Article

Sensor Agnostic Semantic Segmentation of Structurally Diverse and Complex Forest Point Clouds Using Deep Learning

Sean Krisanski [1,*] **, Mohammad Sadegh Taskhiri** [1] **, Susana Gonzalez Aracil** [2] **, David Herries** [2] **and Paul Turner** [1]

1   ARC Training Centre for Forest Value, University of Tasmania, Churchill Ave., Hobart, TAS 7005, Australia; mohammadsadegh.taskhiri@utas.edu.au (M.S.T.); paul.turner@utas.edu.au (P.T.)
2   Interpine Group Ltd., Rotorua 3010, New Zealand; susana.gonzalez@interpine.nz (S.G.A.); David.Herries@interpine.nz (D.H.)
*   Correspondence: sean.krisanski@utas.edu.au

Abstract: Forest inventories play an important role in enabling informed decisions to be made for the management and conservation of forest resources; however, the process of collecting inventory information is laborious. Despite advancements in mapping technologies allowing forests to be digitized in finer granularity than ever before, it is still common for forest measurements to be collected using simple tools such as calipers, measuring tapes, and hypsometers. Dense understory vegetation and complex forest structures can present substantial challenges to point cloud processing tools, often leading to erroneous measurements, and making them of less utility in complex forests. To address this challenge, this research demonstrates an effective deep learning approach for semantically segmenting high-resolution forest point clouds from multiple different sensing systems in diverse forest conditions. Seven diverse point cloud datasets were manually segmented to train and evaluate this model, resulting in per-class segmentation accuracies of Terrain: 95.92%, Vegetation: 96.02%, Coarse Woody Debris: 54.98%, and Stem: 96.09%. By exploiting the segmented point cloud, we also present a method of extracting a Digital Terrain Model (DTM) from such segmented point clouds. This approach was applied to a set of six point clouds that were made publicly available as part of a benchmarking study to evaluate the DTM performance. The mean DTM error was 0.04 m relative to the reference with 99.9% completeness. These approaches serve as useful steps toward a fully automated and reliable measurement extraction tool, agnostic to the sensing technology used or the complexity of the forest, provided that the point cloud has sufficient coverage and accuracy. Ongoing work will see these models incorporated into a fully automated forest measurement tool for the extraction of structural metrics for applications in forestry, conservation, and research.

Keywords: deep learning; segmentation; forest; point cloud; LiDAR; photogrammetry; terrestrial laser scanning; structure from motion; automated; digital terrain model

1. Introduction

Forest measurements are important in a number of fields including, but not limited to, forestry, climate science [1–3], fire risk management [4,5], and understanding habitat structural complexity [6–8]. Modern remote sensing techniques such as Light Detection and Ranging (LiDAR) and photogrammetry are enabling high-quality 3D reconstructions of forests to be collected by operators with little or no surveying training. Particularly transformative are techniques such as close-range photogrammetry, which enable researchers and foresters to collect high accuracy and high-resolution 3D reconstructions of forests with low-cost, consumer-grade cameras [9,10] and low-cost Unoccupied Aircraft Systems (UAS) [11,12]. While the capability to collect such rich datasets is becoming more widespread and accessible, a major obstacle impeding the utility of such datasets is the complexity of extracting reliable and useful measurements from them.

There are a number of tools available for extracting measurements from forest point clouds [13–17]; however, many of these tools require manual tuning of parameters, manual

interventions/point cloud editing, and can have complex end-user workflows. There are many challenges such as occlusions, complex structures, understory vegetation, and rugged terrain that are not yet well handled by existing approaches [18], resulting in applied forest studies commonly resorting to time-consuming manual methods of extracting measurements from forest point clouds [19]. Of particular note is that most point cloud tools are intended primarily for very high-quality Terrestrial Laser Scanning (TLS) point clouds and are rarely transferable to noisier point clouds captured by approaches such as photogrammetry, Mobile Laser Scanning (MLS), or high-resolution Aerial Laser Scanning (ALS).

A number of the previously mentioned tools begin with a similar processing pattern. First, a Digital Terrain Model (DTM) is extracted, followed by extracting point cloud slices parallel to this DTM. These slices will ideally contain circular clusters of points that represent only stems, which can be clustered and measured with circle or cylinder fitting algorithms. Some approaches will use these as seed points for applying more complex information extraction techniques, while others will simply measure a slice in the vicinity of 1.3 m above the DTM to measure Diameter at Breast Height (DBH). This approach is relatively simple to implement and can be highly effective in situations without any leaves or understory in the sliced region and when stems are well separated (such as in intensively managed and pruned plantation forestry). However, this method can perform poorly when points belonging to understory vegetation result in the incorrect clustering of multiple trees into one or false detections of stems. In practice, it is common for forest point clouds to contain understory vegetation, so it would be beneficial if we could automatically and robustly segment stems and vegetation prior to the measurement process. It is relatively simple for a human to visually differentiate stems from vegetation, even without color information (as demonstrated in Figure 1), yet it is challenging to explicitly program an algorithm to perform this task effectively.

Semantic segmentation refers to the separation of a dataset into meaningful subsets. In the case of this paper, we are focused on separating parts of a forest into terrain, vegetation, coarse woody debris, and stem categories from a point cloud. There have been many different approaches to the segmentation of forest point clouds [20–34] so far. Some approaches use heuristics [20,22,25,28,29] or morphological operations [27], while others use supervised [23,26,30–33] or unsupervised [21,34] machine learning techniques. With supervised machine learning techniques, the first major challenge to building a model is to obtain or generate sufficient and appropriate training data. Some works in this area [23,33] approach this through the use of artificially generated datasets, which are made from simulating a forest and a terrestrial laser scanning operation to create perfectly labeled point clouds on demand. This approach is certainly logical, as manually labeling point cloud datasets is time-consuming and monotonous while also requiring skilled and attentive operators; however, it can be difficult to generate synthetic datasets with all of the same challenges present within real-world point clouds. Occlusions and ranging noise can be generated in these workflows with relative ease; however, it is difficult to account for all possible sensing difficulties and sources of error. Movement of the trees due to breeze, imperfect reconstructions during photogrammetry, and variable optical properties of the environment can all present sensing challenges and artefacts that are difficult to simulate at this time.

The segmentation works described above were mostly designed with individual sensing methods in mind, such as TLS [20–23,27,28,33,34], MLS [25,28], or ALS [26,28–32], resulting in limited transferability to point clouds captured using other methods with the exception of [28], whose approach was demonstrated on ALS, TLS, and MLS. To move toward the idea of a fully automated and universal forest point cloud processing tool, the goal of our work was to create a workflow capable of automatically segmenting forest point clouds from a variety of sensors with varied point cloud accuracy, density, and mapping challenges. We aim for this model to handle point clouds from TLS, MLS, high-resolution ALS, and photogrammetry (terrestrial/aerial close range or aerial high-resolution nadir).
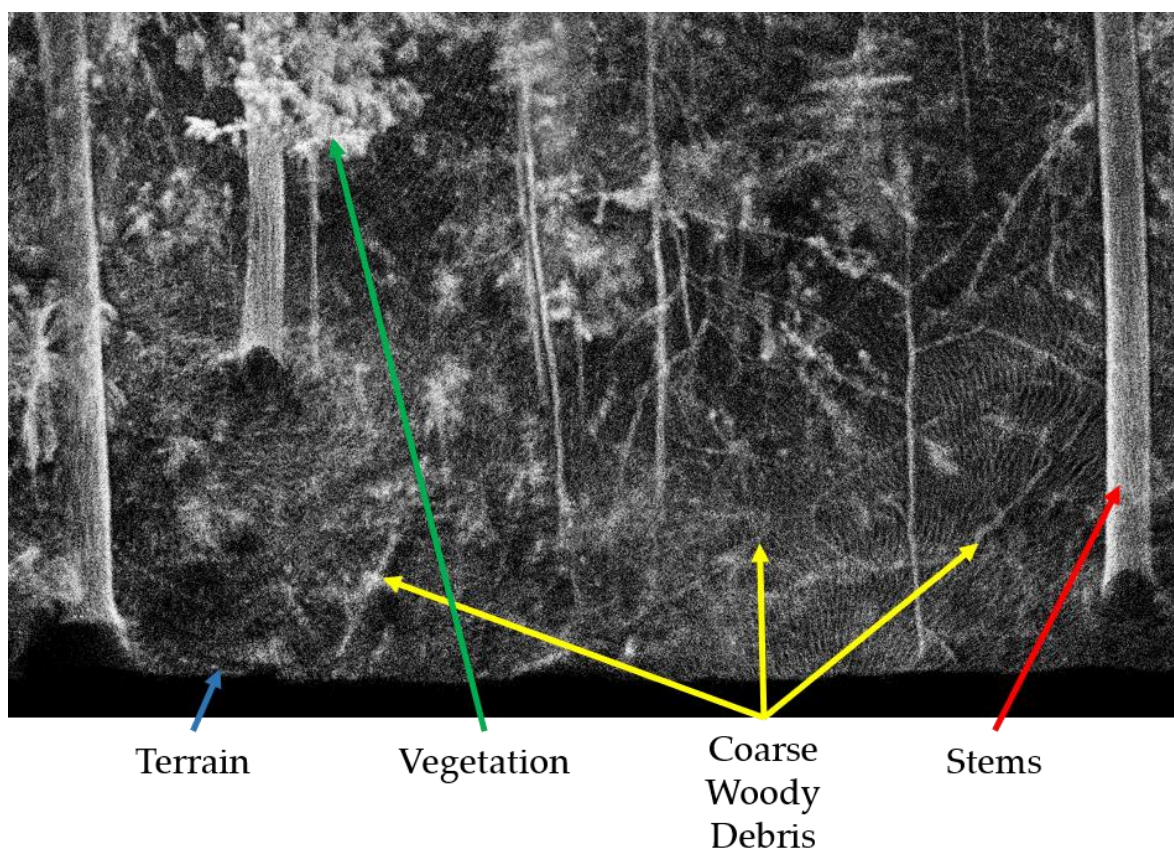
Figure 1. Objects within a Mobile Laser Scanned (MLS) point cloud can still be interpreted with relative ease by a human despite having no color information. We can easily identify which points belong to terrain, vegetation, coarse woody debris, and stems in most cases. While this figure is only two-dimensional (making interpretation more challenging), these objects are considerably more recognizable when viewing the point cloud directly, as it is easier for us to perceive the structure while translating/rotating the point cloud.

Our paper contributes a successful semantic segmentation approach based upon a modification of the Pointnet++ [35] architecture. To train the model to perform on diverse datasets, we manually segmented point clouds from a diverse set of sensors. As this segmentation approach extracts the terrain points, we also provide a method to exploit this information to create a Digital Terrain Model (DTM) that is robust to complex understory vegetation, photogrammetry noise, and uneven terrain. Finally, we validated the accuracy and coverage of our DTM approach against six point clouds from an international benchmarking dataset [18].

