COSC470 Research Project Report

Semantic Segmentation of Trees in Aerial Point Clouds of New Zealand Urban Areas

---

Grant Wong

Supervisors: Dr. Oliver Batchelor,
Dr. Richard Green

Department of Computer Science and Software Engineering
University of Canterbury
Christchurch, New Zealand

October 2022

Abstract

Urban trees are critical in mitigating many negative impacts of urban areas, and understanding the extent to which they do this requires accurately segmenting and identifying the precise area of tree regions in a city. For urban areas represented by aerial point cloud data, existing methods rely on datasets with eight or nine classification classes, with the segmentation performance on highly imbalanced datasets with only two classes yet unexplored. We develop a pre-processing and data loading pipeline and use the SoftGroup sparse convolutional neural network (CNN) library to segment a custom urban point cloud dataset, and we explore the effects of different training parameter values on the segmentation accuracy. From our experiments, we are able to achieve a maximum average mIOU percentage of 72.8% on our test dataset.

Contents

1 Introduction

Urban trees hold a critical role in improving the livelihoods of the people who live amongst them by mitigating the negative impacts of urban areas. They mitigate the poorer air quality encountered in urban regions by removing pollutants from the atmosphere [1]. Trees in urban areas also serve as a carbon sink, with 14% of the carbon sequestration from US forests from trees in urban forests alone [2]. Furthermore, they reduce urban temperatures by mitigating the 'urban heat island' (UHI) effect. This effect is where a high density of artificial urban structures increases the heat-absorbing surfaces of an urban area, causing human discomfort and increased energy consumption [3]. Understanding the extent that urban trees provide these benefits requires accurately identifying the total precise area and location of the tree regions in a city. Examples include estimating the volume of carbon stored by urban trees [4] and modelling the degree of temperature alleviation that urban trees provide in the summer [5].

Determining the area and precise boundaries of tree regions is also critical as a prerequisite for individual tree crown delineation (ITC delineation), which is the delineating of tree canopies in a given tree region. ITC delineation allows for further analysis on forest inventory data, such as estimating the number of trees, classifying tree species [6], and modelling tree growth. In New Zealand, this can be used to assess the degree of degradation and restoration of indigenous forests over time. Additionally, this application is useful for monitoring ecological threats, such as tree dieback from myrtle rust [7] and the pathogen *Phytophthora agathidicida* [8].

There have been many existing approaches to identifying and segmenting trees with satellite imagery [9, 10, 11, 12]. Satellite imagery is significantly less memory intensive than point clouds, which are three-dimensional collections of points that are each assigned colour values. However, there is a limitation with using satellite imagery: raster images cannot accurately represent any overlap in points, such as a

building that partially obscures a tree (and vice versa). Additionally, point clouds are more flexible for several further applications of semantic segmentation, including the accurate reconstruction of the 3D shape of individual trees. Because of this, we focus on aerial views of urban areas in the form of point clouds.

In our report, we examine the performance of using a sparse convolutional neural network (CNN) on these 3D point clouds to segment urban areas into tree and non-tree regions. Here, we develop a pipeline for pre-processing and segmenting point clouds of an urban area. Additionally, we explore the use of different training parameters, and we analyse and compare the resulting effects of different parameter values on the segmentation output.

The organisation of our report is as follows. In Chapter 2, we discuss the concepts directly relevant to our project. This includes light detection and ranging (LiDAR) and sparse convolutional neural networks (sparse CNNs). We also give an overview of the related work in the segmentation of urban areas, both in the form of satellite imagery and 3D point clouds. Chapter 3 provides an overview of our proposed method, as well as an overview of the dataset and convolutional neural network library used. We discuss the results from our experiments in Chapter 4 and Chapter 5. Finally, we conclude our report in Chapter 6.

2 Background

2.1 Light Detection and Ranging (LiDAR)

Light detection and ranging (LiDAR) is a remote sensing method that uses infrared laser pulses to measure the elevation and 3D shape of a landscape. LiDAR is popular as a tool for ecosystem monitoring [13], which may involve mapping the structure of forest canopies and classifying tree species. A prominent strength of LiDAR is its performance in high-resolution, large-scale measurement of 3D landscapes [14]. Data collected from LiDAR sensors are usually in the form of sets of 3D points that give a 3D model of a scene. These sets are called point clouds [15], and they may be dense or sparse depending on the sensor and location. A common approach is to combine these points with colour information from digital images, especially in autonomous driving [16, 17].

LiDAR was first used for tree data in the 1980s, when Nelson et al. [18] used a pulsed laser system to determine the heights of tree canopies, and the graphs of canopy heights were used to analyse the health of forests in Pennsylvania. There were serious limitations with the LiDAR technology used at the time, notably that the laser system was restricted to heights between 150–460 metres, and it could only operate during the day.

Since then, advancements in LiDAR technology resulted in a gradual decrease in cost. By the late 1990s, research into using LiDAR for semantic and instance segmentation emerged, typically in the form of canopy height models [19, 20]. Canopy height models are described in more detail in subsection 2.4.1. However, studies in individual tree crown delineation continue to predominately use multispectral and hyperspectral imagery as opposed to LiDAR data, due to the former being cheaper to collect [21, 20].

From the early 2010s, as the cost of collecting LiDAR data continued to decrease,

tree delineation algorithms using LiDAR depth information became more popular. This was because the depth information in LiDAR data was useful in allowing researchers to avoid the challenges of determining tree crown locations from visual appearance alone. The point clouds from LiDAR data also allowed one to theoretically achieve higher accuracy than with raster images, because understorey trees (trees under the canopies of other trees) and small trees could be detected [22].

LiDAR data are used for both instance segmentation and semantic segmentation. For both instance and semantic segmentation, there are two common methods of using the LiDAR data: the first approach is to convert the point cloud data into a height model, which is a raster image where the colour or brightness of each pixel represents the height at the corresponding location. An alternative approach is to use the LiDAR point cloud data directly for 3D instance segmentation.

2.2 Convolutional neural networks

Artificial neural networks (commonly abbreviated to 'neural networks') are systems that consist of nodes that roughly model the tightly interconnected neurons in the human brain. These empower the development of systems that can complete tasks that require identifying complex patterns, particularly in finding the boundaries of objects in a scene, and in classifying what object or label to which something belongs. These possibilities have made neural networks crucial for applications such as facial recognition, identifying arbitrary objects in a scene, and playing the board games Go and chess. An example of a neural network is shown in Figure 1; note that we only discuss feed-forward neural networks, which are neural networks that do not contain cycles.

