found in the literature and observations from real plants. Morphometric measurements (i.e. diameter and length of organs), architectural structures (i.e. leaf formation order) and physiological data were taken from real plants, analyzed and integrated into the model.

Using this simulation procedure, we generated 48 different rosebush models in the triangle mesh structure. The triangle mesh model and point cloud model of some rosebush are given in Fig. 1a. Each triangle in a model is inherently classified into one of seven organs: Leaflet, petiole, stem, stipule, petal, sepal, and receptacle. Since the ROSE-X labels are not as fine-grained, the petiole, stem and stipule classes were merged together to form the stem class and the sepal, petal and receptacle classes were merged into the flower class.



(a) Synthetic plant as a triangle mesh model (b) Sampled point cloud

Figure 1: Synthetic plant as a triangle mesh model and the corresponding sampled point cloud

In order to generate point clouds from these triangle mesh models, we homogeneously sampled points from the triangular surfaces. A point cloud is a set of 3D points $\mathcal{P} = \{p_1, p_2, ..., p_N\}$, where each point $p_i \in \mathcal{P}$ is represented with the point's coordinates (x, y, z) in the 3D space. N is the number of points in the $\mathcal{P}$, which defines the size of the point cloud. The sampling rate was set to 120 points per square unit resulting in point clouds of size of 150,000 to 300,000 points. The point cloud sampled on the model surface is given in Fig. 1b. The dimensions of synthetic models in $x-$, $y-$ and $z-$ axes are in the range of 30 to 50 cm, in accordance to the scale of the real rosebush models.

From the 48 point clouds of synthetic rosebush models, 40 are used for training, while 8 models are used for validation.

3.2 Data preprocessing

The point-based deep learning architectures accept fixed-size data as input. Feeding the entire rosebush model to the networks requires a large sub-sampling rate resulting in a significant loss of geometric information. Therefore, we follow the strategy utilized by the point-based deep learning methods to deal with large-scale point clouds [19]: We partition the rosebush point cloud into blocks of fixed-size. The points in a block are gathered as a separate point cloud. These sub-point clouds are then used to train the networks. A new test point cloud is processed in the same way. It is divided into blocks and the point cloud in each block is separately segmented by the neural network. Then, the predictions from the blocks are combined to obtain a full segmentation.

The points in a block should be sampled such that each block includes a fixed number N of points (N is 4096 for the architectures used in this study). If there are less than 10% of N points in the block, points from neighbouring blocks are added to the point set. Then, the distribution of data in each block is analyzed. For this purpose, the grid size parameter is introduced. Each block is divided into voxels with this parameter and the average of the number of points in the voxels is calculated. For voxels that have points below the average value, the number of points they contain is

increased by duplications. Finally, if the points in the block are higher then the allowed number of points, they are separated so as not to exceed the allowed number. In other words, new blocks that define the same region with different points are obtained.

Data preprocessing step is applied for the training and test data set. To enrich the training data for synthetic models, blocks were obtained with two different offset parameters. Using the same size blocks, new blocks containing different data were obtained by partitioning the full point cloud into blocks with two different offset values. Two sets of blocks from each model were created in this way, providing additional input training data for the networks.

During prediction of a new test point cloud, two offset values were used during block partitioning and the two versions are fed into the network. As a result, for each point in the point cloud, two sets of probability scores for the part classes were obtained. The class with the highest probability score is assigned to the point.

3.3 3D point-based deep learning architectures

We applied six different 3D point-based deep learning architectures to the problem of part segmentation of rosebush models: 1) PointNet [19], 2) PointNet++ [62], 3) Dynamig Graph CNN (DGCNN) [63], 4) PointCNN [64], 5) ShellNet [65], and 6) RIConv [66]. In the following subsections, we briefly describe the key approaches of these architectures to the problem of encoding local geometric structure of 3D point clouds. For details, please refer to the original articles.

3.3.1 PointNet

PointNet architecture [19] is the first deep neural network architecture that directly accepts a point cloud as input. It uplifts the (x, y, z) coordinates of each 3D point separately to high-dimensional features through Multilayer Perceptrons (MLP) with shared weights. A single maximum pooling operation is applied to summarize all the point features into a single global feature vector. This feature vector is concatenated to individual point-based features to be processed by successive layers. Weight-shared MLP layers are applied to the concatenated features to extract the class scores for each point. The PointNet architecture for segmentation is given in Fig. 2.

The predictions produced by PointNet heavily depend on the locations of the points in an isolated manner. There are no connections in the architecture to relate points in close proximity to each other in Euclidean space.

3.3.2 PointNet++

PointNet++ architecture [62] was devised to summarize point-based features in different local scales instead of on a global level. The input point cloud is partitioned into overlapping local regions, and the original PointNet was applied to these regions resulting in feature vectors capturing geometric details of local neighbourhoods. Grouping and feature extraction are applied in a hierarchical manner.

PointNet++ architecture consists of two types of layers: 1) Set abstraction layer (SA) and 2) Feature propagation layer (FP). SA layer consists of two phases: sampling and grouping. In the sampling phase, representative points were found using farthest point sampling algorithm. Local overlapping groups are then formed with the radius neighbourhood around each representative point. PointNet is applied individually to the groups to extract features summarized over all the points in the group. FP layers are responsible to propagate the group-based feature vectors to the original points in the input point cloud. The propagation of features to a point is performed via interpolation from the features of its closest neighbours.

By combining the interpolated and existing features of SA phase, PointNet architecture is used to update the features of each point. In Fig. 3, the PointNet++ architecture for segmentation is given. The R parameter indicates the radius of the local regions, while F is the number of sampled points within each local region.

3.3.3 DGCNN

Dynamic Graph CNN (DGCNN) architecture [63] was designed to integrate a convolution process of 3D points directly into the network, rather than a separate grouping process as done in PointNet++. The local neighbourhood of a point is represented with a graph structure. A process called EdgeConv is applied to extract edge features to encode the spatial relationship between a point and its K neighbours.

Unlike the CNN structures used in regular grids, fixed graphs are not used. The graphs are updated since the K nearest neighborhoods of the points changes in each layer. Only in the first layer, geometrical proximity between nearest points are considered. In the following layers, edge convolution is performed between nearest neighbours that are close in feature space. That might be an advantage in terms of diffusing the information with respect to the proximity in the feature space; however, a multi-scale hierarchical local spatial grouping is not present in DGCNN. The local