2. Materials and Methods

2.1. Methodology Overview

This work was motivated by the idea that a well-segmented point cloud would simplify the forest point cloud measurement process in the presence of diverse and imperfect datasets. Here, we describe the creation of our training and evaluation datasets, the architecture and training approaches for the deep learning model, an approach to the generation of a Digital Terrain Model (DTM) from the segmented point cloud, and how these models and approaches were validated. Figure 2 shows a schematic of how the methods described in this paper fit into a larger-scale project that will incorporate the semantic segmentation and DTM generation tools into a comprehensive forest structural measurement tool that will be able to handle diverse forest point clouds (of high resolution).
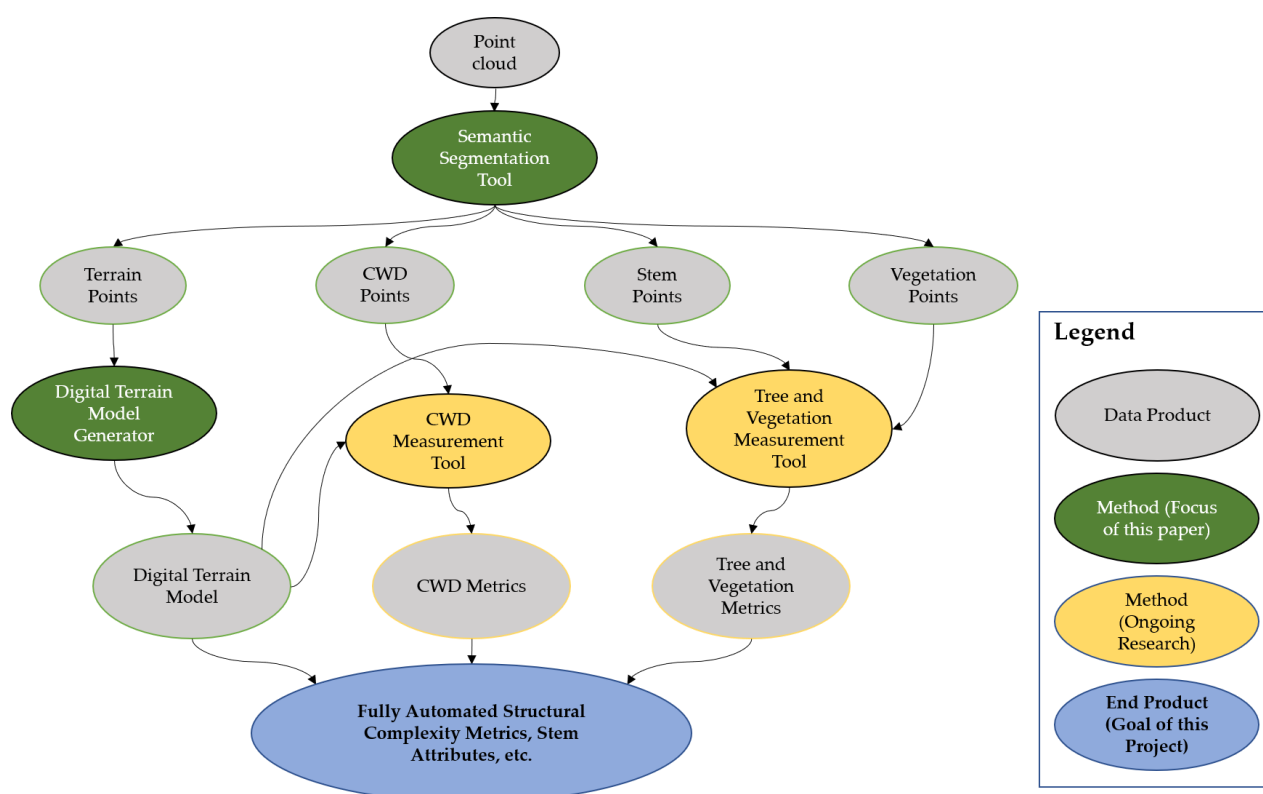
Figure 2. Schematic diagram describing how this research, which focuses on semantic segmentation and Digital Terrain Model generation, fits into our larger goal of creating a fully automated forest point cloud measurement tool.

2.2. Class Selection Approach

The classes for semantic segmentation were chosen based on visual inspection of the point clouds with color information omitted. While some implementations of Pointnet-like architectures exploit Red, Green, and Blue (RGB) color information or LiDAR return intensity/reflectance, our model is intended to work on spatial (X, Y, Z) coordinates alone such that it can work on most (if not all) high-resolution* forest point clouds. The point cloud visualization and editing tool CloudCompare [36] was used with "Eye-Dome-Lighting" mode enabled for this step, which makes it possible to perceive the 3D structure without a colored point cloud. Classes that the authors could reliably distinguish from 3D structure alone were noise, terrain, vegetation, coarse woody debris (CWD), and stems.

A point is considered to be terrain if it appears to be part of the ground surface (according to the human labeling the dataset). The vegetation class is used as a catch-all class for any points that were not terrain, CWD, or stem points. As such, any nearby points above or below the ground surface that were not considered to be terrain or CWD were labeled as vegetation. The CWD class consists of any obvious fallen timber/branches laying on the ground. While this class could have been merged with the stem class, it was separated for the following reasons. First, we need to distinguish between a log on the ground and a standing tree stem that may be adjacent. When clustering with the Density-based spatial clustering of applications with noise (DBSCAN) algorithm in post-segmentation processing steps (beyond the scope of this paper), these would be considered one tree if they were not in separate classes, which is undesirable for our processing approach. Secondly, the reconstructed CWD that we wish to classify included more variable structures than the stem class. For example, we wish to detect partially decomposed CWD, which can have a different structure to what the stem class is intended to represent, particularly in photogrammetric datasets.

The stem and vegetation classes were intended to separate the well-reconstructed woody material from the leaf material. We observed that as the reconstruction quality

reduces, points that may be from a branch or stem become indistinguishable from points that may belong to leaf material. As a result, there is typically a gradual transition from stem to vegetation class (per our class definitions) as noise/measurement errors increase, stem/branch sizes decrease, or occlusions lead to poor reconstruction. If a section of a branch or stem is very poorly reconstructed, there is little use in trying to measure the diameter of it, as it will almost certainly be incorrect; however, the points can still be useful for measuring the amount of canopy vegetation.

Manually labeling the point clouds requires some operator discretion; so for the sake of consistency, only one person manually segmented the entire dataset.

* When we refer to "high-resolution" in this paper, we are referring to any forest point clouds where tree stem diameters could be directly measured from the point cloud.

2.3. Segmentation Model Dataset Generation

The point clouds used in this study came from a variety of sources, forest conditions, and sensor systems. These point clouds were captured using Terrestrial Laser Scanning (TLS), Aerial Laser Scanning (ALS), Mobile Laser Scanning (MLS), and Unmanned Aerial System (UAS)-based aerial photogrammetry (UAS_AP). Forest conditions included open woodlands, pine plantations, and dense eucalyptus forests of varying structural complexity and were collected in various locations throughout Australia and New Zealand. Seven point clouds (described in Table 1) were manually segmented using the segmentation tool in CloudCompare [36] into the 4 class categories (Terrain, Vegetation, CWD, Stem).

Table 1. Description of manually labeled point cloud datasets, their dominant plant species, and plot dimensions. Counted stems were >100 mm diameter at breast height. Stem counts were automatically extracted and not manually measured.

Dataset IDs	Sensing Method (Sensor)	Plot Details	Forest Type	Location	Source
TLS_1	Terrestrial Laser Scanner (Riegl VZ400)	Square 20 × 20 m 11 Stems 275 Stems/ha	Dry Sclerophyll Box-Ironbark Woodland	Rushworth forest, Victoria, Australia	Provided through the TERN Data Portal [1,37,38]
TLS_2	Terrestrial Laser Scanner (Riegl VZ400)	Square 20 × 20 m 7 Stems 175 Stems/ha	Dry Sclerophyll Box-Ironbark Woodland	Rushworth forest, Victoria, Australia	Provided through the TERN Data Portal [1,37,38]
TLS_3	Terrestrial Laser Scanner (Leica_RTC360)	Square 20 × 20 m 32 Stems 800 Stems/ha	Pinus radiata Plantation	Rotorua, New Zealand	Interpine Group Ltd.
VUX_1LR_1	Aerial Laser Scanner (Riegl VUX-1LR, helicopter mounted)	Square 20 × 20 m 10 Stems 250 Stems/ha	Pinus radiata Plantation	Tumut, New South Wales, Australia	Interpine Group Ltd.
UAS_AP_1	Above canopy UAS Photogrammetry (DJI Phantom 4)	Square 20 × 20 m 18 Stems 450 Stems/ha	Eucalyptus amygdalina Open Woodland	Midlands, Tasmania, Australia	Collected by authors.
HOVERMAP_1	Mobile Laser Scanner (Emesent Hovermap)	Square 20 × 20 m 25 Stems 625 Stems/ha	Pinus radiata Plantation	Rotorua, New Zealand	Interpine Group Ltd.
HOVERMAP_2	Mobile Laser Scanner (Emesent Hovermap)	Circular 40 m diameter 74 Stems 589 Stems/ha	Pinus radiata Plantation	Rotorua, New Zealand	Interpine Group Ltd.

When manually segmenting the point clouds, color information can be helpful for the human operator to differentiate objects in the point cloud; however, care must be taken to avoid the creation of contradictory training information. The model is relying

on spatial coordinates alone, so any segmentation performed using the color information must be carefully checked to ensure that the class of interest is able to be identified by a human with only spatial information (i.e., no color). Particularly in the cases of the photogrammetry datasets, CWD can be visible in colorized point clouds, but spatially, it can not be reconstructed in a way that is distinct from the underlying terrain points. We wish to train the model to predict CWD only when it is clearly present in the 3D structure.

Once segmented, these point clouds were split (0.5/0.25/0.25) into training, validation, and test sets at the individual point cloud level. Figure 3 visualizes the data split and the manually labeled point clouds. We did not split these datasets blindly, as it is necessary to ensure that representative samples of each point cloud were present in the training, validation, and test sets. This is necessary as the datasets are imbalanced simply due to the structure of forests, as there were far more stem and vegetation points than CWD points. If we were to take the approach of blindly splitting the data, we would run the risk of providing insufficient CWD samples to the model during training or none during validation/testing, leading to poor performance and/or an inappropriate evaluation of the model's performance.
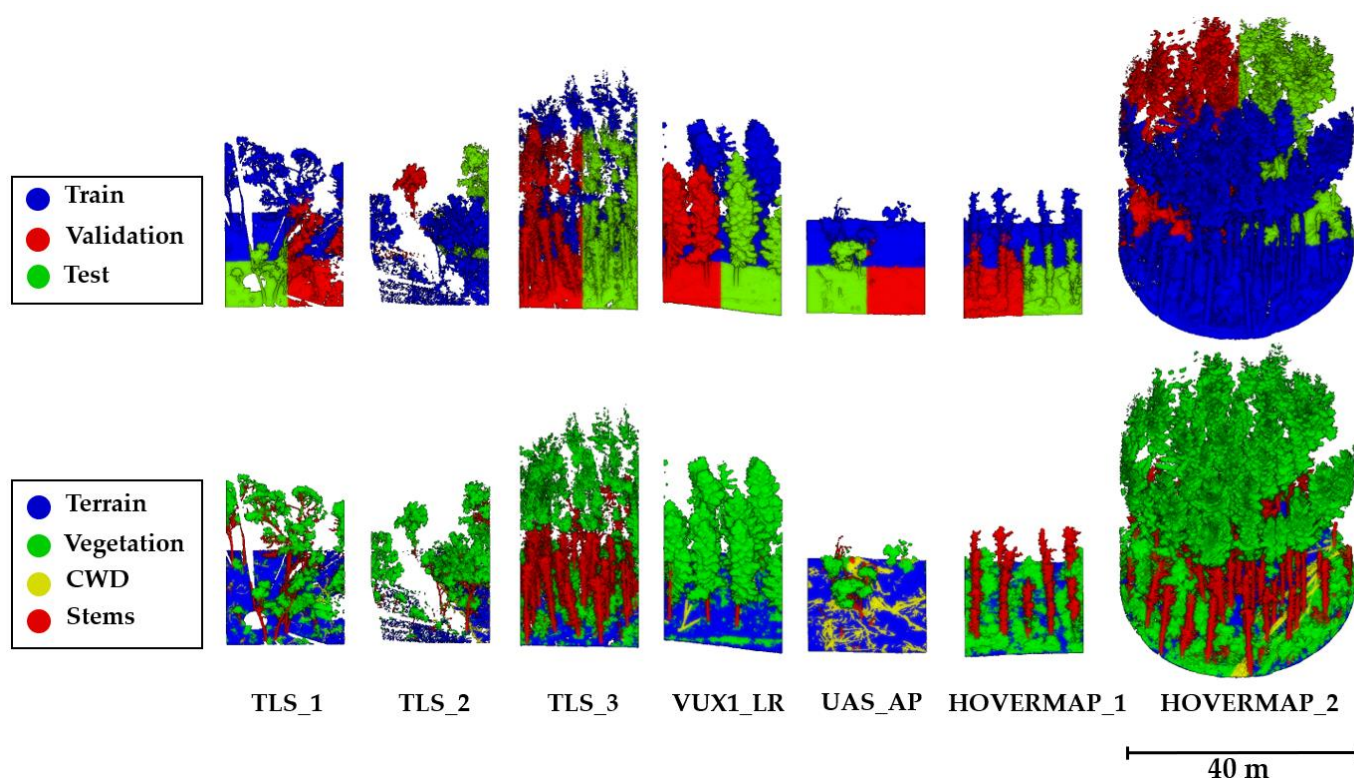

Figure 3. Datasets from multiple sensing systems and multiple forest conditions were manually segmented using the segmentation tool in CloudCompare. These were split into training, validation, and test sets as per the top row. The dataset labels represent the sensor used to collect the point clouds. The expanded abbreviations are as follows: Terrestrial Laser Scanning (TLS), Aerial Laser Scanning from a helicopter using a Riegl VUX-1LR Light Detection and Ranging (LiDAR) (VUX1_LR), Unmanned Aircraft System Aerial Photogrammetry (UAS_AP), Mobile laser scanning using a handheld Emesent Hovermap LiDAR (HOVERMAP). For better visualization, please see this video of these datasets (Krisanski, S. et. al, Sensor Agnostic Semantic Segmentation of Forest Point Clouds using Deep Learning (Part 1), https://www.youtube.com/watch?v=MGRQDZZ1QBo, accessed on 30 March 2021).