A weakness of traditional feed-forward neural networks is that the nodes are strongly interconnected. This is problematic when the input is of more than one dimension, as the number of connections required increases in a polynomial manner. Even in two dimensions, an image of dimensions 300×300 pixels and three colour channels will require $300 \times 300 \times 3$ connections. In addition, the neural network is prone to overfitting, as it will identify patterns only in specific locations of the input. For example, for an image dataset with trees in specific locations of the image, a traditional neural network may not be able to successfully identify the trees, should

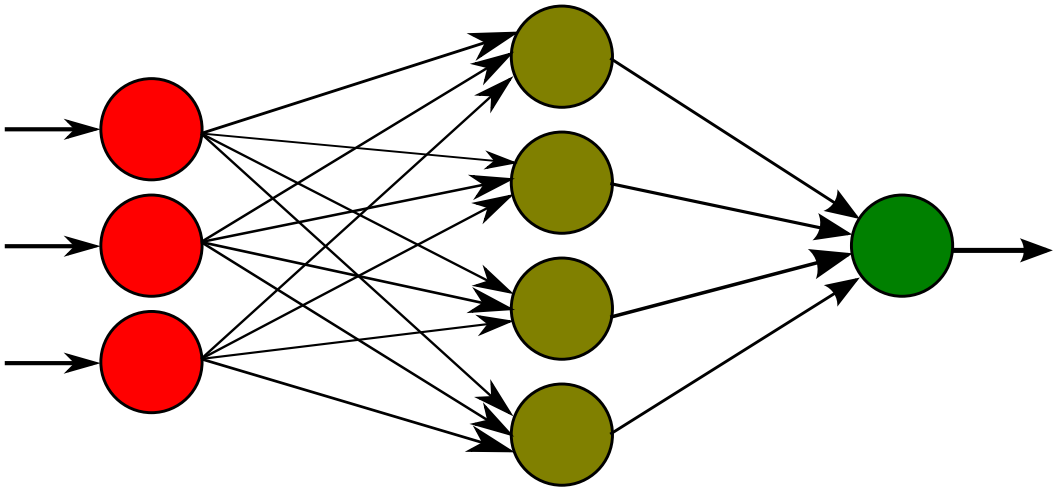

Figure 1: Example of a feed-forward neural network. The red nodes are the input nodes of the neural network, which represent a multivariable input from the problem we wish to solve. The green node is the output node, which represents the final result. Between the input and output layers are the yellow nodes, which form the hidden layer(s) and are used to identify complex patterns that appear in the input.

they appear in locations different to the training data. To overcome both of these weaknesses, convolutional neural networks were devised.

Convolutional neural networks (CNNs) are a form of neural network that allows the identification of features in 2D and 3D data, such as images, videos, and point clouds. A notable innovation in this type of network is the use of the *convolution* mathematical operation, where a small, multi-dimensional kernel is multiplied across each small section of an input. These convolutions allow the neural network to recognise features in the images with a useful property named *translational equivariance* – if an object appears in a different part of the image than expected, it will still appear as an output by the neural network, with the only difference being its location in the output.

A typical convolutional neural network comprises three parts, which may be chained and repeated several times to form a full network: a *convolution layer*, a *pooling layer*, and a *fully-connected layer* [23]. The convolution layer performs the convolution mathematical operation over the input, producing a two or three-dimensional feature map. The pooling layer reduces the size of the feature map to extract the most prominent features. For example, in a two-dimensional feature map, the pooling

layer may extract the maximum values for each local neighbourhood. Finally, the fully-connected layer computes the maximum of the input neurons from the previous layer.

CNNs are responsible for the performance breakthroughs in image classification and image segmentation [24]. Additionally, CNNs have been used to achieve state-of-the-art results in fields such as speech recognition, object tracking, and text recognition. In these diverse applications, many variants of CNNs have been developed to increase performance in specific contexts. These include customised loss functions, novel data augmentation methods, and changes to the pooling layer [25]. One such variant that is directly relevant to our research is a sparse convolutional neural network, which we discuss below.

2.3 Sparse convolutional neural networks (sparse CNNs)

Sparse convolutional neural networks (sparse CNNs) are one such CNN variant that has emerged in the late 2010s to allow classifying 3D point clouds in a more efficient way. This is possible by introducing two variants of the convolution operation – the sparse convolution (SC) and the submanifold sparse convolution (SSC) – that ignore areas in the 3D input that do not contain data [26]. Figure 2 contains an example demonstrating the submanifold sparse convolution operation.

Traditionally, Sparse CNNs are most commonly applied in two contexts. One such context is autonomous driving [27], where CNNs are used to identify or segment various objects in a point cloud of a road. This data can be collected from modified cars equipped with a LiDAR scanner and/or a stereo camera, which is the case with the KITTI dataset [28]. The objects identified may include cars, pedestrians, trees, and so on. Another context is indoor scenes [29], where there is a focus on segmenting objects such as furniture and indoor appliances. Here, the ScanNet dataset [29] has been released to serve as a benchmark. For both contexts, researchers such as Li et al. [30] and Wang et al. [31] demonstrate successful results when performing semantic segmentation using sparse CNNs.

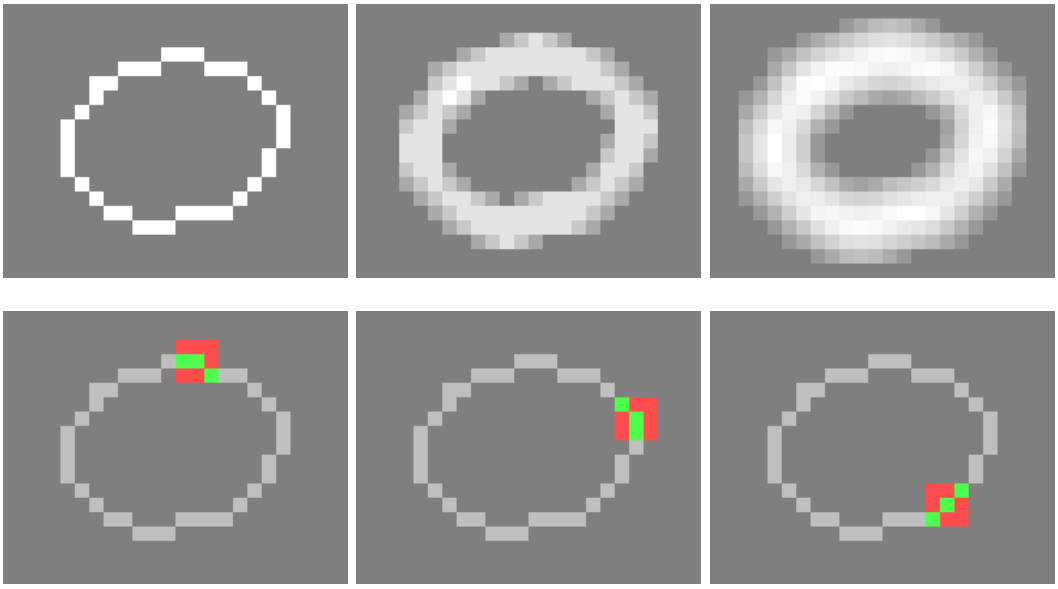

Figure 2: Regular convolution vs submanifold sparse convolution. Top images: the result of applying a regular convolution multiple times. Repeated convolutions reduce the sparsity of the features drastically. Bottom images: applying a submanifold sparse convolution, one of the operations in sparse convolution, does not change the locations in the input that do not contain data (red areas). Only the green areas are affected. Figure from Graham et al. [26]