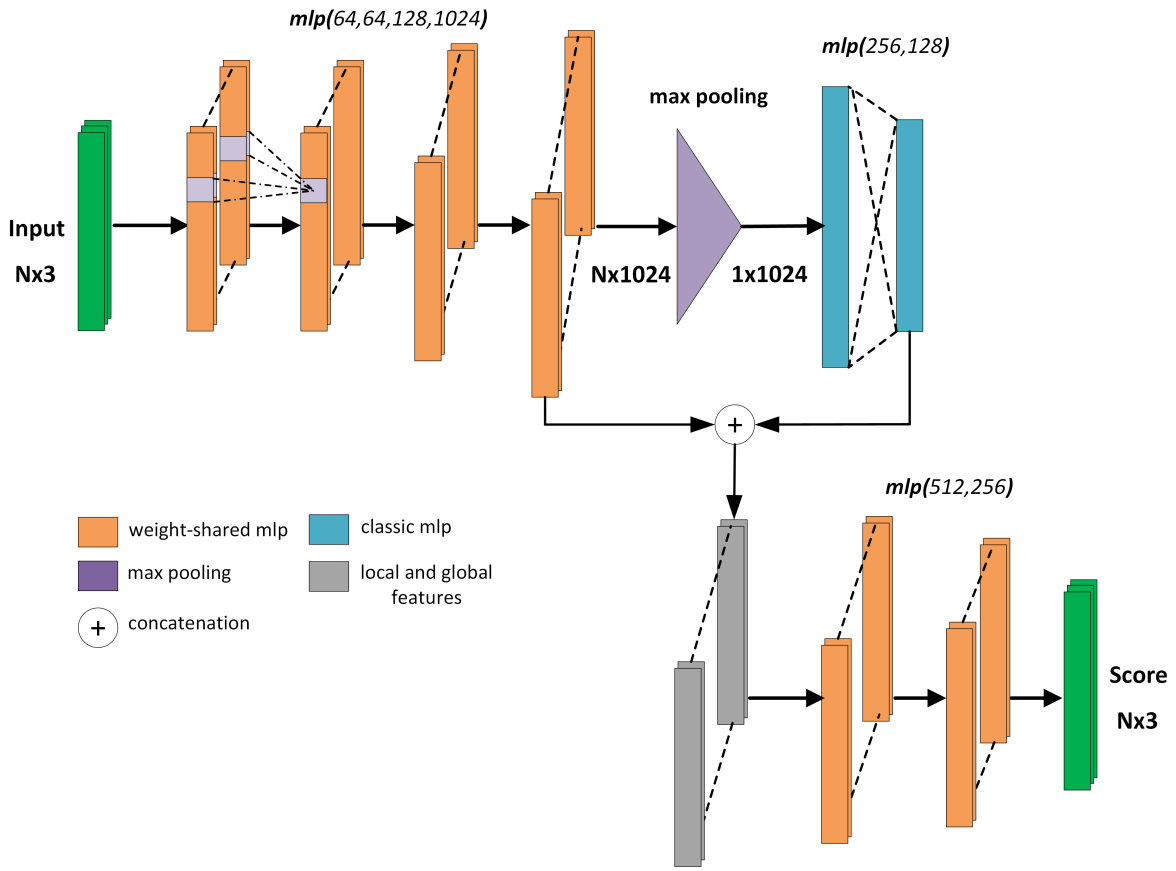


Figure 2: PointNet architecture;

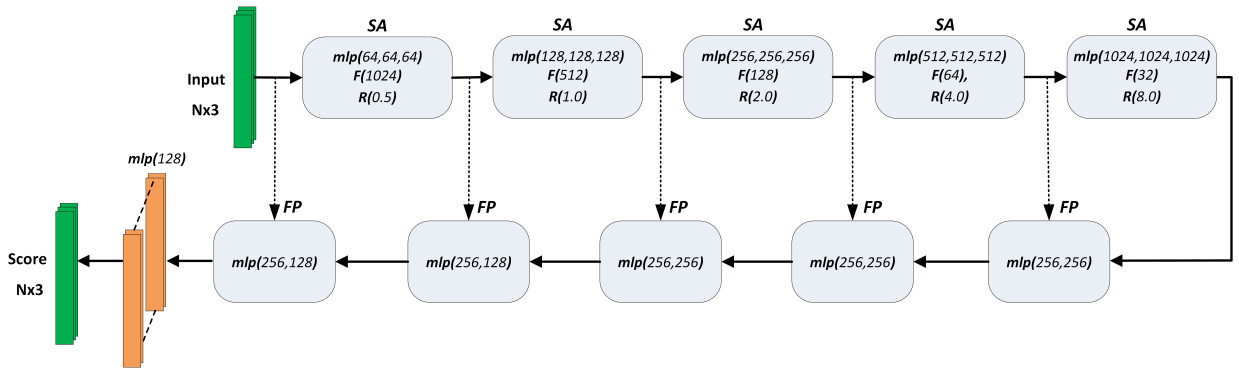


Figure 3: PointNet++ architecture.

geometric structure is only captured at a very localized level; i.e. only within the nearest neighbours of a point. The full architecture of DGCNN for segmentation is given in Fig. 4. K indicates the number of nearest neighbours in spatial or feature space.

3.3.4 PointCNN

A convolution operator that weights the features of the neighbours of a point has been introduced with PointCNN architecture [64]. In this convolution process defined as X-Conv, a KxK-sized transformation matrix is predicted for K adjacent points with multi-layer perceptrons. Typical convolution layers are then applied to the transformed features. To apply convolution to receptive fields of higher size, representative points are generated by farthest point sampling,

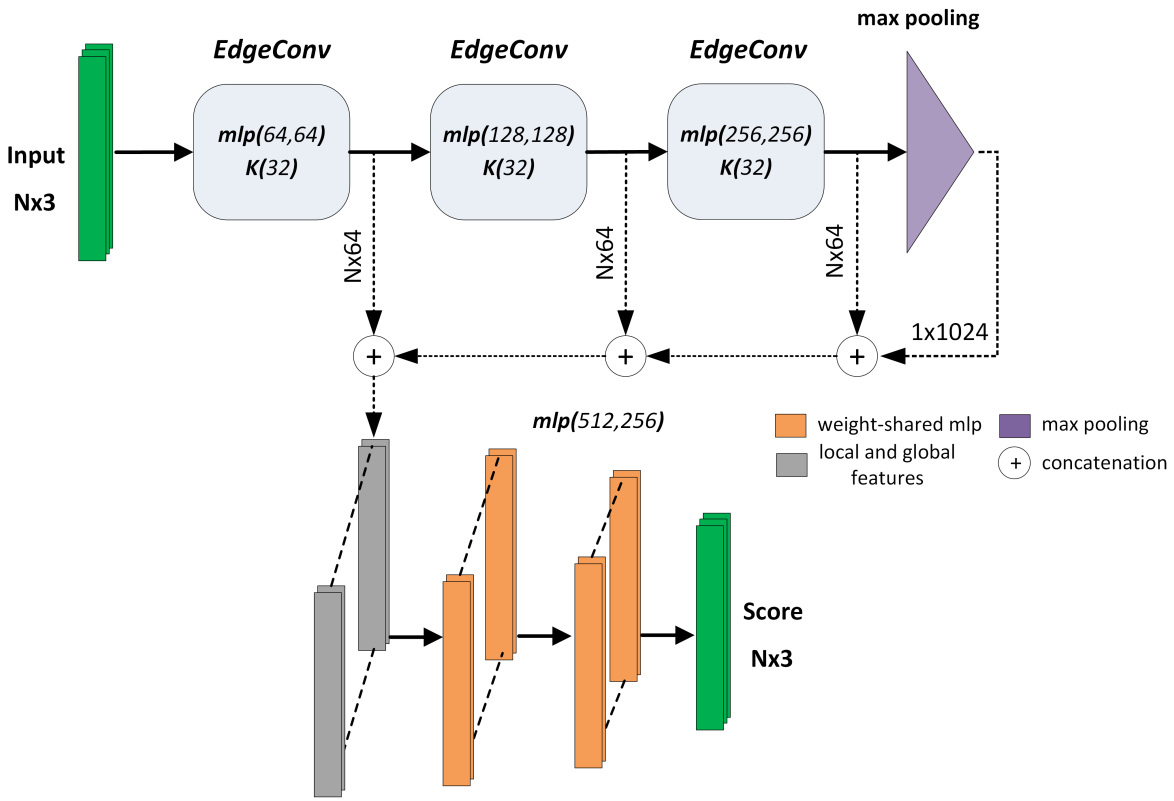


Figure 4: DGCNN architecture.

and features resulting from X-conv are aggregated onto these representative points. By dilating points by a factor and hierarchically applying X-conv, features are aggregated into fewer points, representing larger spatial areas.

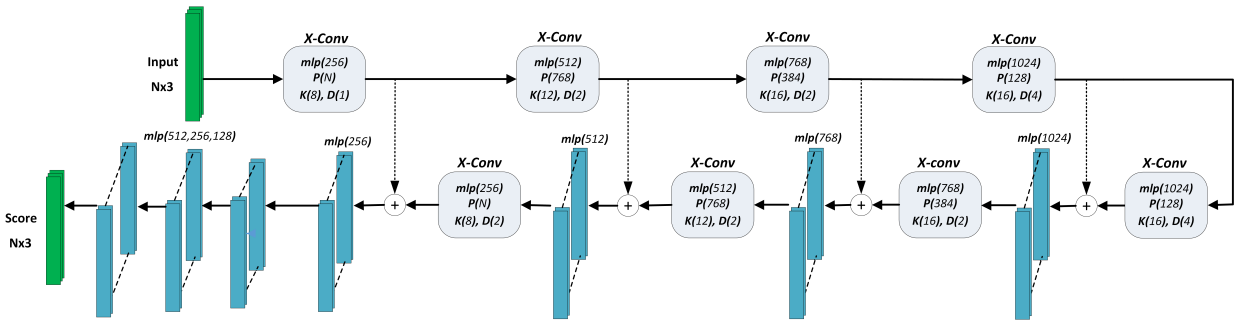


Figure 5: PointCNN architecture.