For the purposes of training and evaluating the model, the canopy was fully removed from the HOVERMAP_1 dataset and partially trimmed from the TLS_3 and HOVERMAP_2 datasets, as manually segmenting the multitude of small branches was both unfeasible from a time perspective and also highly ambiguous. The ambiguity arises due to choosing a boundary between points belonging to either the stem or vegetation class

as stems/branches become noisier with increasing height due to the increasing sensing distance, beam divergence effects, and occlusion effects (from a ground-based LiDAR). Figure 4 shows a close-up section of the HOVERMAP_2 dataset, which visualizes the ambiguity existing in the original point clouds. We cannot evaluate the model against a human baseline in regions where a human cannot consistently label the points, so we took the approach of removing the ambiguous regions from several of the training, testing, and validation datasets as appropriate. The model was trained and evaluated on the clear examples, leaving the model to choose the decision boundaries when used in practice, based upon what was learned from the clear examples. For further clarity on the training, validation, and test datasets, we encourage readers to watch this visualization video (Krisanski, S. et. al, Sensor Agnostic Semantic Segmentation of Forest Point Clouds using Deep Learning (Part 1), https://www.youtube.com/watch?v=MGRQDZZ1QBo, accessed on 30 March 2021).
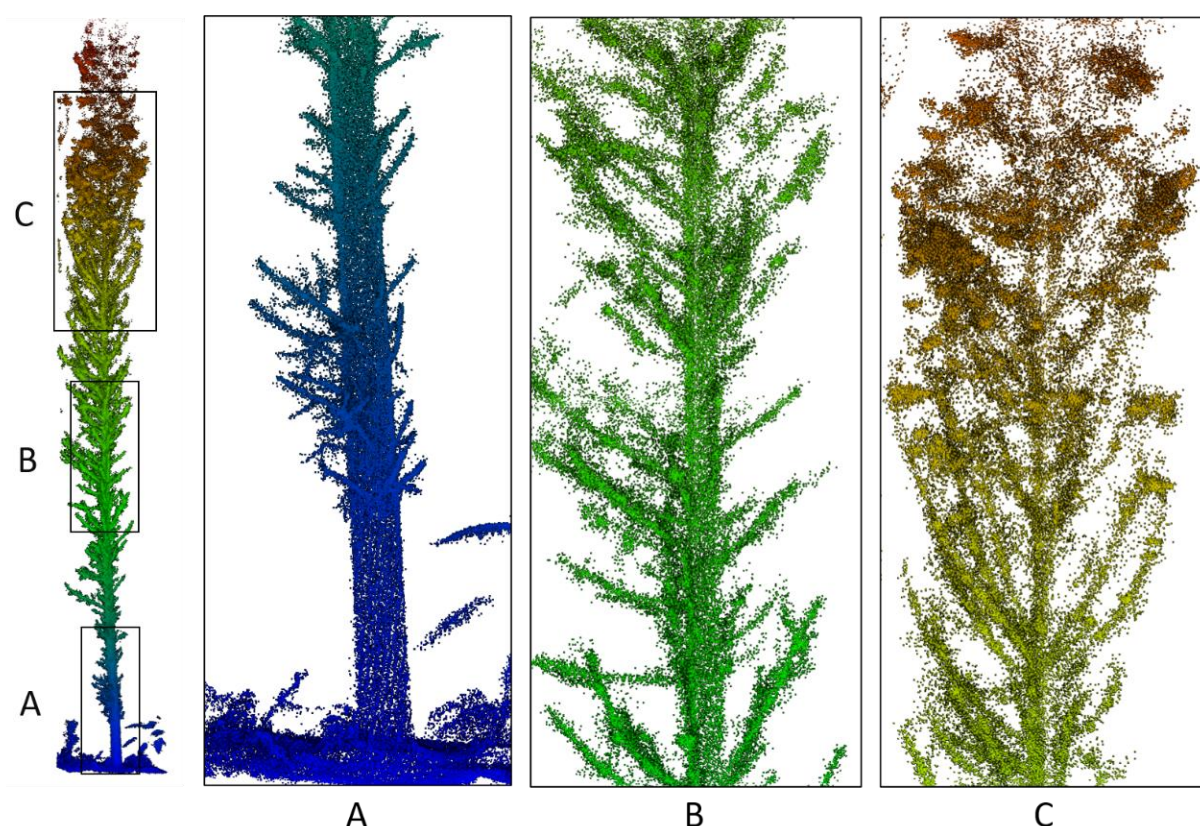

Figure 4. Above shows an example of the ambiguity resulting from a combination of error induced by beam divergence effects and increasing occlusion effects with height. (**A**) Shows a very clearly reconstructed stem. (**C**) Shows points from the canopy which would be labeled as vegetation class per our definitions. (**B**) Shows the ambiguous region in between (**A**) and (**C**), where stems are identifiable, but are in-between the stem (**A**) and vegetation (**C**) class definitions. Ambiguous regions as shown in (**B**) were removed from several of the training, testing and validation datasets as needed.

The training dataset (shown in Figure 3) was cloned twice with each clone being scaled by a factor of 0.5 and 2.0, respectively. The cloned point clouds that were downscaled to 0.5 of their original size were subsampled to 0.01 m resolution (minimum distance between points) to provide training examples of smaller sized objects at the same 0.01 m resolution as the original dataset. The upscaled clone was not subsampled as doubling the point cloud scale halves the effective point cloud resolution.

2.4. Network Architecture

The architecture we used was based upon Pointnet++ [35], which was chosen due to its ability to perform semantic segmentation of unordered point clouds directly and efficiently without the need for voxelization. The main changes we made from the Pointnet++ architecture were to increase the size of the model to increase the learning capacity enough to handle up to 20,000 points per sample versus 1024 points per sample in the original paper. For detailed explanations of the set abstraction and feature propagation modules, please see the original Pointnet [39] and Pointnet++ papers [35]. The Pytorch Geometric [40] implementation of the Pointnet++ segmentation architecture was used as the starting point for this work, with our modified architecture shown in Figure 5. We have described the architecture in the same structure as the Pytorch Geometric segmentation examples for ease of implementation.
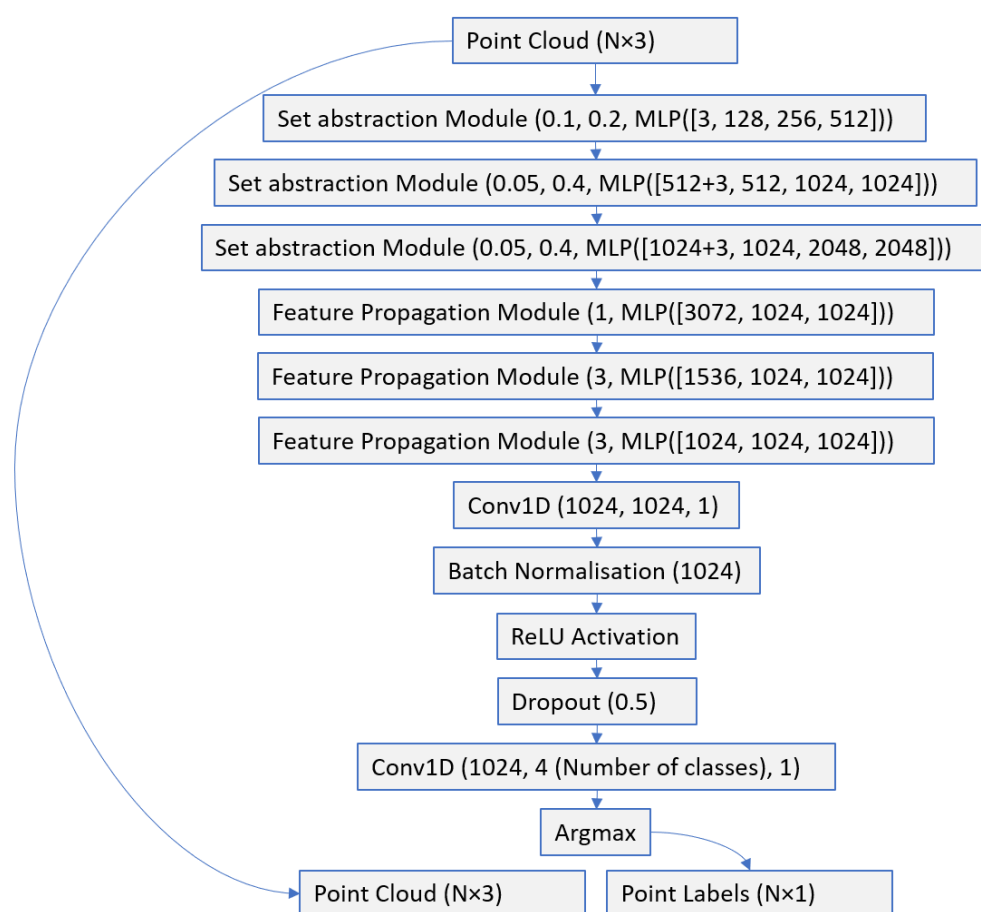


```
Point Cloud (N×3)
        ↓
Set abstraction Module (0.1, 0.2, MLP([3, 128, 256, 512]))
        ↓
Set abstraction Module (0.05, 0.4, MLP([512+3, 512, 1024, 1024]))
        ↓
Set abstraction Module (0.05, 0.4, MLP([1024+3, 1024, 2048, 2048]))
        ↓
Feature Propagation Module (1, MLP([3072, 1024, 1024]))
        ↓
Feature Propagation Module (3, MLP([1536, 1024, 1024]))
        ↓
Feature Propagation Module (3, MLP([1024, 1024, 1024]))
        ↓
Conv1D (1024, 1024, 1)
        ↓
Batch Normalisation (1024)
        ↓
ReLU Activation
        ↓
Dropout (0.5)
        ↓
Conv1D (1024, 4 (Number of classes), 1)
        ↓
Argmax
        ↓
Point Cloud (N×3)        Point Labels (N×1)
```

Figure 5. The network architecture used in this paper was based upon the Pytorch Geometric [40] implementation of Pointnet++ [35] with some modifications to increase the size and learning capacity of the network. Abbreviations: Multilayer Perceptron (MLP), 1 Dimensional Convolution (Conv1D), Rectified Linear Unit (ReLU).

2.5. Data Pre-Processing

While the Pointnet++ [35] architecture is able to process point clouds directly, it was not made to ingest large point clouds all at once (such as TLS point clouds, which may contain greater than 1 billion points). For segmentation in the original Pointnet++ paper, they used subsets of the point cloud with 1024 points or less. Our approach involves slicing the point cloud into cube-shaped regions of side length 6 m with a minimum of 500 and a maximum of 20,000 points. If one of these cubes contains greater than 20,000 points, points are removed at random until 20,000 points remain. If a cube contains less than 500 points, it is not used to avoid processing empty or nearly empty cube samples. Then, 6 m cube sized

samples with less than 500 points were typically difficult for humans to correctly classify, so they were considered to be sub-optimal examples for the model to learn from. The 6 m size is an arbitrary size chosen during early experimentation to capture enough context that humans could identify objects in the samples in most cases, while a sample was limited to 20,000 points. Smaller sample boxes offered less context, making them more difficult to classify. Larger sample boxes become lower resolution if we retain the 20,000 point cut-off we assigned and again become more difficult to classify by humans and machines alike.

These cube regions are overlapping in X, Y, and Z dimensions with 0.75 overlap used for training data, 0.5 overlap used for validation, and 0.5 overlap for testing data.