2.4 Related work

2.4.1 Semantic segmentation of tree regions

Existing research into semantic segmentation for urban trees typically involves a combination of 2D satellite imagery and a canopy height model (CHM), as opposed to point clouds. A canopy height model (CHM, also known as a digital canopy model) comprises a 2D pixel grid where each pixel contains the height from the ground to the top of the canopy at that location. This model is often found by subtracting the digital terrain model (DTM) from the digital surface model (DSM) [32]. These are two raster images derived from the LiDAR point cloud data that show the elevation of the ground and the elevation of the tree canopies, respectively. From this raster data, systems such as neural networks can be used to segment tree regions in an urban area.

Comparisons to other works are difficult due to most existing research focusing on the instance segmentation of urban trees in satellite imagery [33, 34, 11, 35] – in other words, individual tree crown (ITC) delineation – as opposed to the semantic segmentation of tree regions. However, in some cases, the segmentation of the tree regions is also evaluated alongside the accuracy of the ITC delineation. For example, Yang et al. [35] achieve a 82.3% intersection over union (IoU) with their test dataset of satellite images of New York's Central Park. Hao et al. [34] achieve an IoU of 76.53%–91.27% depending on the model and the data used, with the highest IoU achieved by using a CHM and a normalised difference vegetation index (NDVI) [36] derived from satellite imagery.

2.4.2 Semantic segmentation of urban point clouds

Performing semantic segmentation of point clouds for urban areas has gained increasing interest by researchers in the past five years. This rise in popularity is likely made possible by the availability of 3D airborne datasets of urban areas, which provide a means with which researchers can evaluate their CNN models for 3D semantic segmentation. These are the Vaihingen 3D semantic dataset from the International Society for Photogrammetry and Remote Sensing (ISPRS), and the Dayton Annotated Laser Earth Scan (DALES) dataset from the University of Dayton. These datasets do not focus on solely distinguishing between tree and non-tree regions; rather, they contain multiple classes (for example: ground, power lines, buildings) of which 'tree' is but one class.

Recent research into semantic segmentation of aerial point clouds rely on neural networks of various types. This includes Wen et al. [37], who present a *global-local graph attention convolution neural network* (GACNN) that uses two types of custom modules to analyse the spatial positions of the 3D points, achieving a 83.2% overall accuracy across nine segmentation classes and correctly predicting 80.2% of tree points. Yang et al. [38] also achieve a similar accuracy of 85.1% by iteratively learning features across network layers. Rim et al. [39] opt for a variant of the PointNet++ architecture instead of a convolutional neural network, thereby achieving a lower overall accuracy of 76.43% with the DALES dataset.

However, a significant difference with the above approaches is that they use *balanced*

datasets, meaning that the numbers of points belonging to each classification class in a dataset were roughly equal. Additionally, the datasets in use contain a large number of classes, which would reduce any data imbalance between classes. For example, the DALES dataset contains eight classes, and the Vaihingen dataset contains nine.

3 Approach

The goal of our research was twofold: Firstly, we aimed to examine the segmentation accuracy when using a sparse CNN on an *imbalanced* urban point cloud dataset consisting of two classes. The dataset of interest was of the Wellington urban region, and the two classes represented the tree and non-tree regions of the dataset. Secondly, we aimed to compare the accuracy to an equivalent approach based on a Mask R-CNN and 2D raster imagery. To this end, we applied several steps: we chose and modified an existing CNN architecture named SoftGroup to perform the semantic segmentation on our dataset. These steps are described in more detail below.

3.1 Procurement and pre-processing of hand-segmented dataset

Figure 3: An example of a tile in the Wellington region, comprising 3D points and RGB colour values.

For our research, we had access to LiDAR-RGB data for six aerial scenes of the Wellington region, ranging from heavily forested rural areas to the relatively tree-sparse urban suburbs. LiDAR-RGB is a form of 3D point cloud data that comprised a combination of Light Detection and Ranging (LiDAR), which provided height information in the form of 3D points, and RGB colour values for each of those points. Each scene was approximately 1000 metres wide and 600 metres high. Due to their rectangular and tile-like nature, these scenes are called 'tiles' in this report. These tiles were ultimately sourced from Land Information New Zealand's website at https://data.linz.govt.nz/.

In order for this data to be useful as training and validation data for semantic segmentation, the points that constituted the trees needed to be segmented from the points that constituted non-tree areas, such as buildings and grass. The trees in each of the six tiles were segmented by hand by the researchers at Manaaki Whenua. Each

tile consisted of roughly 400–1100 point cloud files representing each of the trees in the area, as well as another point cloud file representing all of the non-tree points in the tile.

Using this dataset as input, we created a pre-processing and data loading script to collate and convert these point cloud files to SoftGroup's custom input format. This was because there were many differences between the dataset from Manaaki Whenua (which we call the 'tree dataset') and the datasets for which SoftGroup was designed, which included an dataset of 3D indoor scenes named ScanNet [29]. The tree dataset contained roughly 5 million points per tile, whereas a typical ScanNet tile contained approximately 200,000 points; additionally, the coordinates of the tiles from the tree dataset were approximately 80 times larger than those of ScanNet. To account for the first difference, our pre-processing script also split each tile into 25 subtiles. We found that splitting the tiles in this way was also critical in overcoming memory constraints with our computer, as our computer could not process more than 1 million points at a time. The next few differences were handled by adjusting the training configuration parameters, which involved trial and error and included the batch size, scale, and maximum number of points to process.

The tree dataset was also notable for only containing two classes, and there was a significant data imbalance between these two classes; in other words, one class appeared more frequently in the tree dataset than the other. Each point in a subtile was classified as either 'not tree', which represented points that belonged to buildings, grass, concrete, and so on; or 'tree', which represented points that belonged to trees and their canopies. In contrast, a ScanNet tile contained 20 semantic classes. A subtile, which contained an average of roughly 236000 points, contained roughly 61000 points (25%) which belonged to the 'tree' class, and 175000 points (75%) which belonged to the 'not tree' class. Figure 4 shows an example of the data imbalance between the 'tree' and 'not tree' classes. For comparison, the most frequently-occurring class in the DALES dataset (the 'ground' class) was present for only 50.1% of points.

