Detecting Trees in AHN2 and AHN3 Point Clouds with a Deep Learning Detection Method

Tjerko Kieft
2629727
Vrije Universiteit van Amsterdam
t.p.kieft@student.vu.nl

Nadav Levi
2696717
Vrije Universiteit van Amsterdam
n.m.y.levi@student.vu.nl

Jacco van Wijk
2606200
Vrije Universiteit van Amsterdam
j.vanwijk@student.vu.nl

ABSTRACT

1. INTRODUCTION

Forests, trees and urban greenery play a critical role for people and the environment - not merely as an aesthetic addition to the urban landscape, but the presence of trees has been demonstrated in a number of studies to improve the physical and mental well-being of people [30]. For instance, trees remove fine particles from the air and consequently improve air quality [20]. Furthermore, trees can counter the negative impacts of urbanization, by providing shade and help in cooling down cities, and as a result reduce energy consumption [13, 17], as well as increase biodiversity and dim unpleasant traffic noise with sounds of birds chirping and rustling leaves [21, 26]. Forest conservation and restoration is broadly supported as an effective mean to fight climate-change [9] since forests can substantially contribute to counterbalancing the increase of carbon found in the atmosphere as a result of human activities [4]. In its recently adopted Biodiversity Strategy, the European Union has pledged to plant nearly 3 billion trees by 2030 [8]. Within the European union, some countries have taken concrete steps to increase tree cover; For instance, the Netherlands has implemented a law to support reforestation by requiring to report tree felling and obligatory reforestation [2], meaning that for every tree being cut down another tree must be planted to offset the loss.

Consequently, cities around the world are pursuing tree planting as a way to increase tree cover. Tree planting programs have been implemented in cities such as New York, Los Angeles, and Chicago [22, 6].

Municipalities often use dedicated systems to manage and keep track of their green infrastructure, and trees in particular are kept in dedicated tree cadasters, which contain information such as position, height, health, and species of the individual trees [27]. However, these databases are often incomplete, as cities do not have information of trees found in private areas such as domestic gardens which make a significant component of the urban green infrastructure, yet are largely unaccounted for and their impact not quantified [5]. Another contributing factor for the incompleteness of such municipal tree cadasters can be attributed to the fact that information is traditionally acquired by field-based surveys which are both time consuming and costly activities, thus some of the information can potentially be missing or outdated [27].

Therefore, a complete and automatic mapping of trees in remote sensing data would be of particular scientific and practical interest [27, 14].

The contents of the paper are as follows: In section 2, an overview of methods related to tree detection will be given. Then, in section 3 the research question will be formulated, and followed by section 4.1 where the project setup, data used, processing steps, modelling approach and the visualisation will be outlined. Finally, the results will be given in section 5 followed by a discussion in 6.

2. RELATED WORK

Tree detection and delineation from airborne laser scanning is a widely studied task in forestry[27]. Numerous methods have been developed to identify individual trees. There are a variety of methods used for analyzing and processing point cloud data in the context of tree detection. These methods usually consist of a number of tasks, such as height measuring and object detection.

The task of identifying individual trees in urban areas is particularly challenging due to fact that trees can be arranged in close proximity to each other, as groups, or individually.

While simple methods that could for instance consider height, and detect local maxima generally performed well, there has been a shift towards using deep learning methods, which became the golden standard in computer-vision- related tasks and was also applied to individual tree detection. As pointed by Strîmbu et al. [29], many sophisticated techniques are often overly fine-tuned by hand and are tailored to work in certain data conditions, and rely on assumptions made for the structure of the tree. While in contrast to that, deep learning techniques which are more data driven can be more generalised in the sense that once a basic neural network methodology is found, adaption to new data distributions is straightforward by labeling and re-training on new reference data [27]. As a result of that, deep learning methods have recently became popular for vegetation mapping [10] and could be applied for tree detection as well.

In 2017, Qi et al. [24] introduced PointNet, a deep learning network architecture that can be used to perform a number of tasks on point cloud data, such as instance segmentation and object classification. This model was then further improved with PointNet++ to be able to process more fine-grained patterns [23].