Each cube is shifted to the origin prior to inference to avoid floating point precision issues when dealing with large numbers from global coordinates. Pre-processing is performed before training or inference, and each sample is stored in a file. During training, the samples are seen repeatedly, so we need not pre-process each sample multiple times this way. During inference, the pre-processing will bottleneck our inference process, so loading each sample from a file allows the Graphics Processing Unit (GPU) to be working at near full capacity. By pre-processing the data before training or inference, we can also take advantage of parallel processing more easily. To minimize computational time, our pre-processing approach takes advantage of vectorization as much as possible through extensive use of the NumPy [41] package.

2.6. Data Augmentation and Model Training

Data augmentation is applied for training samples in the form of random rotations about X, Y ($\pm 15°$), and Z ($\pm 180°$) axes and random scale changes by multiplying coordinates by a factor of 0.8 to 1.2. If there is no terrain or CWD present in a sample, the X and Y axis rotations are randomly chosen between $\pm 90°$ instead of $\pm 15°$. We did this because we do not wish to train the model to predict terrain class on vertically oriented surfaces (such as on the side of a particularly large diameter tree), but valid stems can be completely horizontal.

For each cube-shaped sample, there was a 50% chance of adding random noise to the X, Y, and Z coordinates with a randomly chosen standard deviation of between 0.01 m and 0.025 m and a mean of 0 m, applied at a per-point level. The training dataset consisted of 112,758 samples prior to the random augmentation, which was applied throughout training to minimize the risk of overfitting and to aid the generalizability of the model to unseen data.

To minimize contradictory training information, if a sample contains CWD but no ground points, the CWD is relabeled to Stem class during training. The intent behind this condition is for the model to learn that CWD should be near the ground and that CWD is similar to the Stem class in some circumstances.

All training and testing were performed on a desktop computer with an Intel i9-10900K CPU, 128 gigabytes (GB) of DDR4 RAM, and an Nvidia Titan RTX graphics processing unit (GPU) with 24 GB of Video Random Access Memory (VRAM). The model was trained for 300 epochs with a batch size of 8 (limited by GPU VRAM), taking approximately 3 days. Figure A1 shows the changes in accuracy and loss of the train and validation sets over the 300 epochs.

The model was trained using cross-entropy loss with an initial learning rate of 5×10^{-5}, which was reduced to 2.5×10^{-5} after 150 epochs. The initial learning rate was chosen through experimentation, where we found that higher learning rates lead to erratic loss values or exploding gradients.

2.7. Model Inference

When used for inference, the model is used with a sliding box overlap of 0.5 in X, Y, and Z axes. For each point in the segmented point cloud, up to 16 nearest neighbors are found within a maximum search radius of 0.1 m. The median prediction scores are computed for each class prediction, followed by an argmax function to select the final point

label. The initial segmented point cloud may be down-sampled in some regions through the process of enforcing a maximum of 20,000 points per sample region, so to label the full original point cloud, each point in the original point cloud is assigned the label of its nearest neighbor in the segmented point cloud.

2.8. Semantic Segmentation Evaluation Method

The segmented point cloud was evaluated on an individual point basis against the manually segmented point cloud dataset. The Python package Scikit-Learn [42] was used to evaluate the model and generate a confusion matrix of the results. As manually labeling point clouds is highly time-consuming, there was a practical limit to how many point clouds we could quantitatively evaluate the segmentation model on. In the interests of transparency and in order to demonstrate the utility and limitations of the tool on a larger range and scale of datasets, we have provided a fly-through video of several additional datasets segmented by the model. These datasets are described in Table 2.

Table 2. Description of unlabeled point cloud datasets shown in the fly-through video. Counted stems were > 100 mm diameter at breast height. Stem counts were automatically extracted and not manually measured.

Dataset IDs	Sensing Method (Sensor)	Plot Details	Forest Type	Location	Source
TLS_4	Terrestrial Laser Scanner (Riegl VZ 400i)	Circular 30 m diameter 110 Stems 589 Stems/ha	Araucaria Cunninghamii	Queensland, Australia	Interpine Group Ltd.
HOVERMAP_3	Mobile Laser Scanner (Emesent Hovermap)	Circular 50 m 205 Stems 1556 Stems/ha	Pinus Radiata Plantation	Rotorua, New Zealand	Interpine Group Ltd.
UAS_AP_2	Above canopy UAS Photogrammetry (DJI Phantom 4)	Square 90 × 90 m 350 Stems 432 Stems/ha	Eucalyptus Amygdalina Open Woodland	Midlands, Tasmania, Australia	Collected by authors.
VUX_1LR_2 *	Aerial Laser Scanner (Riegl VUX-1LR—helicopter mounted)	Rectangle 120 × 60 m 220 Stems 306 Stems/ha	Pinus Radiata Plantation	Carabost, New South Wales, Australia	Interpine Group Ltd.
UC_UAS_AP_1	Under canopy UAS Photogrammetry (DJI Phantom 4)	Circular 26 m diameter 47 Stems 885 Stems/ha	Eucalyptus Pulchella Native Forest	Fern Tree, Tasmania, Australia	Collected by authors, same point cloud as "Plot 1" in [11]

* CloudCompare's "Statistical Outlier Removal" was applied to VUX_1LR_2 with the default settings prior to processing to speed up the processing time.

2.9. Digital Terrain Model Generation

Once a point cloud has been segmented by the model, the points labeled with "terrain" class can easily be extracted for use in the generation of a Digital Terrain Model (DTM). In Figure 6, we provide a pseudocode describing the process used to generate a DTM using the segmented point cloud.

1. Start with Segmented Point cloud.

2. Extract points classified as "terrain".

3. Use the DBSCAN algorithm with epsilon of 0.1, metric = "Euclidean", algorithm = "KDTree".

4. Remove clusters with less than 500 points.

5. Generate a 2D grid to become the DTM.

6. For each point in the 2D grid:

    a. Search radius = DTM Resolution (0.2 m)

    b. While (number of neighbours < 20) and (radius < 5 m):

        i. Neighbours = all terrain points within 2D (X, Y) search radius (performed using a KDTree).

        ii. Search radius += DTM Resolution (0.2 m)

    c. If (number of neighbours > 0)

        i. The Z value for the current point in the 2D grid = the median Z value of the neighbours.

        ii. Add the current point to the DTM array. Points without neighbours are therefore ignored.

7. Remove any DTM points that do not have neighbours from the original point cloud within a 2D (X, Y) search radius of DTM Resolution (0.2 m). This trims the DTM to the shape of the point cloud in X and Y.

8. For all points in the 3D DTM grid:

    a. Find all neighbours within a 2D radius of 3 m.

    b. Compute the mean of the neighbours Z value and store it in an array.

9. Replace all Z values of the DTM with the smoothed Z values.

10. Cluster the DTM points with the DBSCAN algorithm with epsilon of DTM Resolution*3, metric = "Euclidean", algorithm = "KDTree", minimum samples = 15

11. Any outlier points have their Z value replaced with the median Z value of all DTM points within a 2D (X, Y) radius of DTM Resolution*3

12. Return the DTM.

Figure 6. Pseudocode for our method of generating of a Digital Terrain Model (DTM) from a segmented terrain point cloud. DBSCAN and KDTree implementations were from [42] and [43], respectively.

2.10. Digital Terrain Model Evaluation Method

To evaluate the performance of our Digital Terrain Model (DTM), we applied it to 6 point clouds made publicly available by a TLS benchmarking study [18]. In their study, they generated the reference DTMs by first classifying the ground points using the TerraScan software [44], followed by manual removal of non-ground objects. They applied a 20 cm resolution grid for rasterization and used the mean height of the ground points within each cell. In cases without points, the height value was interpolated using the average of the neighboring cells.

To compare our DTM height measurements against the reference DTMs provided by the benchmarking study, we used a 20 cm grid resolution in our algorithm, followed by 2D linear interpolation to the positions of the reference DTM, as there was a small offset between predicted and reference grid point positions due to minor differences in how the grid is generated.

The benchmarking study also introduced a measurement called DTM Coverage. The DTM coverage is the ratio of the covered reference DTM points and the total reference DTM points. For each point in the reference DTM, the nearest neighbor from our DTM was found. If the distance to the nearest neighbor was less than 0.2 m, the point was "covered". A score of 1 implies the predicted DTM completely covered the region of the reference DTM. This approach accounts only for the points in the reference DTM that are or are

not covered and it does not consider the possibility of covering a greater area than the reference DTM.

3. Results

3.1. Semantic Segmentation Evaluation

The semantic segmentation results are visualized in Figure 7, with the manually segmented reference point clouds on the left and the model's predictions on the right of each pair. These point clouds are the "Test" dataset first shown in the top half of Figure 3.
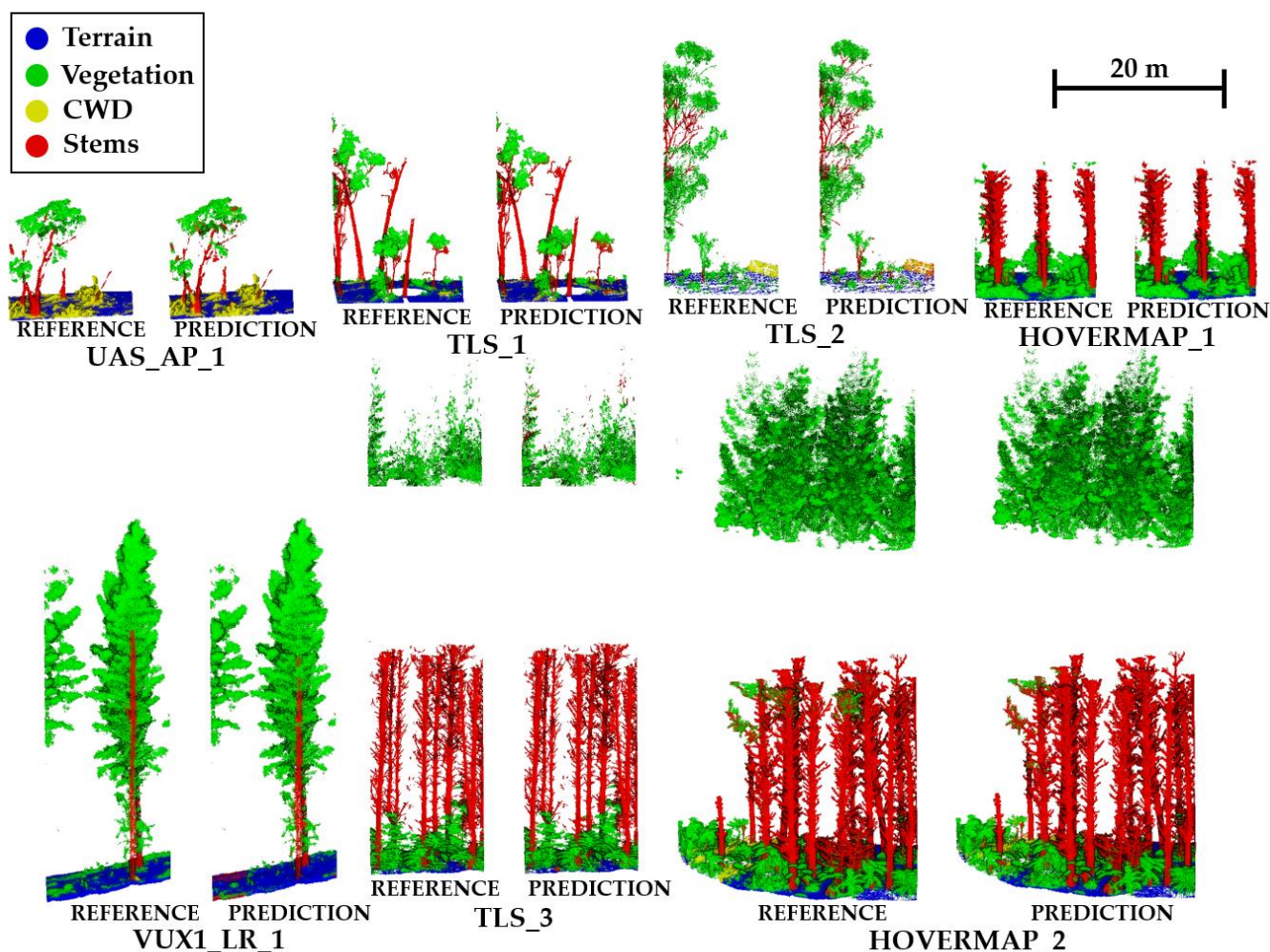

Figure 7. Visualization of the semantic segmentation results. For each pair, the left point cloud shows the manually labeled reference and the right point cloud is the model's label predictions. The predicted labels are visually very similar to the reference dataset, with the most obvious differences being the few misclassifications of coarse woody debris (CWD) as stems in TLS_2, and some terrain being misclassified as stem in VUX_1LR_1.