In Fig. 5, the PointCNN architecture is shown. In this segmentation architecture, point-based outputs are obtained by using an encoder-decoder structure. K corresponds to the number of nearest neighbours that are used in convolution. F indicates the number of sampled points, and d is the point dilation rate.

3.3.5 ShellNet

The ShellConv convolution operator, introduced with the ShellNet architecture [65], is applied to areas within the concentric shells of the local neighbourhood of a point. The size of the sphere is grown until fixed number of points are included in each shell. Descriptive features are extracted for each shell using statistical information of the points. Since a sequence of convolution was defined outwards from starting the inner shell, it became relatively independent from the ordering of the points. The only thing that is not clear is the order of the points in the shell. For this, maximum pooling is used to summarize the features of the points in each shell.

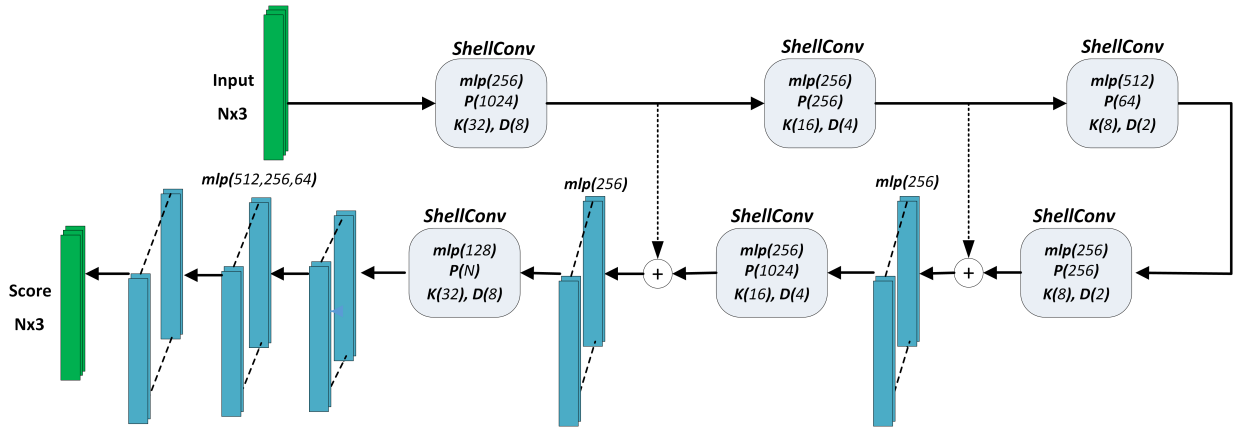


Figure 6: ShellNet architecture.

This process is applied hierarchically by sub-sampling the points to representative points, thus operating on larger receptive fields. The full ShellNet architecture for segmentation is given in Fig. 6. K corresponds to the number of nearest neighbours that are used in convolution. P indicates the number of sampled points, and D is the number of shells.

3.3.6 RIConv

Many 3D deep learning architectures rely on the raw 3D coordinates of the input points, hence are inherently dependent on pose variations of objects in the scene. To provide some form of rotation-invariance, data augmentation with rotated versions of the point clouds is applied. However, the networks cannot model unseen rotations. To ensure rotation invariance, a new convolution process called RIConv [66] is proposed. The main idea is to define the convolution process on rotation-invariant features such as angle and distance between points, rather than the raw 3D coordinates. The learned model is effective against transformations such as translation and rotation in 6-axis space. A simple binning approach for the point permutation problem is integrated into the feature extraction process. The disadvantage of aggregating distances and angles is the loss of geometric data; since two different constellations of 3D points can result in the same rotation-invariant features.

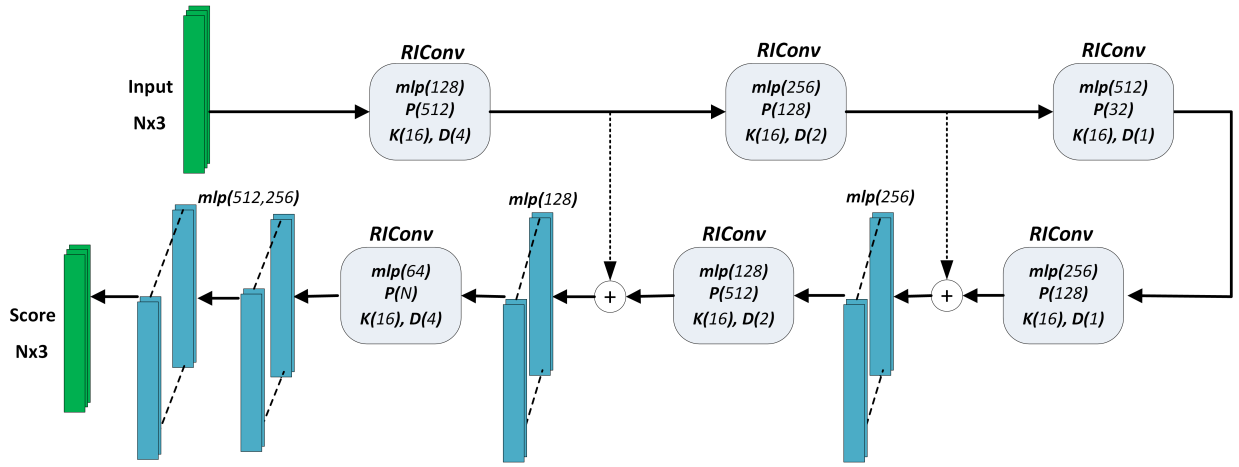


Figure 7: RIConv architecture.

The encoder-decoder architectural structure of RIConv for segmentation is given in Fig. 7. K corresponds to the number of nearest neighbours that are used in convolution. P indicates the number of sampled points, and D is the number of shells.

4 Results

In this paper, we used recall (Re), precision (Pr) and Intersection over Union (IoU) to evaluate the success of each method for separating flower, leaf, and branch parts of the models. We denote the number of true positives, false positives and false negatives for each class as TP_C, FP_C, and FN_C, respectively, where $C \in \{Flower, Leaf, Stem\}$ is the class of the structural part of a rosebush. Recall (Re), precision (Pr) and Intersection over Union (IoU) are defined as

$$Re = \frac{TP_C}{TP_C + FN_C} \tag{1}$$

$$Pr = \frac{TP_C}{TP_C + FP_C} \tag{2}$$

$$IoU = \frac{TP_C}{TP_C + FN_C + FP_C} \ . \tag{3}$$

We also use the mean of the IoU scores over all three classes ($MIoU$) and the total accuracy (Acc). Acc is defined as the ratio of all correctly classified points to the total number of points in the model.

Using the synthetic data generated by L-studio and the real rosebush models from ROSE-X data set we conducted seven types of experiments with each point-based deep learning algorithm:

- **Single real rosebush model for training (I):** We used a single point cloud from the ROSE-X data set of real rosebush models for training the networks. Blocks were extracted from the point cloud to provide training data (96 blocks in total). 20% of the blocks were used as validation set. The corresponding networks trained using one real rosebush plant are called as **I**-trained networks.
- **Two real rosebush models for training (II):** In this experiment, blocks extracted from two real rosebush point clouds (159 blocks in total) are used for training data, where 20% of the blocks are reserved for validation. The corresponding networks trained using two real rosebush plant are called as **II**-trained networks.
- **Three real rosebush models for training (III):** In this experiment, blocks extracted from three real rosebush point clouds (251 blocks in total) are used for training data, where 20% of the blocks are reserved for validation. The corresponding networks trained using three real rosebush plant are called as **III**-trained networks.
- **Synthetic data for training (S):** 40 of the 48 of the synthetic models generated by L-studio was used as training data. 8 models were reserved for validation. Using the results on the validation models, the parameters of each method were optimized. The corresponding trained networks will be denoted as **S**-trained networks.
- **S-trained networks updated with single real rosebush model (S+I):** The S-networks, which were trained and optimized with synthetic data, are re-trained using the blocks extracted from single real rosebush model. The re-training was performed with fixing the network parameters of initial layers and updating the parameters of the last MLP layers with the new training data. We call these update networks **S+I**-trained networks.
- **S-trained networks updated with two real rosebush models (S+II):** In this experiment, the S-networks were re-trained using the blocks extracted from two real rosebush model. We call these update networks **S+II**-trained networks.
- **S-trained networks updated with three real rosebush models (S+III):** In this experiment, the S-networks were re-trained using the blocks extracted from three real rosebush model. We call these update networks **S+III**-trained networks.