In 2018, Windrim et al applied a region based CNN and 3D CNN deep learning algorithms on a volumetric model of the Australian Tumut pine forest derived from point cloud

data[31]. In their approach, which involves the removal of ground points, object detection and segmentation, they found that it is beneficial to transfer and fine-tune a segmentation network learnt on a different site in cases where the training examples are insufficient.

Schmohl et al. investigated the use of a deep 3D single-shot detector for individual urban tree identification in ALS point clouds[27]. Their model showed promising results and was able to outperform a variety of baseline models.

A number of papers have also been applied on the AHN dataset. For example, Meijer et al. [18] used aerial images from AHN2 to identify trees. Additionally, Lucas et al. [15] used the point cloud data from AHN3 to detect trees in rural areas by classifying single points, which were then segmented. Soilán et al. [28] attempted to reproduce the AHN classification by using the PointNet model, which showed promising accuracy after training on a single tile, but the model had difficulty distinguishing trees with other type of vegetation. And more recently, Kippers et al. [11] created a tree map using the PointNet model, which was then followed by segmentation, which is the process of grouping multiple points that belong the same object, using the watershed algorithm. While the authors report an accuracy of 0.92, they noted that their model can be further improved by introducing additional data such as infrared.

3. RESEARCH QUESTION

The first step of testing the effect of the deforestation law is to be able to count the number of trees. It may be possible to do this using the Actueel Hoogtebestand Nederland point cloud data and an algorithm to detect trees. Therefore, the main question of this project is:

- Is it possible to make a probable estimate the number of trees in the Netherlands using a deep learning algorithm on Point Cloud Data?

A subquestion of this would be:

- Is it possible to make a probable estimate the change in the number of trees over time using different versions of the point cloud data?

4. METHOD

4.1 Project Setup

4.1.1 AHN dataset

The focus of this project is on two datasets of the Actueel Hoogtebestand Nederland (AHN), which is a collection of detailed and precise data on the height of the Netherlands [3]. The two point cloud datasets that will be used for the time comparison are AHN2 (2007 - 2012) and AHN3 (2014 - 2019). Both datasets are comprised of LiDAR point cloud data stored in LAZ files, which are compressed LAS files [7]. Each LAZ file consists of points with attributes, from which the ones relevant to this project are described in Table 1. Every point has an X, Y and Z variable. The AHN3 dataset has an intensity for every point, while AHN2 doesn't. This intensity difference is the reason AHN2 and AHN3 use different pre-trained models, as shown in Section 4.1.3. Besides intensity, the AHN3 dataset also provides the number of returns and the return number per point. This is the amount of pulses that are emitted and returned from one point. If the number of returns of one point is equal to the return number, then that point is expected to be a ground point. This is an import attribute for the data processing of AHN3, as shown in Section 4.1.2. All other attributes are not used in this model.

The two datasets AHN2 and AHN3 are mounted to the Databricks file system DBFS. The AHN2 dataset is 1.6 terabyte, consisting of approximately 41.000 LAZ files. The AHN3 dataset is 2.6 terabyte, consisting of approximately 1370 LAZ files.

Every LAZ file consists of a certain amount of points. A single LAZ file has a minimum and maximum X and Y value in rijksdriehoekcoördinaten format, which is expressed in meters, and have a maximum X and Y range. The point cloud only creates points for robust objects or surfaces. Therefore not every LAZ file has the same amount of points, as for example a lake in a tile's range can reduce the amount of points compared to a LAZ file which represent a part of the Netherlands with only robust objects or surfaces. The maximum X and Y range are respectively 1000 and 1000 meter for the AHN2 dataset and respectively 5000 by 6250 meter for the AHN3 dataset.

4.1.2 Data processing

The processing of the data will be done on the Databricks cluster. To read a LAZ file, the Python package LasPy is used [19].

The original datasets are processed, such that the resulting sizes of the datasets are smaller than the original dataset, which will improve the loading and computational time of the detection model. The processing consists of the following steps:

1. Reading the LAZ file;
2. AHN3 only: Remove the ground points using the number of returns;
3. Extracting the X, Y and Z coordinates;
4. AHN3 only: Extract the intensity of the points;
5. Split the original tile into subtiles of 1000 by 1000 meter;
6. Save the subtiles as separate parquet tables.