The predictions were visually very similar to the reference dataset. Of the 4 class labels, we observed that the model was least accurate at segmenting the CWD class, with the clearest examples of this in the TLS_1, TLS_2, and HOVERMAP_2 datasets. Ground points that were misclassified as stem points can be seen in the VUX_1LR_1 dataset, and the stem was not extracted as far up the tree as in the human segmented point cloud. It appears to be uncommon for ground points to be misclassified as stem points in our other testing datasets, but when they do occur, it tends to be on the edges of the point clouds where the ground does not completely cover the sample box region. These observations align with what we would expect based upon the quantitative results shown in the confusion matrix in Figure 8.
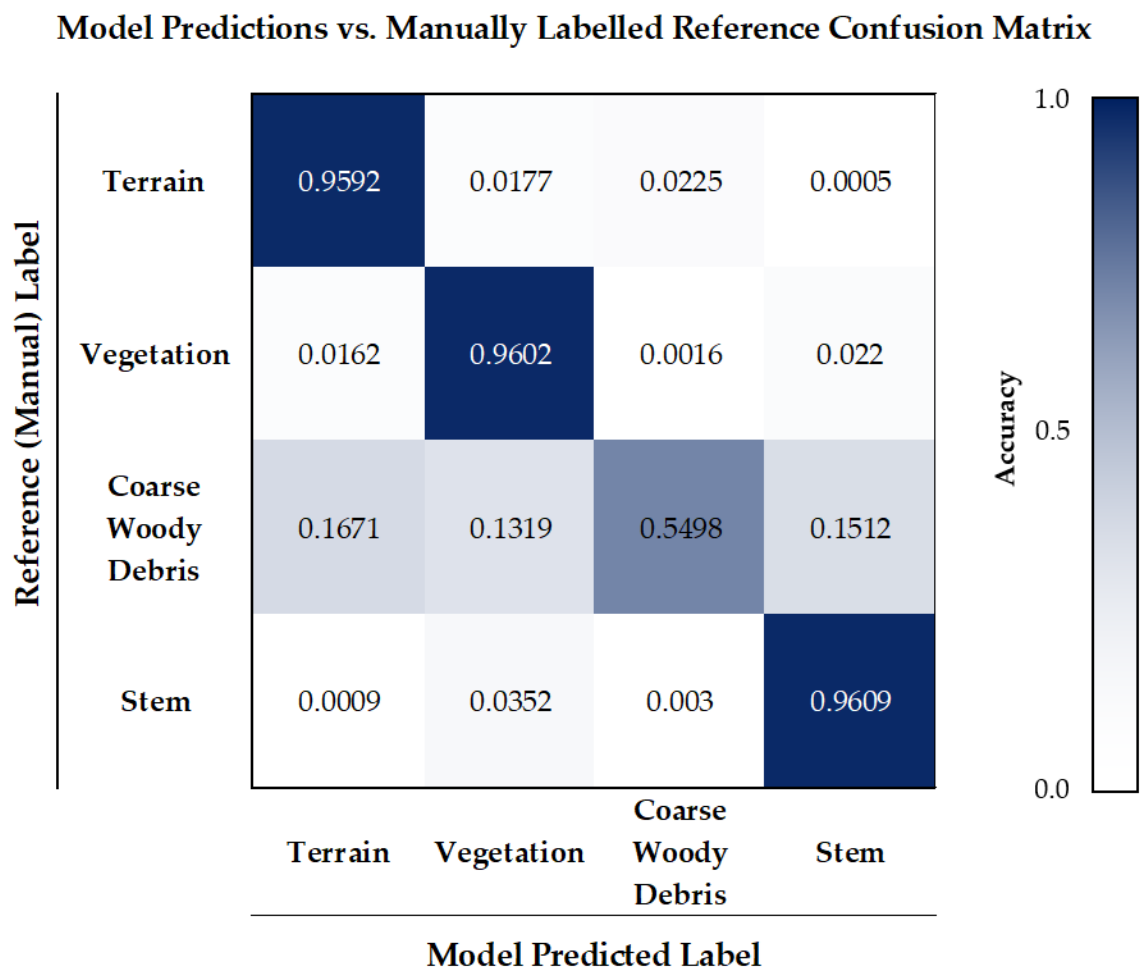
Model Predictions vs. Manually Labelled Reference Confusion Matrix

Figure 8. Confusion matrix showing performance of the semantic segmentation process compared against manually segmented points.

The terrain, vegetation, and stem classes had notably greater accuracy than the CWD class, with stems being predicted with the greatest accuracy on this test dataset. Table 3 presents the per-class recall and precision scores, as well as the overall accuracy, precision, and recall. Per-class accuracies are shown in the confusion matrix in Figure 8.

Table 3. Semantic segmentation results.

	Terrain	Vegetation	CWD	Stem
Recall	0.959	0.960	0.550	0.961
Precision	0.926	0.974	0.610	0.948
IoU	0.891	0.936	0.407	0.913
	Overall			
Accuracy	0.954			
Precision	0.864			
Recall	0.858			

We have also provided a comparison of our segmentation results against others in the literature in Table 4; however, it should be stressed that without identical test datasets and agreement on per-class definitions, it can only be used as an indication of their relative performances.

Table 4. Comparison of segmentation results against similar studies. Bold numbers denote the top score in each metric.

Study	Method	Stem Precision	Stem Recall	Vegetation Precision	Vegetation Recall	Overall Precision	Overall Accuracy
[26] *	3D Fully Convolutional Network	0.595	0.771	**0.985**	0.971	0.790	-
	3D Fully Convolutional Network (with LiDAR intensity)	0.652	0.744	**0.985**	**0.975**	0.819	-
	Pointnet	0.517	0.572	0.976	0.959	0.747	-
	Pointnet (with LiDAR intensity)	0.554	0.727	**0.985**	0.960	0.770	-
[23]	Pointnet++ inspired approach	-	-	-	-	-	0.900 **
[24]	Custom Feature Set + Random Forest	-	-	-	-	-	0.910
[21]	Unsupervised Learning	-	-	-	-	-	0.888
[34]	Unsupervised Learning	-	-	-	-	-	0.925 ***
Ours	Modified Pointnet++ approach	**0.948**	**0.961**	0.974	0.960	**0.961** **** **0.864** *****	**0.961** **** **0.954** *****

* We compared against the Tumut dataset from [26] as it had higher scores than their Carabost dataset. ** Referred to as "Close to 90%" in original paper. Exact value not reported. *** Updated accuracy value since original paper (Wang, D., et al., Unsupervised tree leaf-wood classification from point cloud data. Available online: https://github.com/dwang520/LeWoS, accessed on 10 February 2021) **** Including only stem and vegetation classes. ***** Including all of our classes (terrain, vegetation, CWD, stem).

3.2. Video Demonstration of Semantic Segmentation Performance

The segmentation model was also applied to additional point clouds that were not manually segmented (nor seen to the model during training in any way) to qualitatively identify the performant aspects of the model and identify the weaknesses under various scenarios. To present our results as transparently as possible, a fly-through video of the unseen datasets is provided here (Krisanski, S. et. al, Sensor Agnostic Semantic Segmentation of Forest Point Clouds using Deep Learning (Part 2), https://www.youtube.com/watch?v=v0HwNu6SK6g, accessed on 30 March 2021).This video shows five datasets from five different sensor types as described in Table 2 in the methodology section. The datasets shown in the video are also visualized in Figure 9.
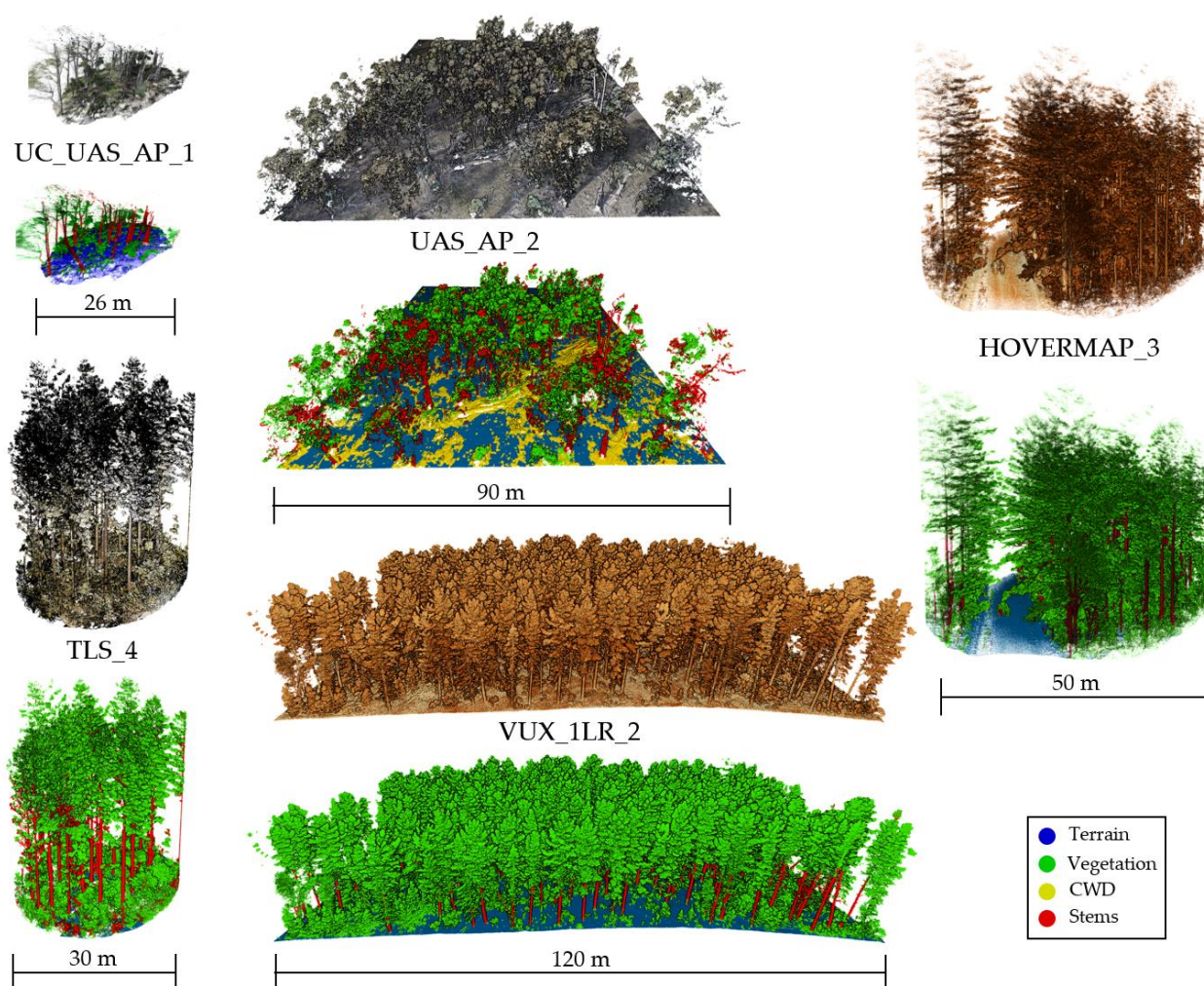
Figure 9. This figure presents five additional larger-scale point clouds from five different sensing techniques/sensors that were automatically segmented by the model. We provide a fly-through video of these datasets with the intent of transparently showing the strengths and weaknesses of the model (Krisanski, S. et. al, Sensor Agnostic Semantic Segmentation of Forest Point Clouds using Deep Learning (Part 2), https://www.youtube.com/watch?v=v0HwNu6SK6g, accessed on 30 March 2021).

Below, we have provided comments on the fly-through video with associated timestamps. These timestamps are linked to the video sections in the description on YouTube. We also provide Figure A2 in the appendix to show an example of the per-class feature maps of the TLS_4 dataset.

3.2.1. TLS_4

- Successfully identified CWD can be seen (00:30).
- Some understory vegetation is misclassified as stem (00:38).
- TLS_4 has some point cloud registration errors in the canopy, which is potentially due to wind during data capture; however, this does not appear to have affected the predictions negatively (00:48).