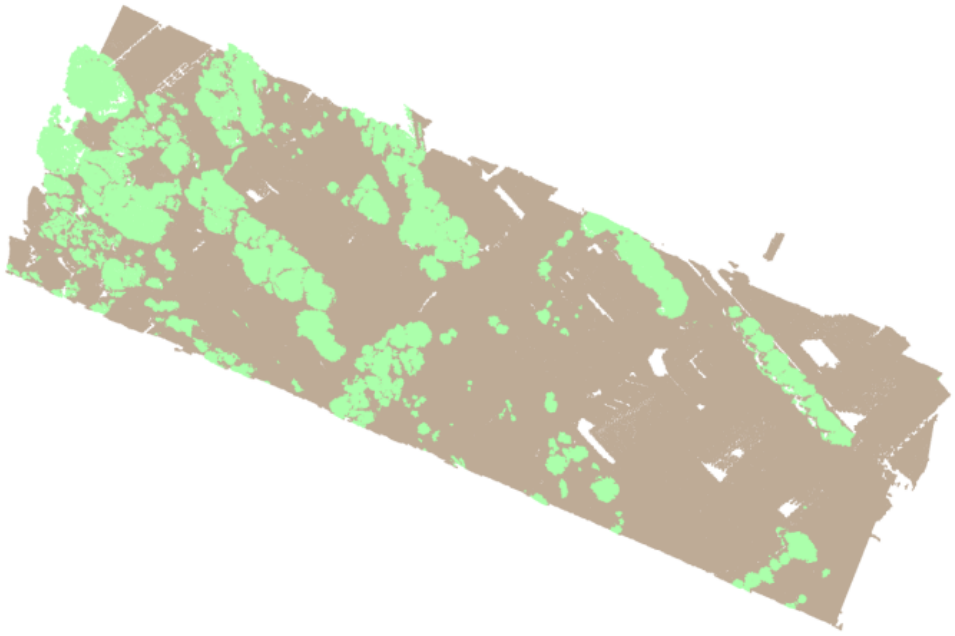

Figure 4: An example of the data imbalance between the 'not tree' and 'tree' classes, shown in brown and green, respectively. Here, the 'not tree' class appears much more frequently.

3.2 Semantic segmentation

An existing library named SoftGroup, designed by Vu et al. [40], was selected as the basis for the semantic segmentation. We extended this library to support and work well for our dataset. SoftGroup was a 3D instance segmentation library released in March 2022 for several indoor 3D point cloud datasets, notably ScanNet [29] and the Stanford 3D Indoor Scene Dataset (S3DIS) [41]. Its source code was available and licensed under the MIT License at https://github.com/thangvubk/SoftGroup. Additionally, our modifications are available at https://github.com/dddlr/softgroup.

Although SoftGroup was a library originally designed for instance segmentation of 3D scenes, a slightly different research problem, it could nonetheless directly be used for semantic segmentation due to it being an extension of grouping-based instance segmentation libraries. Grouping-based instance segmentation libraries, such as HAIS [42] and PointGroup [43], were bottom-up pipelines that relied on

two stages: a semantic stage that produced semantic predictions for each point (i.e. semantic segmentation), and a second stage that grouped points of similar features into instances. By only enabling the first stage, one could perform semantic segmentation on 3D point clouds.

This library was chosen for this project due to the extensibility of the source code for our research, and the detailed documentation regarding the dataset preprocessing, configuration parameters, and the visualisation of results. As our research project relied on a point cloud dataset that had not been previously used for 3D semantic segmentation, this allowed us to create custom preprocessing code, debug the library's backbone network, and adjust the semantic segmentation parameters. Vu et al. [40] had also provided examples for multiple indoor and outdoor datasets: point cloud datasets of indoor scenes, such as ScanNet [29] and the Stanford 3D Indoor Scene Dataset (S3DIS) [41]; as well as the 2D and 3D point cloud dataset STPLS3D [44] and SemanticKITTI [45]. This demonstrated that it was likely possible to adapt and extend the library to support our dataset.

A second reason that SoftGroup was chosen was that it was also highly performant for semantic segmentation, achieving an average mIOU percentage of 71.8% on the ScanNet dataset. For semantic segmentation, the goal of our project, SoftGroup achieved a good inference time of 106 ms in our experiments for the ScanNet dataset. This was faster than the fastest performing library discussed by Wen et al. [37], which gave an inference time of 140 ms.

3.3 Evaluation method

To allow comparisons between each model that we considered, we used the *Mean Intersection over Union* (mIOU) metric, which was a common evaluation criterion used in semantic segmentation [46]. The equations we present below are based on those from Rim et al. [39]. For each of the two classes ('not tree' and 'tree'), we found the *class-wise mIOU* as defined in Equation 1:

$$mIoU_c = \frac{TP_c}{TP_c + FP_c + FN_c} \tag{1}$$

where TP, FP, and FN represent the number of true positive points, false positive

points, and false negative points across the test dataset, respectively. c represents the cth class. This may also be described as the intersection of two sets divided by the union of two sets A and B. We defined A as the ground truth from the test dataset and B as the predicted segmentation from one of our models, both filtered to contain only points that were classified as a certain class.

We also calculated the average of the two mIOU values to derive the *average class-wise mIoU*, defined in Equation 2.

$$mIOU = \frac{1}{C}\sum_{n=1}^{C} mIOU_c \tag{2}$$

where C represents the total number of classes. In the tables in Chapter 4, we refer to the two class-wise mIOU values and the average class-wise mIOU collectively as *class-wise mIOU*.

Additionally, we measured the *overall accuracy* by dividing the number of correctly classified points by the total number of points. This is defined in Equation 3, where N represents the total number of points in the test dataset. These are listed as 'Accuracy' in the tables below.

$$Acc = \frac{1}{N}\sum_{n=1}^{C} TP_c \tag{3}$$

4 Experiments

To maximise and evaluate the performance of our approach to semantic segmentation, we performed several experiments, modifying a single parameter value each time, to determine the effects of different parameter values on the segmentation accuracy. Below is a list of the parameters that we found, through trial and error, to substantially affect the results. These parameters are discussed in more detail in section 4.2.

- Semantic weight: whether a weight was applied to each semantic class ('not tree' and 'tree') in the cross-entropy loss function.
- Height axis scaling: whether the tree dataset was scaled in the direction of the height, and by what amount.
- Voxel size: The size of each voxel, when the tree dataset was passed through SoftGroup's voxelisation step.