The LAZ files of the AHN2 dataset are already tiles of 1000 by 1000 meter. Therefore, these won't be split up. The tile's points' coordinates X, Y and Z will be extracted and saved in a parquet file.

The LAZ files of the AHN3 dataset can be up to 2.6 gigabytes. The processing is done parallel, by making use of Resilient Distributed Datasets (RDD's) in Apache PySpark [34]. When loading LAZ files parallel, the RAM overloads and crashes the workers. Therefore, to reduce the RAM load, the processing steps shown above are done on chunks of a LAZ file. This is done by using LasPy's `chunk_iterator` function. The LAZ file's points are loaded per 1/8th of the total points. To reduce the RAM load even more, the processing is done on one column of subtiles instead of all subtiles at once. If one would do this process on all subtiles at once, the total amount of points will saved on the RAM at the end of the chunk iterator, which would create the same

Variable name	Description \hl{ref where descriptions are from}	Note
X	X coordinate in "Rijksdriehoek" coordinate system \hl{ref}	
Y	Y coordinate	
Z	Z coordinate	
Intensity	Return strength of the laser	Only in AHN3
Number of returns	Number of pulses returned	Only in AHN3
Return number	Number of returns	Only in AHN3

Table 1: Used attributes and description of the LAZ point format.

problem as loading all LAZ file's points at once. The processing steps of an AHN3 LAZ file are shown in Algorithm 1.

Algorithm 1 Pre-processing an AHN3 LAZ file

1: Calculate the x and y boundaries of the subtiles based on the x and y range of d_n
2: **for** x range $= 1, 2, \ldots, N$ **do**
3: **for** $chunk_1, \ldots chunk_8$ **do**
4: Remove points from $chunk_i$ for which the number of returns is equal to the return number
5: Divide the non-ground points in $chunk_i$ over the subtiles within the x range and the y ranges
6: **end for**
7: Save the subtiles as parquet files
8: **end for**

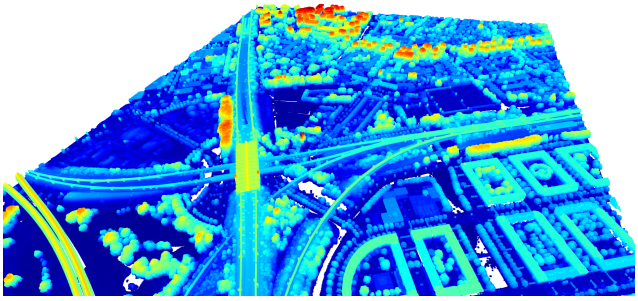
The ground points removal, only applicable to the AHN3 dataset, causes the biggest reduction of filesizes. On average, the amount of points reduces approximately five times by removing the points for which the number of returns is equal to the return number. In Figure 1, a visualization of an AHN3 point cloud before and after ground removal is shown. The amount of points after the ground removal is 3.5 times smaller than before the removal.

In attempts to further reduce the data, more methods for points removal are evaluated, but none have been proven to be efficient enough. One of those is removing the points that represent buildings. These points can be found using the Basisregistratie Adressen en Gebouwen (BAG) dataset [1]. The BAG dataset is loaded in using Web Feature Service (WFS), which only loads the building within the X and Y range of the LAZ file. The points in the LAZ file are then extracted and saved in a GeoDataFrame, which can overlay the geometrical data of the buildings over the points to remove those points that are within buildings. However, the total computational time of the processing induces significantly, while the reduction of the data is not significant. Therefore the building removal is not efficient enough and will not be used.

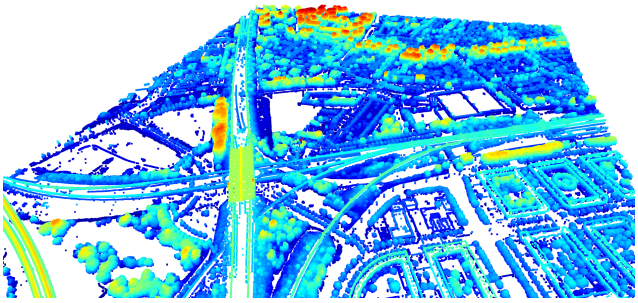
4.1.3 *Modelling*

To detect the cluster of points in the point cloud that represent a tree, a detection model must be chosen. The used model is the Forest 3D app, publicly available on Github, made by Windrim et al. [33, 32]. This model uses the YoloV3 architecture to detect trees in a LiDAR point cloud. The model detects trees as a cluster of points and gives that cluster a label. Therefore, the amount of unique labels will be the amount of trees detected.