3.2.2. UAS_AP_2

- As this was captured by above-canopy nadir aerial photogrammetry, many stems were not well reconstructed (01:13)
- Rocks can be seen to be classified as CWD. This was not considered a misclassification since we never provided examples of rocks; however, this suggests rocks could be worth including into future models for quantifying habitat (01:23).

- The bases of many stems were classified as CWD (01:35).
- The upper regions of some CWD were misclassified as stem (01:40).
- Some canopy vegetation was misclassified as stem (01:48).

3.2.3. HOVERMAP_3

- Mostly desirable performance on the Hovermap dataset.
- Some minor branches/stems were mislabeled as vegetation; however, most of these examples are in the ambiguous region between our definition of stem and vegetation, where it would be difficult to measure accurate diameters from the point cloud even if they were detected as stems (02:39).

3.2.4. VUX_1LR_2

- The bases of many stems in this dataset were misclassified as vegetation (03:05).
- CWD was not well detected in noisy point clouds, which is likely as a result of limited training examples in this data type (03:15).
- Upper stems were misclassified as vegetation (03:33).

3.2.5. UC_UAS_AP_1

- A major stem (leaning almost horizontally) and some minor branches/small stems were missed by the model and labeled as vegetation (04:32).
- The main CWD object in the point cloud was partially correctly segmented but was misclassified as vegetation in some regions and misclassified as stem where the CWD contacts a standing stem (04:33).
- A small patch of terrain points were misclassified as stem (04:36).

3.3. Digital Terrain Model Evaluation against Benchmarking Dataset

Our approach to DTM generation was able to cover the entirety of the reference DTM in five out of six cases, with the exception being effectively completely covered also at 0.991 coverage. Table 5 shows the results of the DTM evaluation against the benchmarking.

Table 5. Digital Terrain Model evaluation results.

Plot	Difficulty	DTM Coverage	Mean Error (m)	RMSE (m)
1	Easy	1.0	0.018	0.079
2	Easy	1.0	0.020	0.066
3	Medium	1.0	0.085	0.250
4	Medium	1.0	0.038	0.137
5	Difficult	0.991	0.051	0.166
6	Difficult	1.0	0.025	0.110
Overall Mean		0.999	0.040	0.135

3.4. Processing Times

We have provided Table 6 to demonstrate the processing times on the desktop computer described in Section 2.6. We have also provided Figure A3 in the appendices to visualize these numbers more clearly. A high-resolution TLS point cloud TLS_4 was able to be processed from start to finish in 29 min. However, the VUX_1LR_2 dataset exceeded the 128 GB of RAM available on our desktop computer, which meant that it needed to spill over onto swap space on the M.2 solid state drive for the excess ($\approx$200 GB of swap space was used).

Table 6. Processing times of our method with respect to the number of points (after subsampling to 0.01 m minimum distance) between points.

Dataset	Number of Points *	Number of Sample Boxes	Area ** (ha)	Processing Time (Min)			
				Pre-Processing	Inference	Post Processing	Total Time
TLS_1	819,279	348	0.039	0.36	1.06	0.21	1.63
TLS_2	199,398	304	0.031	0.07	0.28	0.24	0.59
TLS_3	2,200,477	744	0.039	0.63	2.51	0.35	3.49
TLS_4	13,315,371	1343	0.068	10.72	16.81	1.45	28.98
HOVERMAP_1	3,085,477	312	0.040	0.73	1.85	0.42	3.03
HOVERMAP_2	11,328,579	2007	0.125	9.02	16.68	1.64	27.34
HOVERMAP_3	51,310,332	3462	0.212	61.28	45.39	6.81	113.48
UAS_AP_1	564,003	214	0.040	0.16	0.59	0.22	0.97
UAS_AP_2	36,613,477	3969	0.640	43.96	57.85	13.88	115.69
UAS_UC_AP_1	16,154,845	538	0.062	5.33	6.22	3.26	14.81
TLS_BENCHMARK_1	16,861,460	1233	0.103	19.92	24.19	3.32	47.44
TLS_BENCHMARK_2	16,211,608	1056	0.102	20.78	23.14	2.83	47.22
TLS_BENCHMARK_3	19,082,314	1206	0.093	24.39	25.98	2.12	53.02
TLS_BENCHMARK_4	19,982,845	1336	0.093	23.02	27.84	2.14	53.54
TLS_BENCHMARK_5	14,101,093	1206	0.098	20.99	24.81	1.91	47.71
TLS_BENCHMARK_6	11,089,765	1056	0.095	17.81	21.04	1.62	40.47
VUX_1LR_1	1,129,243	148	0.040	1.84	2.30	0.46	4.6
VUX_1LR_2	125,936,807	8175	0.720	252.21	262.94	21.46	536.61

* All point clouds were subsampled to 0.01 m minimum distance between points. ** Area was computed automatically using a convex hull on the terrain labeled points.

In Figure A2, it can be seen that the post-processing step (consisting of the DTM generation process) had a smaller impact on the processing time than the pre-processing and inference steps, as is to be expected. Both pre-processing and segmentation steps appear to have a similar relationship with respect to the number of points in the point cloud. We have fitted a 2nd order polynomial to these points for the purposes of visualizing the trend. From these trends, which appear to increase quadratically with respect to the number of points, the best approach to using this model in practice would be to slice large point clouds into sub-point clouds to be processed in batches before reassembling them (if needed). The optimal slicing size will depend on the computational resources available, as the classification model performs worse on the edges of point clouds (smaller slices means more edges for an equivalent point cloud).

This model is currently only suitable for relatively high-performance desktop computers and up; however, due to the computational expense of working with such large point clouds, it is a reasonable expectation that those interested in our approach will already have a sufficiently powerful workstation. Lower-level computers cannot cope well with point clouds containing hundreds of millions to billions of points, so our method is likely out of reach of lower-end computers at this time.

4. Discussion

4.1. Segmentation

In Table 4, we presented our segmentation results alongside other forest point cloud segmentation studies. This is not an exhaustive list of related works but is intended to serve as an indication of the performance of our approach relative to similar studies. For this comparison, we must acknowledge the limitations that we are not comparing these methods on the same datasets, and that our definitions of stem and vegetation classes may differ slightly from the other studies in the field. The top performing model we found in the literature was [34], achieving an overall accuracy of 92.5%, which is a particularly impressive result considering it was using an unsupervised learning technique, negating the need for labeled training data. [24] achieved a 91% overall accuracy using a technique based upon the random forest technique. [23] used a Pointnet++ inspired approach and claimed an overall accuracy of "close to 90%". [26] tested a variety of approaches, with their best results being on their Carabost dataset. An overall accuracy was not reported; however,

we can compare with respect to overall precision. Using a 3D convolutional neural network on voxels, they reported an overall precision of 79% without LiDAR intensity information, and 81.9% with intensity. [26] also tested a Pointnet-based method that achieved 74.7% without intensity information and 77% with intensity. Our model was able to achieve 96.1% overall accuracy (if only comparing stem and vegetation classes) or 95.4% overall accuracy if comparing our model in its entirety (segmenting all four classes). Our model scored a higher overall precision than all of the models tested in [26]. Whilst we cannot conclusively compare these models in this form due to the above-mentioned limitations; of those semantic segmentation studies we compared against, our model ranks among the best performing at this task. This remains the case even while simultaneously segmenting an additional two classes that the compared models did not need to segment.

A limitation of this work is the subjectivity associated with manually labeling forest point clouds. While the majority of points can be segmented consistently, it is inevitable that mislabeled points will be present due to the ambiguity of noisy sections and the limited time that can be spent ensuring a point cloud is correctly labeled. Further to this, humans are not well suited to highly repetitive tasks, and while all possible care was taken to accurately label these point clouds during the two-week long labeling process, some minor (human) misclassifications are almost guaranteed to be present. In synthetic forest point cloud datasets, it is possible to precisely define vegetation and stem as separate categories; however, in real-world point clouds, this distinction becomes less clear. As discussed in Section 2.3, we described a continuous scale between the definitions of stem and vegetation, where stem points begin to resemble vegetation points as the noise increases/reconstruction quality decreases. As a result of this, the intent of our approach was to segment out well-reconstructed stems from poorly reconstructed stems by labeling the difficult to measure stems/branches as the vegetation class. This effect is clearly illustrated in the video provided, which shows the segmentation results of the model on five point clouds. For example, in TLS_4, a dataset with little noise, most of the stem is correctly classified as stem, while noisier and less dense point clouds such as VUX_1LR_1 and VUX_1LR_2 have comparatively more stem sections labeled as vegetation. We considered it to be preferable to misclassify stems as vegetation rather than vegetation as stems. It is preferable to miss a tree than to attempt to fit circles/cylinders to vegetation and risk overestimating the volume of the forest. This undesirable behavior was difficult to avoid entirely with this approach, but it may be possible to remove some of these vegetation-stem misclassifications during post-processing with a well-designed and robust stem fitting approach applied to the segmented stems. At the time of writing, our team is working on this problem as the next step in this project.

The idea behind combining these classes into a single multi-class segmentation model was that the CWD class was intended to make use of the proximity to the terrain class information. Due to the complex nature of the model, it is not clear if this idea was useful; however, the approach was nonetheless successful. Our deep learning approach to CWD detection differs considerably from the cylinder fitting approach used in [45] to detect fallen deadwood. An advantage of our method is the capability of identifying highly irregular, partially reconstructed, and decaying CWD rather than cylindrical CWD. Most of the CWD we are detecting in our model is ill suited to being measured with cylinder-based models, so our future work on measuring the volume of segmented CWD will approach this using mesh-based techniques.

The misclassification of some terrain points as stem points (seen in UC_UAS_AP_1 for example) appears to mostly occur when the sample box region has cropped the terrain on the edges of the point cloud or partially cropped into the terrain with the upper or lower boundary of the box vertically. These cases can change the appearance of the terrain such that even a human may have difficulty identifying it correctly. We suggest that this problem is one of context, as when the sample is seen in context (i.e., with the rest of the point cloud), it is easy for a human to identify these examples correctly as terrain, but without context, a small slice of terrain may look very similar to CWD, a branch/stem in

the air, or vegetation. Alternate sampling strategies to the box approach that could provide a greater context to the model with minimal loss in effective resolution would be a useful direction for future research to explore.

4.2. Digital Terrain Model

To quantify the performance of our DTM method, we tested our approach on point clouds and reference DTMs provided by a benchmarking study [18]. Our Root Mean Squared Errors (RMSEs) of heights relative to the reference DTMs were higher (worse) than the best algorithms tested in the benchmarking study, but they were still within a similar range as the other algorithms tested in that study. With that said, we must acknowledge that we are not truly comparing the same data, as we are measuring only the six point clouds that were made open access: a subset of the 24 point clouds in the original study. Our method had effectively 100% coverage in all point clouds while also being relatively consistent in performance amidst the variable complexity, missing data from occlusions, and steep terrain conditions of some of the point clouds. Our approach generated a smoother DTM surface than the reference DTM method, but we cannot confidently say if one method was more accurate than the other with this test. In this comparison, we are comparing our algorithm's results to another algorithm's results (with some manual intervention in the case of the reference DTM); however, we consider this comparison to be sufficient to validate our DTM generation method's efficacy.

Extracting DTMs using Pointnet and Pointnet++-based approaches has been done before on comparatively low-resolution ALS point clouds [32,41]; however, our approach differs by applying a modified Pointnet++ architecture to simultaneously extract terrain, vegetation, CWD, and stem points from a point cloud. A notable property of our DTM generation method is its robustness to noise points below the ground surface, which are common in photogrammetry datasets. This robustness emerges as a result of the segmentation model classifying the below-ground noise points as vegetation and not terrain points, allowing the DTM method to simply ignore those points.

The most significant limitation of our DTM method would be the computational cost compared to other, simpler DTM generation methods. It is more computationally expensive to segment an entire point cloud prior to generating the DTM; however, we already segment the point cloud as part of our overall point cloud analysis approach, so this is acceptable for our application. Our algorithm was capable of performing similarly to the reference DTMs of the benchmarking study with no manual intervention required, which, as stated in the benchmarking study, is difficult to achieve with a fully automatic algorithm. The priority of our work at this time is reliability and the ability to truly automate forest point cloud analysis, which leads us to the future directions of this project.