The hyper-parameters used to train the networks are given in Table 2.

Table 2: Hyper-parameters used to train the networks.

	Learning rate	**Batch**	**Decay step/rate**	**Weight decay**
PointNet	0.001	48	30000/0.8	None
PointNet++	0.005	12	200000/0.7	None
DGCNN	0.005	12	200000/0.5	None
PointCNN	0.005	8	10000/0.8	1e-8
ShellNet	0.005	12	5000/0.8	1e-8
RIConv	0.005	12	10000/0.8	1e-8

First we examine the results of the **S**-trained networks applied on the 8 synthetic validation models (Table 3). PointNet++, DGCNN, ShellNet and PointCNN were able to produce performance success over 90% for all measures. For the

synthetic models, the local geometric variations between the points in the same class are limited to the generation rules of the L-studio software. Hence, the networks were easily able to learn underlying structures separating one class from the other. PointNet, due to its lack of local feature extraction, could not encode the geometry of neighbourhoods of the points; the $MIoU$ for PointNet remained below 60%.

Table 3: Segmentation results on the validation set of the 8 synthetic rosebush models. 40 synthetic rosebush models were used to train the networks.

		PointNet	**PointNet++**	**DGCNN**	**ShellNet**	**PointCNN**	**RIConv**
	Flower	64.85	99.48	99.21	99.64	99.71	97.29
Re	Leaf	95,28	98.43	99.59	99.54	99.56	97.98
	Stem	45.16	97.82	97.36	96.87	98.23	78.71
	Flower	88.19	99.78	99.68	99.59	99.59	98.57
Pr	Leaf	81.42	99.39	99.28	99.20	99.56	94.28
	Stem	77.39	94.04	98.15	98.22	98.32	91.60
	Flower	59.67	99.26	98.90	99.24	99.30	95.94
IoU	Leaf	78.27	97.84	98.87	98.74	99.13	92.48
	Stem	39.90	92.11	95.60	95.19	96.61	73.41
$MIoU$		59.28	96.40	97.79	97.73	98.35	87.28
Acc		81.82	98.50	99.15	99.10	99.36	94.57

Same behaviour for PointNet is observable in Table 4, where segmentation performance on 8 real rosebush models are presented. Columns in Table 4 correspond to the results of seven types of experiments, as explained before. Despite the increase in the training data and the incorporation of synthetic data, the segmentation performance of PointNet is low, especially for the flower and stem parts. The distinguishing local geometric characteristics of leaf and stems were not captured by PointNet.

Table 4: Segmentation results on 8 real rosebush models from ROSE-X data set with PointNet.

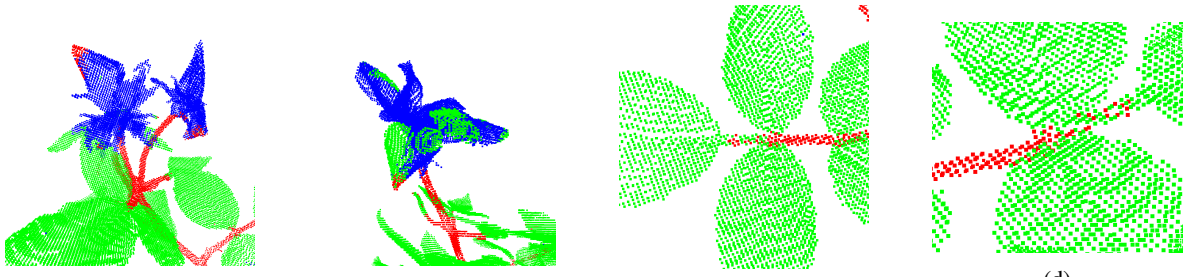
PointNet		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	10.50	11.92	19.18	20.44	14.79	13.85	8.45
Re	Leaf	94.77	97.77	97.55	96.52	90.43	94.63	96.18
	Stem	5.47	2.30	3.15	2.20	9.21	7.25	8.62
	Flower	27,43	34.67	40.35	41.61	31.48	34.95	41.24
Pr	Leaf	81.77	82.05	82.62	82.52	82.18	82.52	82.06
	Stem	19.33	29.13	45.07	15.9	14.41	20.16	28.07
	Flower	8.22	9.73	14.94	15.88	11.19	11.01	7.54
IoU	Leaf	78.23	80.54	80.94	80.14	75.60	78.83	79.48
	Stem	4.46	2.18	3.03	1.99	5.95	5.63	7.06
$MIoU$		30.30	30.82	32.97	32.66	30.92	31.83	31.36
Acc		78.17	80.35	80.81	79.96	75.41	78.52	7.53

Table 5: Segmentation results on 8 real rosebush models from ROSE-X data set with PointNet++.

PointNet++		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	69.98	68.79	81.24	97.02	75.13	73.70	85.39
Re	Leaf	98.02	98.28	97.67	28.72	98.10	98.71	98.71
	Stem	78.73	85.75	87.55	48.81	84.57	87.73	89.03
	Flower	87.80	95.10	87.30	10.60	84.26	91.61	91.57
Pr	Leaf	95.42	96.12	97.27	96.01	96.39	96.55	97.58
	Stem	83.73	83.89	86.20	78.09	89.47	90.48	92.52
	Flower	63.78	66.44	72.72	10.57	65.88	69.04	79.17
IoU	Leaf	93.62	94.53	95.07	28.38	94.62	95.35	96.36
	Stem	68.29	73.63	76.79	42.93	76.91	80.32	83.05
$MIoU$		75.23	78.20	81.53	27.29	79.14	81.57	86.19
Acc		93.70	94.63	95.28	36.15	94.82	95.58	96.60

The segmentation results of 8 test real rosebush models yielded by PointNet++ with seven experimental setups are given in Table 5. The results corresponds to the evaluation values averaged over 8 models. The increase of the training data from a single rosebush model to two and then three models led to an increase in the performance, especially for the stem class. The use of synthetic data alone for training was not effective; however when the network trained with synthetic data was updated with real rosebush models the performance was improved. The results with PointNet++ are

promising with an accuracy rate over 95% and a mean IoU rate over 85%. The main sources of errors are the confusion between stems and thick parts of flowers (Fig. 8a), between leaves and petals of flowers (Fig. 8b), and between petioles and leaves (Fig. 8c, 8d).



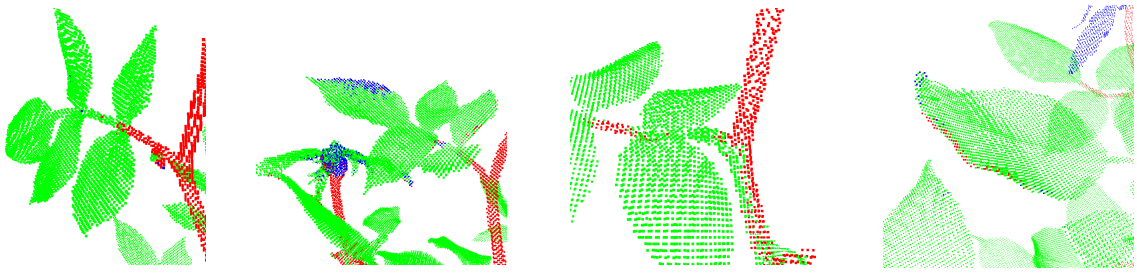
(a) (b) (c) (d)

Figure 8: Examples to erroneous segmentation results produced by PointNet++ (S+III).