Publicly available weights that are trained on trees of the types that are common in the Netherlands in a LiDAR point



(a) Before ground points removal, total of 35.26 million points.



(b) After ground points removal, total of 9.90 million points.

Figure 1: A visualization of a subtile from AHN3 before and after the ground points are removed with the number of returns attribute.

cloud such as AHN2 or AHN3 have not been found. Therefore, training such a model seems the best approach. As stated by Windrim et al., this can be done using the DarkNet architecture [25]. Alas, an attempt to train such a model did not succeed due to the time constraint of the project and the difficulty in obtaining ground truth for the data. However, as mentioned in section 2, Windrim et al claim that transferring a model trained on different data can be beneficial in cases where the training data is insufficient[31]. Therefore, a decision was made to use a pre-trained YoloV3 model with weights provided by the same authors.

For AHN2, the model trained on the Tumut1 dataset will be used and for AHN3 the model trained on the Tumut2 dataset will be used. Both networks are provided by Windrim et al. in [33]. The choice of these models stem from the fact that both models are trained on a high resolution point cloud, where Tumut2 also includes the intensity attribute. However, the models are trained on mature pines in Tumut, Australia. For that reason, the performance of the model is expected to be suboptimal. Therefore, the resulting amount of trees will be an estimation based on two experiments.

The first experiment is a manual true or false positive or negative evaluation to obtain the precision and recall of the model. These two measures indicate the performance of the model. The meanings of the true or false positives or negatives are shown in Table 2. For the best parameters, the model runs on various subtiles. The subtiles are small enough that one can manually count the trees with a visualization similar to Figure 1. One can zoom in on the visualization and count the trees, which can cause human errors to miscount because of f.e. a cluster of trees where the individual trees are not visually distinct. A visualization is made of only the trees detected where every cluster with a unique label is distinct from other clusters. Another visualization is made of the points that are not identified as trees by the model. Using these two visualizations, one can count the trees detected as trees (true positives), count other objects detected as trees (false positives) and the amount of trees that are not detected (false negatives). From these values, the precision and recall of the model can be measured. The precision and recall are

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} \ + \ \text{false positives}}, \tag{1}$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} \ + \ \text{false negatives}}. \tag{2}$$

Using the precision and recall, an estimation for the total trees in the tile N_{est} can be made

$$N_{\text{est}} = N \cdot \frac{\text{precision}}{\text{recall}}, \tag{3}$$

where N is the amount of trees detected. This equation is build on the fact that if one finds N trees, precision is the fraction of actual trees and recall is the fraction of actual trees and missed trees. The manual precision and recall evaluation is done on multiple tiles, for which a mean value and an standard deviation for both the precision and recall will be found. These standard deviations will be included in the estimation error.

The second experiment is based on the hyperparameter `stepsize`, which is the step size of the window that moves over the tile to detect the trees. The size of the moving window is set to be 5/4 by 5/4 of the step size, such that the overlay of the adjacent windows is big enough to find a tree that's only partially in one of the windows. Using a smaller window and smaller step size results in a more accurate detection, but requires more computational power and time. Therefore, the experiment will detect trees for multiple step sizes on multiple tiles, and fit a function that approximates the ratio of the trees found with respect to the trees found for the smallest step size. Using that fitted function, the step size can be set to a higher value, resulting in less computational time, which is crucial due to the large dataset. The smallest step size will be chosen from an assessment on time and performance. The smallest step size used in this experiment will be the same value for the precision and recall evaluation.

	True	False
Positive	Detected tree as "tree"	Detected not tree as "tree"
Negative	Detected not tree as "not tree"	Detected tree as "not tree"

Table 2: Attributes and description of the LAZ point format.