4.3. Future Research Directions

In future work on this project, we intend to use the trained model to expand our training dataset by manually correcting the minor errors made by the model and retraining/adjusting the model iteratively until the desired model performance is reached. Once the CWD segmentation performance is more reliable, there will be a need for further research to measure and validate CWD quantities against reference data. This work is part of an ongoing research effort into the development of a tool for fully automated and sensor agnostic measurement of forest point clouds. Future work by the authors will focus on the exploitation of reliable point cloud segmentation as the starting point for the extraction of detailed tree models and structural complexity metrics under diverse forest conditions and point cloud types.

While beyond the scope of our project, we also suggest that the modified Pointnet++ model we have presented is likely to be transferable to applications outside of forest mapping, particularly where the smaller original Pointnet++ model may not capture sufficient contextual information to segment the point cloud effectively. It would be interesting to explore the effect of varying the number of segmentation classes on the

overall accuracy of segmentation, as well as exploring if even larger models (allowing even more contextual information) could perform better.

5. Conclusions

In this study, we presented and evaluated a methodology for sensor agnostic semantic segmentation of high-resolution forest point clouds and a Digital Terrain Model (DTM) approach that exploits the segmented point cloud. Our semantic segmentation approach was able to achieve an overall accuracy of 95.4% relative to human labeled point clouds but with the considerable benefit of being a fully automated workflow. Our model achieved per class accuracies of 95.92% for terrain, 96.02% for vegetation, 54.98% for coarse woody debris, and 96.09% for stems. Where human operators may require several days to manually segment relatively small (20×20 m) point clouds, the presented methodology allows much larger-scale point clouds to be segmented to an almost human-level accuracy at a rate of up to several hectares per day (depending on point density) on a moderately powerful consumer grade desktop computer. Furthermore, we can now use this model to build larger-scale training datasets through an iterative process of model prediction and manual human correction of errors. Through this process, it will become faster and cheaper to generate reliable reference datasets for training and evaluation of new forest segmentation models, until errors such as those seen in the videos can be mostly overcome. Future work will see the segmentation and DTM extraction methods incorporated into a fully automated forest point cloud measurement tool, which is intended to extract structural measurements from diverse and complex point clouds from a variety of sensors and sensing techniques.

Author Contributions: Conceptualization, S.K., M.S.T. and P.T.; Data curation, S.K., S.G.A. and D.H.; Formal analysis, S.K.; Funding acquisition, M.S.T. and P.T.; Investigation, S.K., M.S.T. and P.T.; Methodology, S.K.; Project administration, S.K., M.S.T. and P.T.; Resources, M.S.T., S.G.A., D.H. and P.T.; Software, S.K.; Supervision, M.S.T. and P.T.; Validation, S.K.; Visualization, S.K.; Writing— original draft, S.K., M.S.T. and P.T.; Writing—review and editing, S.K., M.S.T., S.G.A. and P.T. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: Some restrictions apply to the availability of these data. Data obtained from Interpine Group Ltd is commercial in confidence. The TLS_1 and TLS_2 datasets were extracted from "Terrestrial laser scans—Riegl VZ400, individual tree point clouds and cylinder models, Rushworth Forest" [38] obtained through TERN AusCover (http://www.auscover.org.au, accessed on 5 October 2020). UC_UAS_AP_1, UAS_AP_1, and UAS_AP_2 are available upon request. The trained model will eventually be made available as part of a larger Python package upon release of the second paper of this project at (https://github.com/SKrisanski, accessed on 7 April 2021).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.
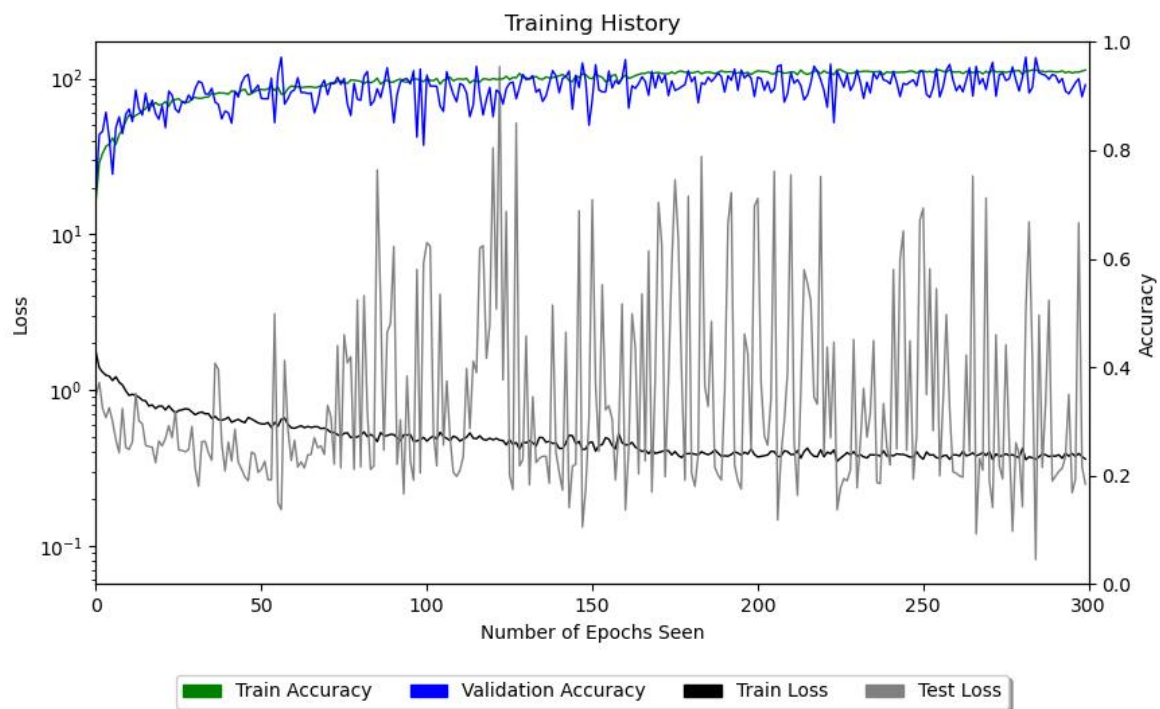
Appendix A

Figure A1. This figure presents the training history of our model, showing the overall training and validation accuracies and their losses over the 300 epochs of training.
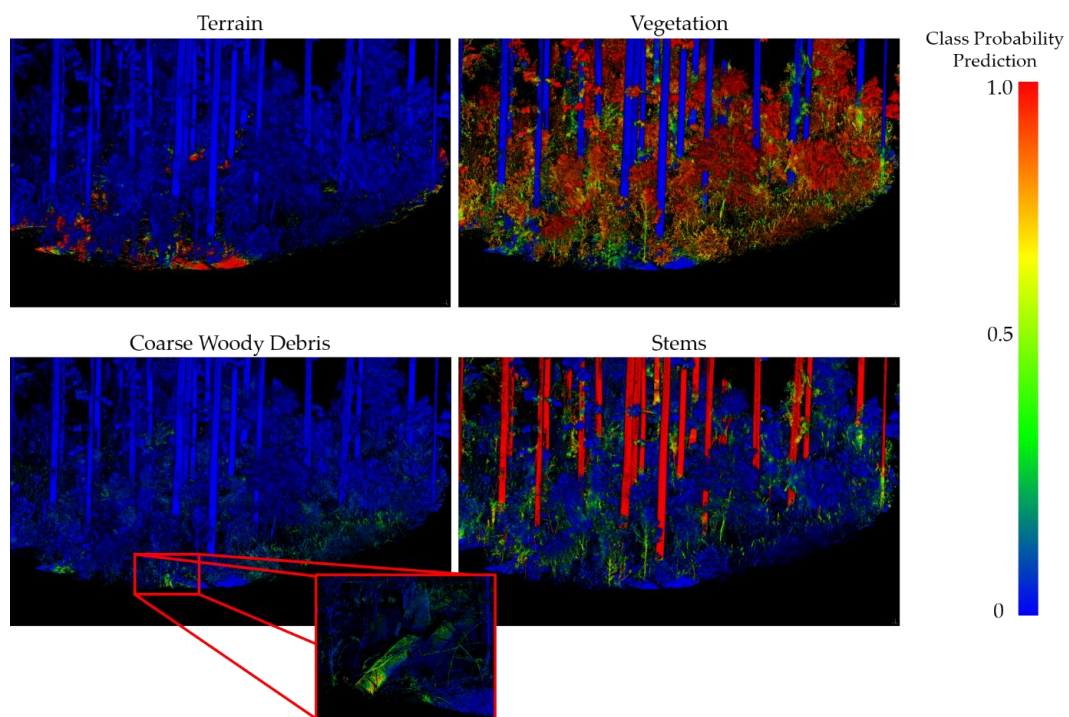

Figure A2. This figure shows the raw output from the model just prior to the argmax function (which chooses the label with the highest confidence), as shown in Figure 5. This dataset is TLS_4, which was never seen to the model during training. The terrain, vegetation, and stem classes were more confident, as per our accuracy results; however, coarse woody debris was still successfully detected in many cases.
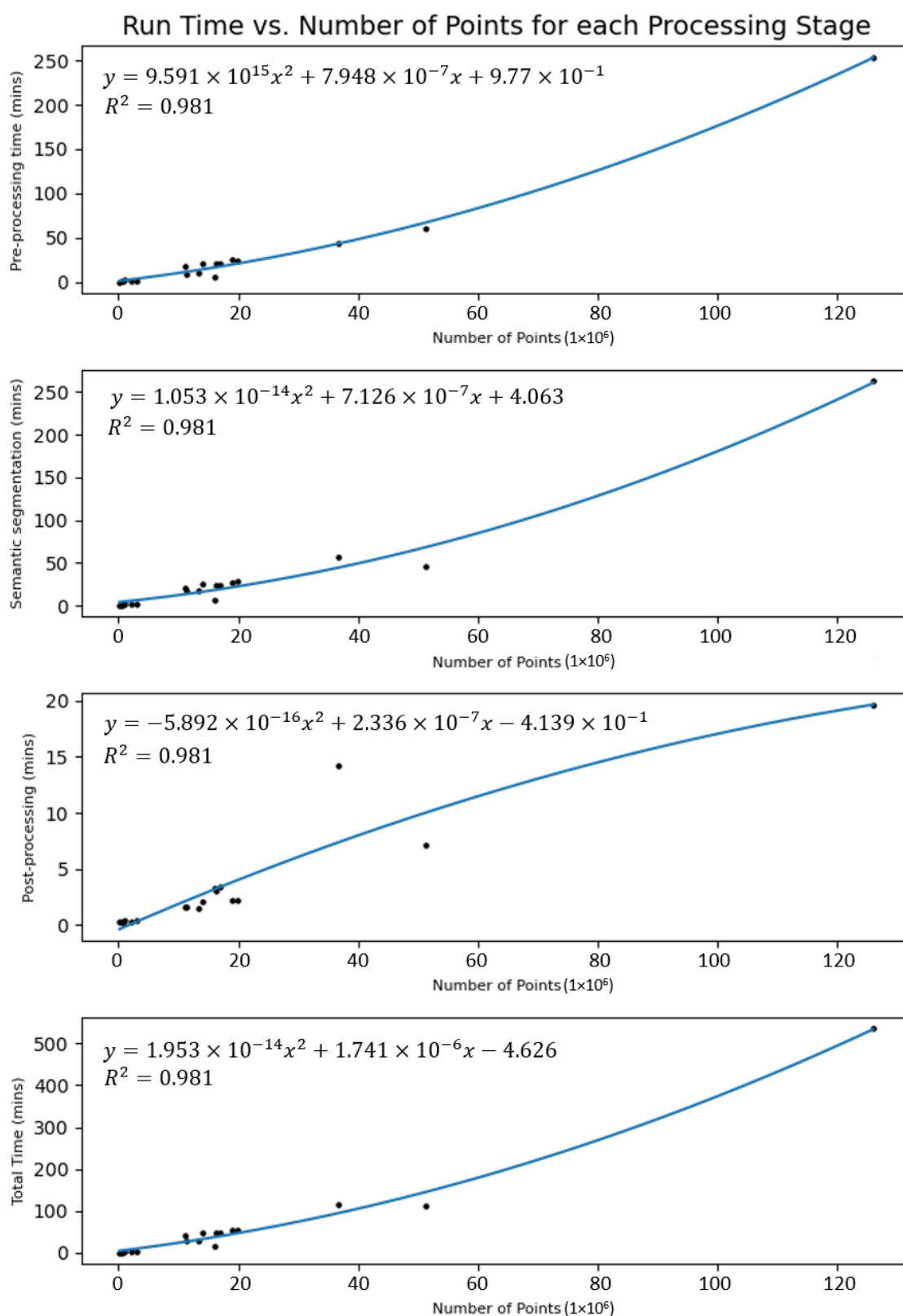
Run Time vs. Number of Points for each Processing Stage

$$y = 9.591 \times 10^{15}x^2 + 7.948 \times 10^{-7}x + 9.77 \times 10^{-1}$$
$$R^2 = 0.981$$

$$y = 1.053 \times 10^{-14}x^2 + 7.126 \times 10^{-7}x + 4.063$$
$$R^2 = 0.981$$

$$y = -5.892 \times 10^{-16}x^2 + 2.336 \times 10^{-7}x - 4.139 \times 10^{-1}$$
$$R^2 = 0.981$$

$$y = 1.953 \times 10^{-14}x^2 + 1.741 \times 10^{-6}x - 4.626$$
$$R^2 = 0.981$$

Figure A3. This figure shows the computation time of each processing step relative to the number of points (after subsampling to a 0.01 m minimum distance between points). A second-order polynomial was fitted to show the approximate trend of the data; however, the largest dataset (top right data point) exceeded the available 128 GB of RAM during the final steps of semantic segmentation, using the swap file on a solid state drive for the excess, which did slow the process.