The effect of using synthetic data on the segmentation results is even more pronounced for DGCNN (Table 6), PointCNN (Table 7), and ShellNet (Table 8). Rather than training a network with real data from scratch (as in the cases of **I**, **II**, and **III**), using the real data to fine-tune a network trained by synthetic data (as in the cases of **S+I**, **S+II**, and **S+III**) boosts the performance, especially for the stem and flower classes.

Table 6: Segmentation results on 8 real rosebush models from ROSE-X data set with DGCNN.

DGCNN		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	5.62	29.85	7.12	79.65	24.36	34.83	59.16
Re	Leaf	97.85	95.93	98.58	84.27	95.91	97.45	98.20
	Stem	9.68	28.59	27.94	49.05	46.81	73.44	67.35
	Flower	22.67	54.80	75.96	42.46	37.51	75.46	79.71
Pr	Leaf	82.34	86.80	84.60	92.26	88.05	92.71	93.27
	Stem	58.97	51.36	65.42	47.21	77.97	74.16	84.65
	Flower	4.72	23.95	6.96	38.30	17.33	31.28	51.42
IoU	Leaf	80.88	83.72	83.57	78.71	84.87	90.51	91.71
	Stem	9.07	22.50	24.34	31.68	41.35	58.48	60.02
$MIoU$		31.56	43.39	38.29	49.56	47.85	60.09	67.72
Acc		80.79	83.23	83.59	79.87	84.90	90.00	91.73



(a) (b) (c) (d)

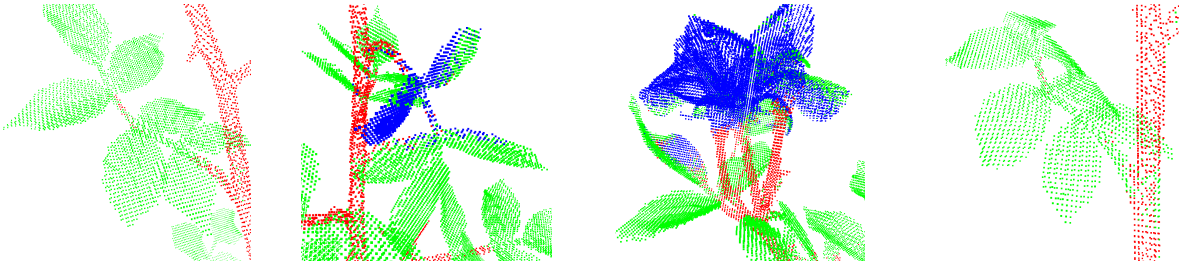
Figure 9: Examples to erroneous segmentation results produced by DGCNN (S+III).

In Fig. 12, we visualized the segmentation results on a sample real rosebush model obtained with **III**-trained networks of the methods; i.e. only three real rosebush models were used for training. We can observe that for DGCNN parts of main stems were classified as leaves and the flower class is not retrieved at all (27.94% and 7.12% recall rate for the flower and stem classes in Table 6). We conjecture that DGCNN is only encoding the geometric structure at the very local level; the spatial receptive field was limited to the K-neighbours of each point in 3D. The data imbalance in the training data in favor of leaves limited the capacity of DGCNN to learn features from stem and flower regions (Fig. 12d). The effect of data imbalance was alleviated with incorporating synthetic data in training data as seen in Fig. 13, where results obtained with **S+III**-trained networks are given. DGCNN was able to capture branch and flower structures (Fig. 13e).

Despite the incorporation of synthetic data, DGCNN's performance lacks behind PointNet++, PointCNN, and ShellNet. These three architectures, in contrast to DGCNN, have the capacity to increase the size of the receptive fields through successive spatial re-grouping. Examples to erroneous segmentation results produced by DGCNN are visualized in Fig. 9. Classifying petioles as leaves (Fig. 9a) is a common error for all architectures, however it occurs more frequently with DGCNN. Confusion between leaves and flowers are present (Fig. 9b). Surfaces of main stems can be classified as leaf points (Fig. 9c). In some cases boundaries of leaves are assigned to the stem class (Fig. 9d).

Table 7: Segmentation results on 8 real rosebush models from ROSE-X data set with PointCNN.

PointCNN		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	5.40	23.49	52.55	97.04	72.90	75.52	68.21
Re	Leaf	99.18	99.00	99.18	59.04	97.75	97.86	98.41
	Stem	34.06	66.46	72.39	46.53	48.08	72.79	75.77
	Flower	80.53	92.07	90.32	19.94	75.83	79.56	84.35
Pr	Leaf	84.85	89.85	92.87	95.81	91.80	95.06	94.83
	Stem	81.31	86.55	91.21	39.33	84.56	88.14	88.23
	Flower	5.33	23.03	49.76	19.82	59.15	63.25	60.55
IoU	Leaf	84.25	89.04	92.16	57.55	89.90	93.12	93.40
	Stem	31.58	60.24	67.67	27.09	44.20	66.30	68.81
$MIoU$		40.39	57.44	69.86	34.82	64.42	74.22	74.26
Acc		84.65	89.60	92.60	60.45	90.17	93.30	93.54



(a) (b) (c) (d)

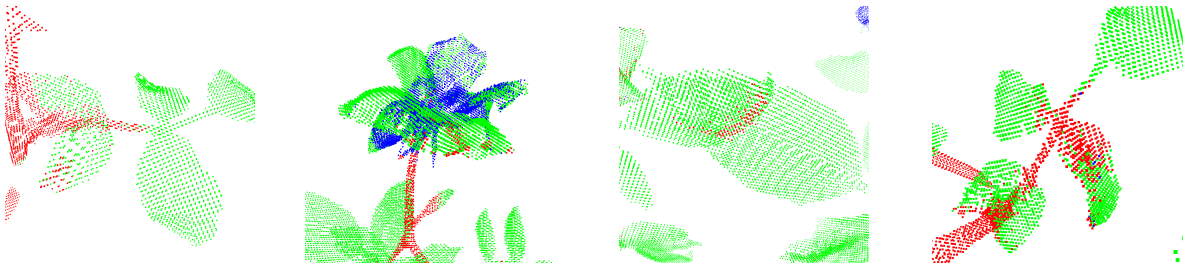
Figure 10: Examples to erroneous segmentation results produced by PointCNN (S+III).

Table 8: Segmentation results on 8 real rosebush models from ROSE-X data set with ShellNet.

ShellNet		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	48.62	44.32	51.54	97.27	68.13	51.12	58.95
Re	Leaf	96.52	97.39	98.09	66.19	95.74	98.62	98.62
	Stem	30.83	54.11	59.87	41.85	58.17	66.79	73.48
	Flower	64.54	82.61	85.05	20.06	74.14	87.63	91.26
Pr	Leaf	87.85	90.35	91.46	94.40	92.59	92.24	93.38
	Stem	66.20	71.80	80.24	70.06	71.54	85.15	87.90
	Flower	38.37	40.54	47.26	19.95	55.05	47.67	55.80
IoU	Leaf	85.16	88.22	89.86	63.69	88.93	91.07	92.18
	Stem	26.64	44.63	52.18	35.50	47.24	59.82	66.73
$MIoU$		50.05	57.80	63.10	39.71	63.74	69.19	71.57
Acc		85.37	88.43	90.20	65.72	89.35	91.40	92.75

The second best results after PointNet++ were obtained with PointCNN (Table 7). Examples to erronous segmentation results produced by PointCNN are shown in Fig. 11. We observe petioles classified as leaves (Fig. 10a and 10d), elongated and thick leaves classified as flowers (Fig. 10b). There is also confusion between leaves and petals (Fig. 10c). In some cases, main stem points close to leaves are classified as leaf points (Fig. 10d).

The quantitative performance results obtained with ShellNet architecture (Table 8) are close to PointCNN. They use similar strategies to group local points; they both recursively sub-sample the point cloud through selecting representative points and aggregate features from the closest neighbours of these representatives. In PointCNN, however, aggregation through convolution is performed through a predicted ordering of all the neighbour points; a property to which we attribute its higher performance compared to ShellNet.



(a) (b) (c) (d)

Figure 11: Examples to erroneous segmentation results produced by ShellNet (S+III).