Table 1 contains a summary of the training parameters used for all of the experiments:

Parameter	Value
Batch size for training	4
Number of epochs	48
Iterations per subtile	4

Table 1: A list of training parameters used for all experiments.

4.1 Setup

The computer used to perform these experiments was a 64-bit Ubuntu 18.04 computer with 125 GiB RAM, an Intel Xeon E5-2683 CPU at 2.10 GHz, and a Tesla V100-

PCIE-16GB GPU. The SoftGroup library was run with Python 3.7, CUDA Toolkit 11.3, and PyTorch 1.11. On this computer, training each model took an average of 20 minutes to complete, and testing the model took approximately 20 seconds to complete.

As discussed in section 3.1, each of the six original tiles that constituted the tree dataset was split into 25 subtiles as part of the pre-processing stage, resulting in a total of 150 subtiles. This was done by dividing each tile spatially into five regions in the x direction and five regions in the z direction. This step was necessary for two reasons: to convert the data into a form more similar to the ScanNet data supported by SoftGroup, and additionally to overcome memory constraints in the computers at the University of Canterbury.

Out of the 150 subtiles in the tree dataset, 60 were placed in the training set, 60 were placed in the validation set, and the remaining 30 were placed in the test set. More data was placed in the training and validation sets to maximise the number of different subtiles to which the model would be exposed during the training stage. Additionally, the allocation was performed such that each of the original tiles would be equally represented across all of the sets, in order to maximise the similarity of the training set to the validation and test sets.

The inference time per scan, when testing the model, was comparable to recent segmentation approaches [37, 39]. However, this was subject to random variation, with the inference time for our models ranging from 96.4 ms to 122.7 ms. We did not identify any specific models in our experiments where the inference time was consistently higher or lower than the average.

4.2 Results

For each parameter that we examined, we used a default model to serve as an additional point of comparison. Table 2 lists the parameters that were used for this model.

Parameter	Value
Semantic weight	Not applied
z axis scale factor	1
Voxel size	1

Table 2: Summary of the parameters that were adjusted in our experiments, and their default values.

4.2.1 Cross-entropy loss weights

In both SoftGroup [40] and in our setup, the cross-entropy loss function was used to learn to output semantic scores for our models. These semantic scores were then used to classify points as not trees or as trees. This cross-entropy loss function optionally accepted weights for each semantic class ('not tree', 'tree'), which could be used to improve the segmentation of the 'tree' class in our imbalanced dataset. We first explored what impact applying these weights had on the accuracy of the semantic segmentation.

These weights w_i for each class i were selected using the equation shown in Equation 4. Here, s represents the total number of points across the tree dataset, and $s(i)$ represents the number of points across all datasets with class i. c here equals 2, which is the number of classes. This resulted in a higher weight for the 'tree' class, which was significantly less frequent (an average of roughly 61000 points per subtile) than the 'not tree' class (an average of 175000 points per subtile). The resulting weights from this equation were 0.714 for the 'not tree' class and 2.049 for the 'tree' class.

$$w_i = \frac{s}{c * s(i)} \tag{4}$$

According to Table 3, applying weights to the cross-entropy loss function resulted in a higher mIOU percentage of 54.9% for 'tree' points, indicating that the resulting model was more reliable at segmenting tree regions in the test dataset. However, correct classification of 'not tree' points was 1.2% lower. Due to the data imbalance between the points classified as 'not tree' and those classified as 'tree', the accuracy was therefore lower by 0.8% as well.

Semantic weight	Class-wise mIOU (%)			Accuracy (%)
	Not tree	Tree	Average	
Not applied (default)	91.6	47.8	69.7	92.2
Applied	90.4	54.9	72.6	91.4

Table 3: Results of modifying the semantic weights.

4.2.2 Height axis scaling

The second area explored in our experiments was scaling the height axis. This involved scaling the axis in the tree dataset which represented the height, before the dataset was used for the training, validation, and testing of a model. In the tree dataset, this was the z axis. This scaling was necessary because of a consequence of the LiDAR scanning used to create the dataset: the 3D points were concentrated in layers spread along the z axis. Additionally, these layers were much farther apart than the area of the tree in the xy plane. The 'layering' effect can be seen in Figure 5. Both of these factors at times resulted in a problem: undesirable output where only some of these height layers in a particular region were classified as a 'tree', and other layers were classified as 'not tree'.

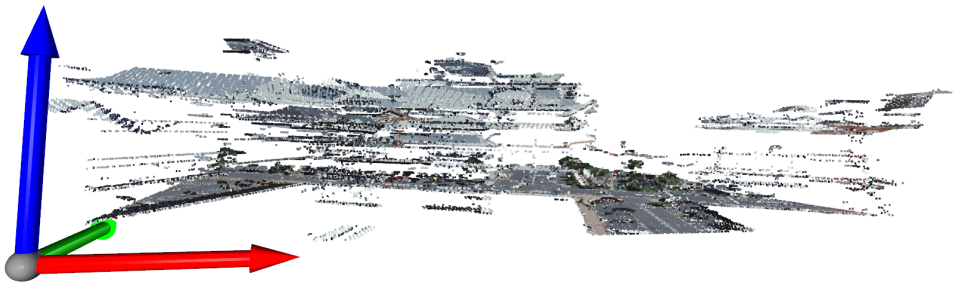

Figure 5: Example of the 'layering' effect. Points in the dataset are concentrated in distinct layers in the z axis (shown here as a blue arrow).

In an effort to mitigate this problem, the z axis of the tree dataset coordinates was scaled in three different ways: by 1/2, 1/4, and 1/8. Additionally, we ran an experiment where the z axis was set to 0. This was equivalent to projecting the 3D

tree dataset on to a two-dimensional plane.

z axis scale factor	Class-wise mIOU (%)			Accuracy (%)
	Not tree	Tree	Average	
1 (default)	91.6	47.8	69.7	92.2
1/2	91.5	47.3	69.4	92.1
1/4	91.1	45.0	68.0	91.7
1/8	90.7	42.1	66.4	91.3
0	89.9	35.5	62.7	90.4

Table 4: Results of modifying the z scale factor.

We had expected the class-wise mIOU and accuracy percentages to increase as the z axis scale factor decreased, as the height layers in the tree dataset would be more likely detected as belonging to one region. However, the results, as shown in Table 4, showed the opposite effect. A z axis scale factor of 1/2 resulted in a slightly lower average class-wise mIOU of 69.4% (from 69.7%), and a slightly lower accuracy of 92.1% (from 92.2%). Lower z axis scale factors resulted in the mIOU percentages for both points classified as 'not tree' and points classified as 'tree' decreasing. The mIOU for 'not tree' points decreased by 0.4–0.8% as the z axis scale factor was divided by 2, and the mIOU for 'tree' points decreased by a higher rate of 1–2%. This suggested that reliable detection of the 'tree' class in our dataset was perhaps reliant on the 3D shape of the tree regions, and not only the colour of the points.