References

1. Murphy, S.; Bi, H.; Volkova, L.; Weston, C.; Madhavan, D.; Krishnaraj, S.J.; Fairman, T.; Law, R. *Comprehensive Carbon Assessment Program (CCAP). Validating Above-Ground Carbon Estimates of Eucalypt Dominated Forest in Victoria*; Victorian Centre for Climate Change Adaptation Research (VCCCAR) and the Department of Environment and Primary Industries (DEPI): Powelltown, VIC, Australia, 2014.
2. Seidl, R.; Schelhaas, M.-J.; Rammer, W.; Verkerk, P.J. Increasing forest disturbances in Europe and their impact on carbon storage. *Nat. Clim. Chang.* **2014**, *4*, 806–810. [CrossRef] [PubMed]
3. Asner, G.P.; Mascaro, J.; Muller-Landau, H.C.; Vieilledent, G.; Vaudry, R.; Rasamoelina, M.; Hall, J.S.; van Breugel, M. A universal airborne LiDAR approach for tropical forest carbon mapping. *Oecologia* **2012**, *168*, 1147–1160. [CrossRef] [PubMed]
4. González-Olabarria, J.-R.; Rodríguez, F.; Fernández-Landa, A.; Mola-Yudego, B. Mapping fire risk in the Model Forest of Urbión (Spain) based on airborne LiDAR measurements. *For. Ecol. Manag.* **2012**, *282*, 149–156. [CrossRef]
5. Ziegler, J.P.; Hoffman, C.; Battaglia, M.; Mell, W. Spatially explicit measurements of forest structure and fire behavior following restoration treatments in dry forests. *For. Ecol. Manag.* **2017**, *386*, 1–12. [CrossRef]
6. Shugart, H.H.; Saatchi, S.; Hall, F.G. Importance of structure and its measurement in quantifying function of forest ecosystems. *J. Geophys. Res. Biogeosci.* **2010**, *115*. [CrossRef]
7. McElhinny, C.; Gibbons, P.; Brack, C. An objective and quantitative methodology for constructing an index of stand structural complexity. *For. Ecol. Manag.* **2006**, *235*, 54–71. [CrossRef]
8. McElhinny, C.; Gibbons, P.; Brack, C.; Bauhus, J. Forest and woodland stand structural complexity: Its definition and measurement. *For. Ecol. Manag.* **2005**, *218*, 1–24. [CrossRef]
9. Piermattei, L.; Karel, W.; Wang, D.; Wieser, M.; Mokroš, M.; Surový, P.; Koreň, M.; Tomaštík, J.; Pfeifer, N.; Hollaus, M. Terrestrial Structure from Motion Photogrammetry for Deriving Forest Inventory Data. *Remote Sens.* **2019**, *11*, 950. [CrossRef]
10. Mokroš, M.; Liang, X.; Surový, P.; Valent, P.; Čerňava, J.; Chudý, F.; Tunák, D.; Saloň, Š.; Merganič, J. Evaluation of Close-Range Photogrammetry Image Collection Methods for Estimating Tree Diameters. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 93. [CrossRef]
11. Krisanski, S.; Taskhiri, M.S.; Turner, P. Enhancing Methods for Under-Canopy Unmanned Aircraft System Based Photogrammetry in Complex Forests for Tree Diameter Measurement. *Remote Sens.* **2020**, *12*, 1652. [CrossRef]
12. Kuželka, K.; Surový, P. Mapping Forest Structure Using UAS inside Flight Capabilities. *Sensors* **2018**, *18*, 2245. [CrossRef]
13. Trochta, J.; Krůček, M.; Vrška, T.; Král, K. 3D Forest: An application for descriptions of three-dimensional forest structures using terrestrial LiDAR. *PLoS ONE* **2017**, *12*, e0176871. [CrossRef] [PubMed]
14. GreenValley International. LIDAR360 Comprehensive Point Cloud Post-Processing Suite. 2020. Available online: https://greenvalleyintl.com/software/lidar360/ (accessed on 10 August 2020).
15. de Conto, T.; Olofsson, K.; Görgens, E.B.; Rodriguez, L.C.E.; Almeida, G. Performance of stem denoising and stem modelling algorithms on single tree point clouds from terrestrial laser scanning. *Comput. Electron. Agric.* **2017**, *143*, 165–176. [CrossRef]
16. Piboule, A.; Krebs, M.; Esclatine, L.; Hervé, J.-C. Computree: A collaborative platform for use of terrestrial lidar in dendrometry. In Proceedings of the International IUFRO Conference MeMoWood, Nancy, France, 1–4 October 2013.
17. Koreň, M. *DendroCloud, 1.47*; Technical University in Zvolen: Zvolen, Slovakia, 2018.
18. Liang, X.; Hyyppä, J.; Kaartinen, H.; Lehtomäki, M.; Pyörälä, J.; Pfeifer, N.; Holopainen, M.; Brolly, G.; Francesco, P.; Hackenberg, J.; et al. International benchmarking of terrestrial laser scanning approaches for forest inventories. *ISPRS J. Photogramm. Remote Sens.* **2018**, *144*, 137–179. [CrossRef]
19. Calders, K.; Adams, J.; Armston, J.; Bartholomeus, H.; Bauwens, S.; Bentley, L.P.; Chave, J.; Danson, F.M.; Demol, M.; Disney, M.; et al. Terrestrial laser scanning in forest ecology: Expanding the horizon. *Remote Sens. Environ.* **2020**, *251*, 112102. [CrossRef]
20. Burt, A.; Disney, M.; Calders, K. Extracting individual trees from lidar point clouds using treeseg. *Methods Ecol. Evol.* **2018**, *10*, 438–445. [CrossRef]
21. Wang, D. Unsupervised semantic and instance segmentation of forest point clouds. *ISPRS J. Photogramm. Remote Sens.* **2020**, *165*, 86–97. [CrossRef]
22. Raumonen, P.; Åkerblom, M.; Kaasalainen, M.; Casella, E.; Calders, K.; Murphy, S. Massive-scale tree modelling from TLS data. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *2*. [CrossRef]
23. Morel, J.; Bac, A.; Kanai, T. Segmentation of unbalanced and in-homogeneous point clouds and its application to 3D scanned trees. *Vis. Comput.* **2020**. [CrossRef]
24. Digumarti, S.T.; Nieto, J.; Cadena, C.; Siegwart, R.; Beardsley, P. Automatic Segmentation of Tree Structure From Point Cloud Data. *IEEE Robot. Autom. Lett.* **2018**, *3*, 3043–3050. [CrossRef]
25. Marselis, S.M.; Yebra, M.; Jovanovic, T.; van Dijk, A.I.J.M. Deriving comprehensive forest structure information from mobile laser scanning observations using automated point cloud classification. *Environ. Model. Softw.* **2016**, *82*, 142–151. [CrossRef]
26. Windrim, L.; Bryson, M. Detection, Segmentation, and Model Fitting of Individual Tree Stems from Airborne Laser Scanning of Forests Using Deep Learning. *Remote Sens.* **2020**, *12*, 1469. [CrossRef]
27. Heinzel, J.; Huber, M.O. Detecting Tree Stems from Volumetric TLS Data in Forest Environments with Rich Understory. *Remote Sens.* **2017**, *9*, 9. [CrossRef]
28. Lalonde, J.F.; Vandapel, N.; Hebert, M. *Automatic Three-Dimensional Point Cloud Processing for Forest Inventory*; Carnegie Mellon University: Pittsburgh, PA, USA, 2006. [CrossRef]

29. Ayrey, E.; Fraver, S.; Kershaw, J.A.; Kenefic, L.S.; Hayes, D.; Weiskittel, A.R.; Roth, B.E. Layer Stacking: A Novel Algorithm for Individual Forest Tree Segmentation from LiDAR Point Clouds. *Can. J. Remote Sens.* **2017**, *43*, 16–27. [CrossRef]
30. Ni, H.; Lin, X.; Zhang, J. Classification of ALS Point Cloud with Improved Point Cloud Segmentation and Random Forests. *Remote Sens.* **2017**, *9*, 288. [CrossRef]
31. Jin, S.; Su, Y.; Zhao, X.; Hu, T.; Guo, Q. A Point-Based Fully Convolutional Neural Network for Airborne LiDAR Ground Point Filtering in Forested Environments. *IEEE J. Sel. Topics Appl. Earth Obs. Remote Sens.* **2020**, *13*, 3958–3974. [CrossRef]
32. Ayrey, E.; Hayes, D.J. The Use of Three-Dimensional Convolutional Neural Networks to Interpret LiDAR for Forest Inventory. *Remote Sens.* **2018**, *10*, 649. [CrossRef]
33. Digumarti, S.T. *Semantic Segmentation and Mapping for Natural Environments*; ETH Zurich: Zürich, Switzerland, 2019.
34. Wang, D.; Momo Takoudjou, S.; Casella, E. LeWoS: A universal leaf-wood classification method to facilitate the 3D modelling of large tropical trees using terrestrial LiDAR. *Methods Ecol. Evol.* **2020**, *11*, 376–389. [CrossRef]
35. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv* **2017**, arXiv:1706.02413.
36. Girardeau-Montaut, D. CloudCompare, v2.11.alpha. Available online: https://www.danielgm.net/cc/ (accessed on 6 October 2019).
37. Calders, K.; Newnham, G.; Burt, A.; Murphy, S.; Raumonen, P.; Herold, M.; Culvenor, D.; Avitabile, V.; Disney, M.; Armston, J.; et al. Nondestructive estimates of above-ground biomass using terrestrial laser scanning. *Methods Ecol. Evol.* **2015**, *6*, 198–208. [CrossRef]
38. Wageningen University, Netherlands; CSIRO Land and Water; Department of Geography, University College London; School of Land and Environment, University of Melbourne; Department of Mathematics, Tampere University of Technology; Environmental Sensing Systems, Melbourne; Remote Sensing Centre, Queensland Department of Science, Information Technology, Innovation and the Arts. Terrestrial Laser Scans—Riegl VZ400, Individual Tree Point Clouds and Cylinder Models, Rushworth Forest. 2014. Available online: gpv1wf_14501655e03676013s_20120504_aa2f0_r06cd_p300khz_x01.las (accessed on 5 October 2020). [CrossRef]
39. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *arXiv* **2016**, arXiv:1612.00593.
40. Fey, M.; Lenssen, J.E. Fast graph representation learning with PyTorch Geometric. *arXiv* **2019**, arXiv:1903.02428.
41. Harris, C.R.; Millman, K.J.; van der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J.; et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef]
42. Pedregosa, F.V.G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
43. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef]
44. TerraSolid. TerraScan; TerraSolid. 2018. Available online: https://terrasolid.com/products/terrascan/ (accessed on 7 April 2021).
45. Yrttimaa, T.; Saarinen, N.; Luoma, V.; Tanhuanpää, T.; Kankare, V.; Liang, X.; Hyyppä, J.; Holopainen, M.; Vastaranta, M. Detecting and characterizing downed dead wood using terrestrial laser scanning. *ISPRS J. Photogramm. Remote Sens.* **2019**, *151*, 76–90. [CrossRef]