With ShellNet, as with the other architectures, petioles (Fig. 11a) and petals (Fig. 11b) were occasionally confused with leaf points. Touching leaves resulting in thick structures are also a cause of error (Fig. 11c). Another source of error with ShellNet is the interference of points from close parts, such as the misclassification of leaf points as stems in Fig. 11d.

The segmentation results obtained with RIConv (Table 9) falls behind all the architectures except PointNet. The local regions were extracted in the same way as in ShellNet, however, use of rotation invariant features resulted in significant loss of geometric information about the constellation of the points, which is especially important in distinguishing plant parts.

Table 9: Segmentation results on 8 real rosebush models from ROSE-X data set with RIConv.

RIConv		**I**	**II**	**III**	**S**	**S + I**	**S + II**	**S + III**
	Flower	45.99	40.40	56.09	81.47	38.16	61.00	55.47
Re	Leaf	98.46	98.36	98.92	72.85	99.27	98.26	98.48
	Stem	31.53	38.09	37.88	38.58	40.62	55.67	57.88
	Flower	85.76	88.46	91.81	21.21	90.16	89.61	92.04
Pr	Leaf	87.49	87.82	89.08	90.17	87.87	91.45	91.33
	Stem	75.03	74.86	79.61	76.89	87.69	81.41	81.78
	Flower	42.72	38.38	53.42	20.24	36.63	56.97	52.93
IoU	Leaf	86.31	86.55	88.22	67.49	87.31	90.00	90.07
	Stem	28.54	33.77	34.53	34.58	38.43	49.39	51.27
$MIoU$		52.52	52.90	58.72	40.77	54.12	65.45	64.76
Acc		86.82	87.08	88.68	69.55	87.94	90.57	90.59

Table 10: Segmentation results on test set of 8 real rosebush models for all architectures. The first row corresponds to $MIoU$ results of networks trained with three real rosebush models. The second row gives the $MIoU$ results for the case, where the networks were trained with synthetic models and updated with three real rosebush models.

	PointNet	**PointNet++**	**DGCNN**	**PointCNN**	**ShellNet**	**RIConv**
III	32.97	81.53	38.29	69.86	63.10	58.72
S+III	31.36	86.19	67.72	74.26	71.57	64.76

As a summary, we give the segmentation performances (in $MIoU$) of all architectures with **III**-trained and **S+III**-trained networks in Table 10. The incorporation of synthetic data was significantly beneficial for all architectures except for PointNet. PointNet was unable to learn local geometric structures since all points were processed independent of each other. RIConv performed poorly compared to other architectures due to the information loss with rotation invariant features. DGCNN used a single spatial receptive field at the very local level and opted for feature proximity in a non-local way; therefore missing the multi-scale spatial variability in plant parts.

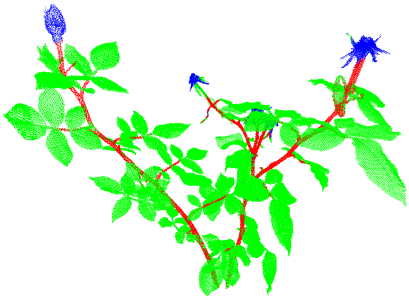
The best results were obtained with PointNet++ with or without the use of synthetic data for training. The hierarchically organized local regions for feature extraction with PointNet++ are defined in terms of metric radius, which are flexible and can be adjusted outside the network structure. The next best two methods are PointCNN and ShellNet, both of which hierarchically regroup points and aggregate features within the network. However, the neighbourhoods are defined with respect to K-neighbourhood of points instead of metric radius. Therefore, it is not straightforward to adjust the size of the receptive fields for these architectures while taking into account both the size of the plant structures and the point density of the point clouds.

(a) Ground Truth.

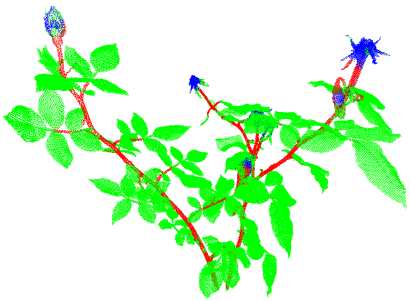
(b) PointNet.

(c) PointNet++.

(d) DgCNN.



(e) PointCNN.



(f) ShellNet.

(g) RIConv.

Figure 12: A real rosebush model segmented with the networks trained with with three real rosebush models (**III**).

(a) Ground Truth.

(b) PointNet.

(c) PointNet++.

(d) DgCNN.

(e) PointCNN.

(f) ShellNet.

(g) RIConv.

Figure 13: A real rosebush model segmented with the networks trained with synthetic models and updated with three real rosebush models (**S+III**).

5 Discussion

For a 3D point-based architecture to operate effectively on plant data, it should consider the multi-scale and self-similar nature of plants. The architecture should be able to handle multiple, hierarchical spatial receptive fields in the network and their sizes should be easily tuned to the scales of various structures in the plants. The multi-scale feature extraction scheme is also necessary to account for the intra-class size variations; such as variations in branch diameter or leaf length and intra-class geometric variations, such as diverse range of curvature on the branches and leaves. The feature extraction process should be invariant to point density, which is typically very heterogeneous for 3D point clouds acquired from plants. Also grouping features with respect to their proximity in the feature space can lead to non-local similarity modeling to capture repetitive structures that are inherent to plants.

Another issue is that the variability of local parts is greatly effected by the intricate plant structure, bringing distinct parts close to each other. The training data should be able to account for diverse local geometric occurrences, such as touching leaves or branches due to dense foliage. More realistic synthetic data or plant-specific augmentation techniques ensuring folding of leaves and branches with a set of rules can help enrich the labeled data.

6 Conclusion

We applied six recent 3D point-based deep learning architectures, PointNet, PointNet++, DGCNN, PointCNN, ShellNet, and RIConv, to segment 3D models of real rosebush plants into their structural parts. We used the annotated 3D models in ROSE-X data set for training and testing the networks. We also conducted experiments where the networks were pre-trained with synthetic rosebush models generated by L-studio software, and then updated by real rosebush data. The results indicate that pre-training with synthetic data boosts the performace of all networks, except PointNet. The best segmentation results were obtained by PointNet++ with a mean IoU rate of 86.19%. We attribute this success to the ease of determining the size of the hierarchical local regions to extract multi-scale features with PointNet++. RIConv was not as effective due to reliance on rotation invariant features providing insufficient local geometric information. DGCNN , PointCNN, and ShellNet produced promisinig results, however defining local regions for feature extraction by K-neighbourhood of points is less practical for modeling plant geometry; since the optimum K for each scale will be dependent on both the size of the plant part structures and the point density of the 3D point cloud.

References

[1] M. Minervini, H. Scharr, S. A. Tsaftaris, Image analysis: The new bottleneck in plant phenotyping [applications corner], IEEE Signal Processing Magazine 32 (2015) 126–131. doi:10.1109/MSP.2015.2405111.

[2] H. Scharr, M. Minervini, A. P. French, C. Klukas, D. M. Kramer, X. Liu, I. Luengo, J.-M. Pape, G. Polder, D. Vukadinovic, X. Yin, S. A. Tsaftaris, Leaf segmentation in plant phenotyping: A collation study, Machine Vision and Applications 27 (2016) 585–606. URL: https://doi.org/10.1007/s00138-015-0737-3. doi:10.1007/s00138-015-0737-3.

[3] A. Paproki, X. Sirault, S. Berry, R. Furbank, J. Fripp, A novel mesh processing based technique for 3D plant analysis, BMC Plant Biology 12 (2012) 63. URL: https://doi.org/10.1186/1471-2229-12-63. doi:10.1186/1471-2229-12-63.

[4] B. Elnashef, S. Filin, R. N. Lati, Tensor-based classification and segmentation of three-dimensional point clouds for organ-level plant phenotyping and growth analysis, Computers and Electronics in Agriculture 156 (2019) 51 – 61. URL: http://www.sciencedirect.com/science/article/pii/S0168169918306008. doi:https://doi.org/10.1016/j.compag.2018.10.036.

[5] W. Gélard, M. Devy, A. Herbulot, P. Burger, Model-based segmentation of 3D point clouds for phenotyping sunflower plants, in: Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP, (VISIGRAPP 2017), 2017, pp. 459–467. doi:10.5220/0006126404590467.