4.2.3 Voxel size

Another area we explored was adjusting the voxel size. In the SoftGroup library [40], the tree dataset was voxelised. In other words, the points in the tree dataset were converted into a regular 3D grid of voxels, on which 3D sparse convolutions could be performed in the training stage. In this voxelisation step, the size of each voxel could be adjusted depending on the dataset. The desired voxel size was a tradeoff: in theory, a lower voxel size was more memory intensive and required more processing power, as there were more voxels to train and classify. At the same time, a lower voxel size would also increase the semantic segmentation accuracy, as the resulting segmentation would be more precise [47].

This was mostly confirmed by our results in Table 5. Decreasing the voxel size resulted in a higher class-wise mIOU percentage for the 'tree' class, from 47.8% in the default model (scale = 1) to 53.2% when scale = 8. There were also slight variations in the 'not tree' class-wise mIOU percentage and the accuracy percentage by 0.4–0.5%. The decreased 'not tree' class mIOU and accuracy for 1/12 m was an outlier – this may be due to random variation.

Voxel size (m)	Class-wise mIOU (%)			Accuracy (%)
	Not tree	Tree	Average	
1 (default)	91.6	47.8	69.7	92.2
1/2	91.5	49.9	70.7	92.2
1/4	91.9	52.0	72.0	92.6
1/8	92.0	53.2	72.6	92.7
1/12	91.4	54.1	72.8	92.2

Table 5: Results of modifying the voxelisation scale factor.

Interestingly, the time taken to train the model was not significantly affected by the voxel size, and all models listed in Table 5 took approximately 20 minutes to train. However, the memory consumption did increase significantly; we were not able to decrease the voxel size to any value lower than 1/12 m due to memory constraints on our computer.

4.2.4 Comparison to Mask R-CNN

To supplement our findings, we compared the semantic segmentation output of our trained models to that of another neural network, a Mask R-CNN (region-based convolutional neural network). A Mask R-CNN differs from a sparse CNN in that it is designed for instance segmentation of 2D satellite imagery, not 3D point clouds. In this case, the Mask R-CNN was trained on satellite imagery corresponding to the same areas of the Wellington region. The 3D points in the tree dataset were then classified, depending on the location to which the point corresponded in the Mask R-CNN output. The resulting segmentation outputs from this Mask R-CNN were kindly provided to us by Manaaki Whenua, and these were available for three of the six aerial scenes that were in the tree dataset.

Since we were investigating semantic segmentation, we extracted the semantic class for each point in the data. We then evaluated the resulting semantic output in a similar manner to the semantic outputs from our trained models, with one difference: when evaluating the results in Table 6, we only used 15 subtiles of the test dataset, instead of all 30 subtiles. This was because the Mask R-CNN segmentation outputs were available for only half of the subtiles in the test dataset. This evaluation was also performed on the optimal model found through our experiments, which was the model with the 1/12 m voxel size from subsection 4.2.3. We named this 'Our model' in Table 6.

According to Table 6, our sparse CNN model did not perform as well as the Mask R-CNN, when tested against the data for which Mask R-CNN output was available. Our model achieved an average mIOU that was lower by 10.4%. The difference in the average mIOU was mostly due to the mIOU for the 'tree' class: in our model, the mIOU was 15.9% lower than the corresponding mIOU in the Mask R-CNN output. Additionally, the accuracy of our model was 4.5% lower than that of the Mask R-CNN output. Figure 6 shows a visual comparison of the segmentation output between the ground truth, the output from our model, and the Mask R-CNN output.

Name	Class-wise mIOU (%)			Accuracy (%)
	Not tree	Tree	Average	
Our model	91.4	54.1	72.8	92.2
Mask R-CNN	96.5	70.0	83.2	96.7

Table 6: Comparison of our best performing model to a Mask R-CNN. Note that unlike the other tables in this section, the class-wise mIOU and accuracy percentages were calculated from only half of the test dataset.

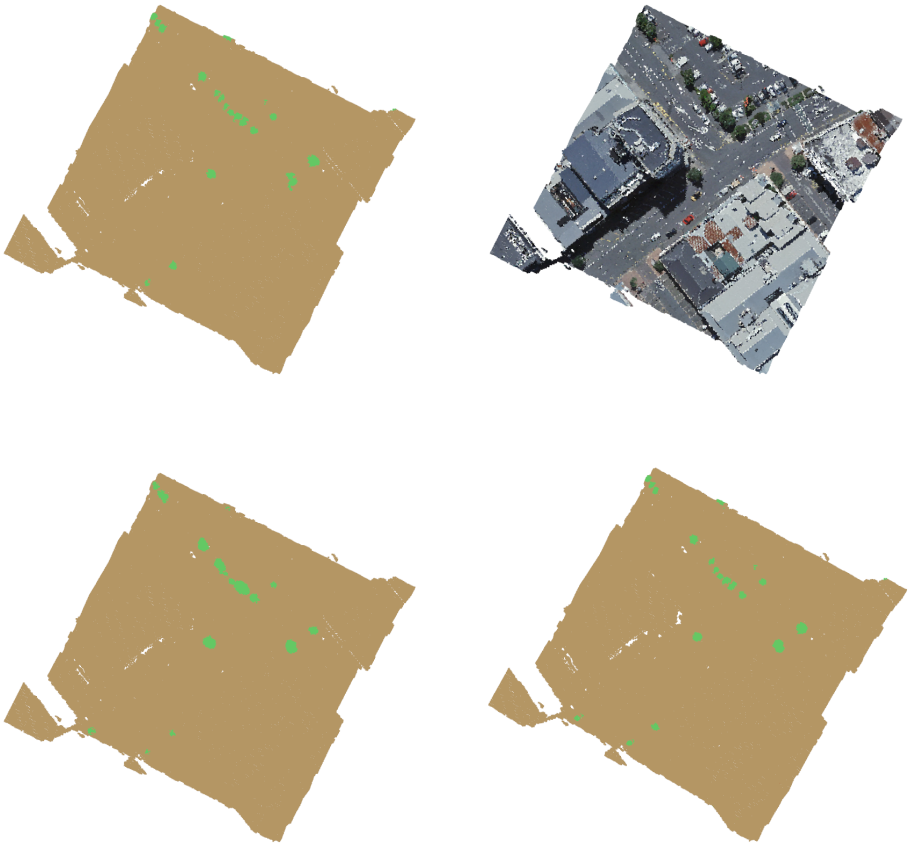

Figure 6: Semantic segmentation output comparison. A comparison of the segmentation output from a tile in the test dataset. Top, from left to right: the segmentation ground truth, and a visualisation of the tile in true colour. Bottom, from left to right: the output from our best model, and the Mask R-CNN output.