[6] M. Wahabzada, S. Paulus, K. Kersting, A.-K. Mahlein, Automated interpretation of 3D laserscanned point clouds for plant organ segmentation, BMC Bioinformatics 16 (2015) 248. URL: https://doi.org/10.1186/s12859-015-0665-2. doi:10.1186/s12859-015-0665-2.

[7] Y. Li, X. Fan, N. J. Mitra, D. Chamovitz, D. Cohen-Or, B. Chen, Analyzing growing plants from 4D point cloud data, ACM Trans. Graph. 32 (2013) 157:1–157:10. URL: http://doi.acm.org/10.1145/2508363.2508368. doi:10.1145/2508363.2508368.

[8] S. Paulus, J. Dupuis, A.-K. Mahlein, H. Kuhlmann, Surface feature based classification of plant organs from 3D laserscanned point clouds for plant phenotyping, BMC Bioinformatics 14 (2013) 238. URL: https://doi.org/10.1186/1471-2105-14-238.

[9] F. Hétroy-Wheeler, E. Casella, D. Boltcheva, Segmentation of tree seedling point clouds into elementary units, International Journal of Remote Sensing 37 (2016) 2881–2907. URL: https://hal.inria.fr/hal-01285419. doi:10.1080/01431161.2016.1190988.

[10] F. Golbach, G. Kootstra, S. Damjanovic, G. Otten, R. van de Zedde, Validation of plant part measurements using a 3D reconstruction method suitable for high-throughput seedling phenotyping, Machine Vision and Applications 27 (2016) 663–680. URL: https://doi.org/10.1007/s00138-015-0727-5. doi:10.1007/s00138-015-0727-5.

[11] M. P. Pound, A. P. French, J. A. Fozard, E. H. Murchie, T. P. Pridmore, A patch-based approach to 3d plant shoot phenotyping, Machine Vision and Applications 27 (2016) 767–779. URL: https://doi.org/10.1007/s00138-016-0756-8. doi:10.1007/s00138-016-0756-8.

[12] Z. Liu, Q. Zhang, P. Wang, Z. Li, H. Wang, Automated classification of stems and leaves of potted plants based on point cloud data, Biosystems Engineering 200 (2020) 215 – 230. URL: http://www.sciencedirect.com/science/article/pii/S1537511020302658. doi:https://doi.org/10.1016/j.biosystemseng.2020.10.006.

[13] J. Mack, F. Rist, K. Herzog, R. Töpfer, V. Steinhage, Constraint-based automated reconstruction of grape bunches from 3D range data for high-throughput phenotyping, Biosystems Engineering 197 (2020) 285 – 305. URL: http://www.sciencedirect.com/science/article/pii/S1537511020301975. doi:https://doi.org/10.1016/j.biosystemseng.2020.07.004.

[14] J. R. Ubbens, I. Stavness, Deep plant phenomics: A deep learning platform for complex plant phenotyping tasks, Frontiers in Plant Science 8 (2017) 1190. URL: https://www.frontiersin.org/article/10.3389/fpls.2017.01190. doi:10.3389/fpls.2017.01190.

[15] M. P. Pound, J. A. Atkinson, A. J. Townsend, M. H. Wilson, M. Griffiths, A. S. Jackson, A. Bulat, G. Tzimiropoulos, D. M. Wells, E. H. Murchie, T. P. Pridmore, A. P. French, Deep machine learning provides state-of-the-art performance in image-based plant phenotyping, GigaScience 6 (2017). URL: https://doi.org/10.1093/gigascience/gix083. doi:10.1093/gigascience/gix083, gix083.

[16] J. Atanbori, A. P. French, T. P. Pridmore, Towards infield, live plant phenotyping using a reduced-parameter cnn, Machine Vision and Applications 31 (2020). URL: https://nottingham-repository.worktribe.com/output/2611579. doi:10.1007/s00138-019-01051-7.

[17] J. Praveen Kumar, S. Dominic, Rosette plant segmentation with leaf count using orthogonal transform and deep convolutional neural network, Machine Vision and Applications 31 (2020). doi:10.1007/s00138-019-01056-2.

[18] J. Grimm, K. Herzog, F. Rist, A. Kicherer, R. Töpfer, V. Steinhage, An adaptable approach to automated visual detection of plant organs with applications in grapevine breeding, Biosystems Engineering 183 (2019) 170 – 183. URL: http://www.sciencedirect.com/science/article/pii/S1537511018311838. doi:https://doi.org/10.1016/j.biosystemseng.2019.04.018.

[19] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3D classification and segmentation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[20] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, M. Bennamoun, Deep learning for 3D point clouds: A survey, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020) 1–1.

[21] W. Liu, J. Sun, W. Li, T. Hu, P. Wang, Deep learning on point clouds and its application: A survey, Sensors 19 (2019) 4188. doi:doi:10.3390/s19194188.

[22] D. Griffiths, J. Boehm, A review on deep learning techniques for 3D sensed data classification, Remote Sensing 11 (2019) 1499. doi:10.3390/rs11121499.

[23] A. Chaudhury, F. Boudon, C. Godin, 3D Plant Phenotyping: All You Need is Labelled Point Cloud Data, in: CVPPP-ECCV 2020 - Workshop on Computer Vision Problems in Plant Phenotyping, Glasgow, United Kingdom, 2020, pp. 1–17. URL: https://hal.archives-ouvertes.fr/hal-03030004.

[24] A. Lindenmayer, Mathematical models for cellular interaction in development: Parts I and II., Journal of Theoretical Biology 18 (1968) 280–315.

[25] P. Prusinkiewicz, A. Lindenmayer, The Algorithmic Beauty of Plants, Springer-Verlag, Berlin, Heidelberg, 1996.

[26] J. Ubbens, M. Cieslak, P. Prusinkiewicz, I. Stavness, The use of plant models in deep learning: An application to leaf counting in rosette plants, Plant Methods 14 (2018). doi:10.1186/s13007-018-0273-z.

[27] H. Dutagaci, P. Rasti, G. Galopin, D. Rousseau, ROSE-X: An annotated data set for evaluation of 3D plant organ segmentation methods, Plant Methods 16 (2020). doi:10.1186/s13007-020-00573-w.

[28] Y. Jiang, C. Li, Convolutional neural networks for image-based high-throughput plant phenotyping: A review, Plant Phenomics 2020 (2020) 1–22. doi:10.34133/2020/4152816.

[29] R. Barth, J. IJsselmuiden, J. Hemming, E. J. Van Henten, Data synthesis methods for semantic segmentation in agriculture: A Capsicum annuum dataset, Computers and Electronics in Agriculture 144 (2018) 284–296.

[30] M. Di Cicco, C. Potena, G. Grisetti, A. Pretto, Automatic model based dataset generation for fast and accurate crop and weeds detection, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2017, pp. 5188–5195.

[31] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, H. Greenspan, Synthetic data augmentation using gan for improved liver lesion classification, in: 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), IEEE, 2018, pp. 289–293.

[32] M. Valerio Giuffrida, H. Scharr, S. A. Tsaftaris, Arigan: Synthetic arabidopsis plants using generative adversarial network, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 2064–2071.

[33] P. Pawara, E. Okafor, L. Schomaker, M. Wiering, Data augmentation for plant classification, in: International Conference on Advanced Concepts for Intelligent Vision Systems, Springer, 2017, pp. 615–626.

[34] D. Ward, P. Moghadam, N. Hudson, Deep leaf segmentation using synthetic data (2018) 26.

[35] Y. Zhu, M. Aoun, M. Krijn, J. Vanschoren, H. T. Campus, Data augmentation using conditional generative adversarial networks for leaf counting in arabidopsis plants., in: BMVC, 2018, p. 324.

[36] C. Douarre, C. F. Crispim-Junior, A. Gelibert, L. Tougne, D. Rousseau, Novel data augmentation strategies to boost supervised segmentation of plant disease, Computers and Electronics in Agriculture 165 (2019) 104967.

[37] J. Evers, J. Vos, Modeling branching in cereals, Frontiers in Plant Science 4 (2013) 399. URL: https://www.frontiersin.org/article/10.3389/fpls.2013.00399. doi:10.3389/fpls.2013.00399.

[38] G. Buck-Sorlin, Functional-structural plant modeling, in: W. Dubitzky, O. Wolkenhauer, K.-H. Cho, H. Yokota (Eds.), Encyclopedia of Systems Biology, Springer New York, New York, NY, 2013, pp. 778–781.

[39] G. Buck-Sorlin, M. Delaire, Meeting present and future challenges in sustainable horticulture using virtual plants, Frontiers in Plant Science 4 (2013) 443. URL: https://www.frontiersin.org/article/10.3389/fpls.2013.00443. doi:10.3389/fpls.2013.00443.

[40] R. Karwowski, P. Prusinkiewicz, Design and implementation of the L+C modeling language, Electronic Notes in Theoretical Computer Science 86 (2003) 134 – 152. URL: http://www.sciencedirect.com/science/article/pii/S1571066104806807. doi:https://doi.org/10.1016/S1571-0661(04)80680-7, 4th International Workshop on Rule-Based Programming (in connection with RDP'03, Federated Conference on Rewriting, Deduction and Programming).

[41] R. Karwowski, P. Prusinkiewicz, The L-system-based plant-modeling environment L-studio 4.0, in: Proceedings of the 4th International Workshop on Functional-Structural Plant Models, Montpellier, France, 2004, pp. 403–405.

[42] F. Boudon, C. Pradal, T. Cokelaer, P. Prusinkiewicz, C. Godin, L-Py: An L-system simulation framework for modeling plant architecture development based on a dynamic language, Frontiers in Plant Science 3 (2012) 76. URL: https://www.frontiersin.org/article/10.3389/fpls.2012.00076. doi:10.3389/fpls.2012.00076.

[43] S. Paulus, H. Schumann, H. Kuhlmann, J. Léon, High-precision laser scanning system for capturing 3D plant architecture and analysing growth of cereal plants, Biosystems Engineering 121 (2014) 1 – 11. URL: http://www.sciencedirect.com/science/article/pii/S1537511014000166. doi:https://doi.org/10.1016/j.biosystemseng.2014.01.010.

[44] J. A. Gibbs, M. P. Pound, A. P. French, D. M. Wells, E. H. Murchie, T. P. Pridmore, Active vision and surface reconstruction for 3D plant shoot modelling, IEEE/ACM Transactions on Computational Biology and Bioinformatics 17 (2020) 1907–1917. doi:10.1109/TCBB.2019.2896908.

[45] D. Dey, L. Mummert, R. Sukthankar, Classification of plant structures from uncalibrated image sequences, in: 2012 IEEE Workshop on the Applications of Computer Vision (WACV), 2012, pp. 329–336. doi:10.1109/WACV.2012.6163017.

[46] S. Paulus, J. Dupuis, S. Riedel, H. Kuhlmann, Automated analysis of barley organs using 3D laser scanning: An approach for high throughput phenotyping, Sensors 14 (2014) 12670–12686. URL: http://www.mdpi.com/1424-8220/14/7/12670. doi:10.3390/s140712670.

[47] P. Sodhi, S. Vijayarangan, D. Wettergreen, In-field segmentation and identification of plant structures using 3D imaging, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017, pp. 5180–5187. doi:10.1109/IROS.2017.8206407.

[48] M. Klodt, D. Cremers, High-resolution plant shape measurements from multi-view stereo reconstruction, in: L. Agapito, M. M. Bronstein, C. Rother (Eds.), Computer Vision - ECCV 2014 Workshops, Springer International Publishing, Cham, 2015, pp. 174–184.

[49] T. T. Santos, L. V. Koenigkan, J. G. A. Barbedo, G. C. Rodrigues, 3D plant modeling: Localization, mapping and segmentation for plant phenotyping using a single hand-held camera, in: L. Agapito, M. M. Bronstein, C. Rother (Eds.), Computer Vision - ECCV 2014 Workshops, Springer International Publishing, Cham, 2015, pp. 247–263.

[50] J. Binney, G. S. Sukhatme, 3D tree reconstruction from laser range data, in: 2009 IEEE International Conference on Robotics and Automation, 2009, pp. 1321–1326. doi:10.1109/ROBOT.2009.5152684.

[51] S. Chaivivatrakul, L. Tang, M. N. Dailey, A. D. Nakarmi, Automatic morphological trait characterization for corn plants via 3D holographic reconstruction, Computers and Electronics in Agriculture 109 (2014) 109 – 123. URL: http://www.sciencedirect.com/science/article/pii/S0168169914002191. doi:https://doi.org/10.1016/j.compag.2014.09.005.

[52] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3D semantic parsing of large-scale indoor spaces, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1534–1543.

[53] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, ScanNet: Richly-annotated 3D reconstructions of indoor scenes, in: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE, 2017.

[54] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, M. Pollefeys, SEMANTIC3D.NET: A new large-scale point cloud classification benchmark, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, volume IV-1-W1, 2017, pp. 91–98.

[55] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, J. Gall, SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences, in: Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV), 2019.

[56] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The KITTI Vision Benchmark Suite, in: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3354–3361.

[57] B. Wu, G. Zheng, Y. Chen, An improved convolution neural network-based model for classifying foliage and woody components from terrestrial laser scanning data, Remote Sensing 12 (2020) 1010. URL: http://dx.doi.org/10.3390/rs12061010. doi:10.3390/rs12061010.

[58] H. Kang, C. Chen, Real-time fruit recognition and grasping estimation for autonomous apple harvesting, 2020. arXiv:2003.13298.

[59] W. Shi, R. van de Zedde, H. Jiang, G. Kootstra, Plant-part segmentation using deep learning and multi-view vision, Biosystems Engineering 187 (2019) 81 – 95. URL: http://www.sciencedirect.com/science/article/pii/S1537511019308098. doi:https://doi.org/10.1016/j.biosystemseng.2019.08.014.

[60] B. Japes, J. Mack, F. Rist, K. Herzog, R. Töpfer, V. Steinhage, Multi-view semantic labeling of 3D point clouds for automated plant phenotyping, ArXiv abs/1805.03994 (2018).

[61] S. Jin, Y. Su, S. Gao, F. Wu, Q. Ma, K. Xu, Q. Ma, T. Hu, J. Liu, S. Pang, H. Guan, J. Zhang, Q. Guo, Separating the structural components of maize for field phenotyping using terrestrial LiDAR data and deep convolutional neural networks, IEEE Transactions on Geoscience and Remote Sensing 58 (2020) 2644–2658.

[62] C. R. Qi, L. Yi, H. Su, L. J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 5099–5108.

[63] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph CNN for learning on point clouds, ACM Transactions on Graphics 38 (2019). URL: https://doi.org/10.1145/3326362. doi:10.1145/3326362.

[64] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, PointCNN: Convolution on X-transformed points, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31, Curran Associates, Inc., 2018, pp. 820–830. URL: http://papers.nips.cc/paper/7362-pointcnn-convolution-on-x-transformed-points.pdf.

[65] Z. Zhang, B.-S. Hua, S.-K. Yeung, ShellNet: Efficient point cloud convolutional neural networks using concentric shells statistics, in: The IEEE International Conference on Computer Vision (ICCV), 2019.

[66] Z. Zhang, B. Hua, D. W. Rosen, S. Yeung, Rotation invariant convolutions for 3D point clouds deep learning, in: 2019 International Conference on 3D Vision (3DV), 2019, pp. 204–213.

[67] P. Favre, G. Guéritaine, B. Andrieu, R. Boumaza, S. Demotes-Mainard, C. Fournier, G. Galopin, L. Huche-Thelier, P. Morel-Chevillet, V. Guérin, Modelling the architectural growth and development of rosebush using L-Systems, in: Workshop on Growth Phenotyping and Imaging in Plants, Montpellier, France, 2007, p. np. URL: https://hal.archives-ouvertes.fr/hal-01192410.

[68] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, in: Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27, Curran Associates, Inc., 2014, pp. 3320–3328.

[69] The ROSE-X dataset, 2020. URL: https://uabox.univ-angers.fr/index.php/s/rnPm5EHFK6Xym9t.