5 Discussion

5.1 Findings and analysis

5.1.1 Accuracy of our approach

In our experiments, we found that our approach to semantic segmentation based on 3D point clouds yielded moderate results. With the optimal parameters we found, we observed good segmentation for non-tree points of 91.4% and a good overall accuracy of 92.2%. However, these results were poorer than that of a Mask R-CNN approach based on raster imagery. Additionally, the discrepancy in performance was moderately large with the mIOU for the 'tree' class, where there was a 15.9% difference between the two approaches. We hypothesise that this may be caused by two factors.

Firstly, the resolution of the point cloud may have been too low for the segmentation accuracy to match that of a Mask R-CNN approach. This issue may have been compounded by the voxelisation step, which was necessary to reduce the memory requirement of the dataset, but at the same time likely constrained the accuracy of the resulting segmentation outputs. We observed this when modifying the voxel size: decreasing the voxel size from 1 m to 1/12 m increased the 'tree' mIOU by 6.3%.

Secondly, there may be complex interactions between the parameters for which our research did not account. Due to time and resource constraints, we had only considered parameters which – when adjusted one at a time – resulted in different mIOU and accuracy values. Moreover, we only modified one parameter value for each of our experiments. This was because we had also evaluated the performance of a model that used the optimal values for each parameter, but we did not achieve better results than that of the 1/12 m voxel size model. However, it may be possible to gain superior results through a combination of several parameters by combining non-optimal values of these parameters.

5.2 Future work

There are several ways in which our research can be extended. One potential area to explore is the choice of CNN library used to perform the semantic segmentation. We had chosen SoftGroup for the extensibility of its source code, and thus, our experiments were performed using this architecture. However, we did not perform experiments using other CNN architectures, and these may provide different results. Additionally, one may explore the combination of non-optimal parameter values, and whether this may yield higher mIOU and accuracy values.

Another area of interest may be to explore the performance of our CNN with a different dataset. For example, this may include point clouds of an urban area with more segmentation classes, which would avoid the data imbalance we encountered in our project. Alternatively, one may explore the use of denser point clouds: point clouds with more points, or other information, defined in a given volume.

Our research into semantic segmentation can also easily be extended to instance segmentation. While this was not the focus of our research, the SoftGroup library configuration we used also supports the instance segmentation of trees in our tree dataset.

6 Conclusion

In our research, we investigated the accuracy and performance of using a CNN to segment point clouds of Wellington urban areas into tree and non-tree regions. We pre-processed the dataset, sourced from Manaaki Whenua, to overcome memory constraints with our computer and to convert the dataset into a format supported by the SoftGroup CNN library. Through our experiments with a variant of SoftGroup, we found that from the three training parameters we examined, decreasing the voxel size resulted in the best segmentation of tree regions in our test dataset.

Additionally, we compared our segmentation results with that of a Mask R-CNN. This Mask R-CNN was trained on satellite imagery of the same region. We found that the segmentation performance for tree regions in Wellington urban areas was poorer than that of the Mask R-CNN. However, the segmentation performance of non-tree regions and the overall accuracy was only slightly lower. Further research may explore the use of alternative CNN libraries and datasets to perform semantic segmentation of urban tree regions, or extend our CNN to instance segmentation.

Bibliography

[1] E. G. McPherson, D. Nowak, G. Heisler, S. Grimmond, C. Souch, R. Grant, and R. Rowntree, "Quantifying urban forest structure, function, and value: the chicago urban forest climate project," *Urban ecosystems*, vol. 1, no. 1, pp. 49–61, 1997.

[2] L. S. Heath, J. E. Smith, K. E. Skog, D. J. Nowak, and C. W. Woodall, "Managed forest carbon estimates for the us greenhouse gas inventory, 1990—2008," *Journal of Forestry*, vol. 109, no. 3, pp. 167–173, 2011.

[3] T. Semeraro, A. Scarano, R. Buccolieri, A. Santino, and E. Aarrevaara, "Planning of urban green spaces: An ecological perspective on human benefits," *Land*, vol. 10, no. 2, 2021.

[4] J. Schreyer, J. Tigges, T. Lakes, and G. Churkina, "Using airborne lidar and quickbird data for modelling urban tree carbon storage and its distribution—a case study of berlin," *Remote Sensing*, vol. 6, no. 11, pp. 10636–10655, 2014.

[5] Y. Hu, Z. Dai, and J.-M. Guldmann, "Modeling the impact of 2d/3d urban indicators on the urban heat island over different seasons: A boosted regression tree approach," *Journal of Environmental Management*, vol. 266, p. 110424, 2020.

[6] A. Barilotti, F. Crosilla, and F. Sepic, "Curvature analysis of LiDAR data for single tree species classification in alpine latitude forests," in *Proceedings of Laser Scanning 2009*, vol. 38, (Paris, France), pp. 129–134, ISPRS, 2009.

[7] R. Sutherland, J. Soewarto, R. Beresford, and B. Ganley, "New Zealand Ecological Society Monitoring Austropuccinia psidii (myrtle rust) on New Zealand

Myrtaceae in native forest," *New Zealand Journal of Ecology*, vol. 44, no. 2, pp. 1–5, 2020.

[8] R. E. Bradshaw, S. E. Bellgard, A. Black, B. R. Burns, M. L. Gerth, R. L. McDougal, P. M. Scott, N. W. Waipara, B. S. Weir, N. M. Williams, R. C. Winkworth, T. Ashcroft, E. L. Bradley, P. P. Dijkwel, Y. Guo, R. F. Lacey, C. H. Mesarich, P. Panda, and I. J. Horner, "Phytophthora agathidicida: research progress, cultural perspectives and knowledge gaps in the control and management of kauri dieback in New Zealand," Jan. 2020.

[9] D. Leckie, F. Gougeon, D. Hill, R. Quinn, L. Armstrong, and R. Shreenan, "Combined high-density lidar and multispectral imagery for individual tree crown analysis," *Canadian Journal of Remote Sensing*, vol. 29, pp. 633–649, Oct. 2003.

[10] M. Alonzo, B. Bookhagen, and D. A. Roberts, "Urban tree species mapping using hyperspectral and lidar data fusion," *Remote Sensing of Environment*, vol. 148, pp. 70–83, 2014.

[11] S. Schmohl, A. Narváez Vallejo, and U. Soergel, "Individual Tree Detection in Urban ALS Point Clouds with 3D Convolutional Networks," *Remote Sensing*, vol. 14, p. 1317, Mar. 2022.

[12] S. Jombo, E. Adam, and S. Tesfamichael, "Classification of urban tree species using LiDAR data and WorldView-2 satellite imagery in a heterogeneous environment," *Geocarto International*, 2022.

[13] M. K. Jakubowski, W. Li, Q. Guo, and M. Kelly, "Delineating individual trees from lidar data: A comparison of vector- and raster-based segmentation approaches," *Remote Sensing*, vol. 5, no. 9, pp. 4163–4186, 2013.

[14] K. Wang, T. Wang, and X. Liu, "A review: Individual tree species classification using integrated airborne LiDAR and optical imagery with a focus on the urban environment," *Forests*, vol. 10, no. 1, pp. 1–18, 2018.

[15] F. E. Fassnacht, H. Latifi, K. Stereńczak, A. Modzelewska, M. Lefsky, L. T. Waser, C. Straub, and A. Ghosh, "Review of studies on tree species classification

from remotely sensed data," *Remote Sensing of Environment*, vol. 186, pp. 64–87, 2016.

[16] C. Premebida, J. Carreira, J. Batista, and U. Nunes, "Pedestrian detection combining rgb and dense lidar data," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4112–4117, 2014.

[17] H. Lee, K. C. Slatton, B. E. Roth, and W. P. Cropper, "Adaptive clustering of airborne LiDAR data to segment individual tree crowns in managed pine forests," *International Journal of Remote Sensing*, vol. 31, no. 1, pp. 117–139, 2010.

[18] R. Nelson, W. Krabill, and G. MacLean, "Determining forest canopy characteristics using airborne laser data," *Remote Sensing of Environment*, vol. 15, no. 3, pp. 201–212, 1984.

[19] J. Hyyppä and M. Inkinen, "Detecting and estimating attribute for single trees using laser scanner," *The photogrametric journal of Finland*, vol. 16, no. 2, pp. 27–42, 1999.

[20] Q. Chen, D. Baldocchi, P. Gong, and M. Kelly, "Isolating Individual Trees in a Savanna Woodland Using Small Footprint Lidar Data," *Photogrammetric Engineering & Remote Sensing*, vol. 72, pp. 923–932, 2006.

[21] Y. Ke and L. J. Quackenbush, "A comparison of three methods for automatic tree crown detection and delineation from high spatial resolution imagery," *International Journal of Remote Sensing*, vol. 32, no. 13, pp. 3625–3647, 2011.

[22] Z. Zhen, L. J. Quackenbush, and L. Zhang, "Trends in automatic individual tree crown detection and delineation-evolution of LiDAR data," *Remote Sensing*, vol. 8, no. 4, pp. 1–26, 2016.

[23] H. H. Aghdam and E. J. Heravi, "Guide to convolutional neural networks," *New York, NY: Springer*, vol. 10, no. 978-973, p. 51, 2017.

[24] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–21, 2021.

[25] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[26] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9224–9232, 2018.

[27] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2647–2664, 2021.

[28] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361, IEEE, 2012.

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

[30] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," *Advances in neural information processing systems*, vol. 31, 2018.

[31] Z. Wang, L. Zhang, L. Zhang, R. Li, Y. Zheng, and Z. Zhu, "A deep neural network with spatial pooling (dnnsp) for 3-d point cloud classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 8, pp. 4594–4604, 2018.

[32] J. Gu, H. Grybas, and R. G. Congalton, "A comparison of forest tree crown delineation from unmanned aerial imagery using canopy height models vs. spectral lightness," *Forests*, vol. 11, no. 6, p. 605, 2020.

[33] B. G. Weinstein, S. Marconi, S. Bohlman, A. Zare, and E. White, "Individual Tree-Crown Detection in RGB Imagery Using Semi-Supervised Deep Learning Neural Networks," *Remote Sensing*, vol. 11, p. 1309, June 2019.

[34] Z. Hao, L. Lin, C. J. Post, E. A. Mikhailova, M. Li, Y. Chen, K. Yu, and J. Liu, "Automated tree-crown and height detection in a young forest plantation using mask region-based convolutional neural network (Mask R-CNN)," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 178, pp. 112–123, 2021.

[35] M. Yang, Y. Mou, S. Liu, Y. Meng, Z. Liu, P. Li, W. Xiang, X. Zhou, and C. Peng, "Detecting and mapping tree crowns based on convolutional neural network and Google Earth images," *International Journal of Applied Earth Observation and Geoinformation*, vol. 108, p. 102764, Apr. 2022.

[36] C. J. Tucker, "Red and photographic infrared linear combinations for monitoring vegetation," *Remote sensing of Environment*, vol. 8, no. 2, pp. 127–150, 1979.

[37] C. Wen, X. Li, X. Yao, L. Peng, and T. Chi, "Airborne lidar point cloud classification with global-local graph attention convolution neural network," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 173, pp. 181–194, 2021.

[38] Y. Yang, R. Tang, J. Wang, and M. Xia, "A hierarchical deep neural network with iterative features for semantic labeling of airborne lidar point clouds," *Computers & Geosciences*, vol. 157, p. 104932, 2021.

[39] B. Rim, A. Lee, and M. Hong, "Semantic segmentation of large-scale outdoor point clouds by encoder–decoder shared mlps with multiple losses," *Remote Sensing*, vol. 13, no. 16, 2021.

[40] T. Vu, K. Kim, T. M. Luu, X. T. Nguyen, and C. D. Yoo, "Softgroup for 3d instance segmentation on 3d point clouds," in *CVPR*, 2022.

[41] I. Armeni, A. Sax, A. R. Zamir, and S. Savarese, "Joint 2D-3D-Semantic Data for Indoor Scene Understanding," *arXiv preprint arXiv:1702.01105*, 2017.

[42] S. Chen, J. Fang, Q. Zhang, W. Liu, and X. Wang, "Hierarchical aggregation for 3d instance segmentation," in *ICCV*, 2021.

[43] L. Jiang, H. Zhao, S. Shi, S. Liu, C. W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *Proceedings of the IEEE*

Computer Society Conference on Computer Vision and Pattern Recognition, pp. 4866–4875, IEEE Computer Society, 2020.

[44] M. Chen, Q. Hu, T. Hugues, A. Feng, Y. Hou, K. McCullough, and L. Soibelman, "Stpls3d: A large-scale synthetic and real aerial photogrammetry 3d point cloud dataset," 2022.

[45] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and 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.

[46] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," *arXiv preprint arXiv:1704.06857*, 2017.

[47] M. Aleksandrov, S. Zlatanova, and D. J. Heslop, "Voxelisation algorithms and data structures: A review," *Sensors*, vol. 21, no. 24, 2021.