Using the results of the two experiments, one can estimate the amount of trees N_{est} using the detected amount of trees N for a certain step size s. The estimation steps are as following,

1. Use the fitted function $f(s)$ to find the ratio of trees found w.r.t. the smallest step size $\tilde{s}$. This ratio is $f(s) = N/\tilde{N}$, such that $\tilde{N} = N/f(s)$ is the estimation for the amount of trees found for step size $\tilde{s}$;

2. Using the recall r and precision p found for step size $\tilde{s}$, the total estimate of trees in the tile N_{est} is found with $N_{\text{est}} = \tilde{N}p/r$.

The error on the estimation is done with error propagation rules using the errors in precision, recall and the fitted formula's parameters [12].

4.1.4 *Data product*

After detection, multiple data products are acquired. The first one is a geoJSON, which is a collection of every detected tree and its coordinates. This collection will be used in the visualization, which is explained in Section 4.2. The second data product is the estimated density per subtile. Due to the time constraint on the project, the priority will be set on cities which provide the data of municipality managed trees. This will provide a more qualitative rather than quantitative result, as without priority the chance exists only parts of the Netherlands are detected for which there is less social relevance such as in a city. The chosen cities, chosen by availability of municipality managed trees, are Amsterdam, Amersfoort, Den Haag, Nijmegen, Utrecht, Groningen and Eindhoven. Using the coordinates of these cities, the amount of trees within those coordinates can be counted and by doing the estimation steps described above, the estimated amount of trees per city is calculated. Using the size of the city, calculated with the sizes of the used subtiles, the estimated density is found. After doing this for both the AHN2 and AHN3 dataset, the change in estimated density can be calculated. The final data product is achieved by calculating the estimated density per subtile of a city. Every

city is divided into 100 subtiles, for which all estimated densities are calculated. These densities and the density changes between AHN2 and AHN3 are used in the visualization.

4.2 Visualization Setup

In order to visualise the output of the model with a web-application, a method of overlaying geographical information on a map had to be found, while adhering to the project constraints and keeping the website static and as simple as possible. The initial idea was to overlay points on a map where ever the model detected a tree, as well as a grid of polygons to display tree density in a specific area. Taking inspiration from a previous project [16], Mapbox was found to be a suitable package. Mabbox provides *Mapbox GL JS*; a JavaScript library for vector maps on the Web. Additionally, Mapbox provides *Mapbox Studio*, which allows to create customized tiles to be used as a layer on the map.

As previously mentioned in 4.1.4, the data product consisted of detected trees and their respective coordinates in geoJSON format, which could be used directly in Mapbox GL JS or uploaded as a layer to Mapbox Studio. The approach taken was to aggregate data into different layers which could then in turn be overlayed on the map with the ability to hide or display a specific layer. Hence, a unique geoJSON frame was collated for AHN2, AHN3, along with a density layer to display change between AHN2 and AHN3.

An additional layer was added to include the cadaster data of selected municipalities - which would serve as a baseline to compare against the model's output.

5. RESULTS

Due to the time constraint of the project, only results for the cities containing data about the trees from the local municipality were analysed. Specifically for Groningen, only the analysis on AHN2 was finished. This means that all of the results in this section will be catered to the cities Amsterdam, Amersfoort, Den Haag, Eindhoven, Nijmegen, and Utrecht.

5.1 Estimation experiments

The estimated number of trees per step size hyperparameter (w.r.t. the smallest value of step size) is shown for AHN2 in Figure 2 and AHN3 in Figure 3 for twenty randomly chosen subtiles. The analysis for AHN2 was done until step size 20 in contrast to the step size 80 for AHN3, because an individual run on the AHN2 dataset took less computational time due to the lower point density in each file. The function that could fit the data the best was found to be an exponential function of the form

$$f(s) = a \cdot \exp(-b \cdot s) + c \tag{4}$$

where $f(s)$ is the ratio of trees found w.r.t. the smallest step size $\tilde{s}$, s is the step size and a, b and c are the parameters of the fit. Table 3 shows the parameters that had the highest R^2 value and hence were the best found fit.

Using the true or false positive or negative evaluation, the precision and recall for AHN2 and AHN3 are found and are shown in Table 4. The resulting precision and recall are found as a mean and standard deviation of 10 experiments on AHN2 and 10 on AHN3. The table shows that the mean precision is lower for AHN2 than AHN3, but not significantly lower due to the error. The recall of AHN2 is also not significantly higher than AHN3.

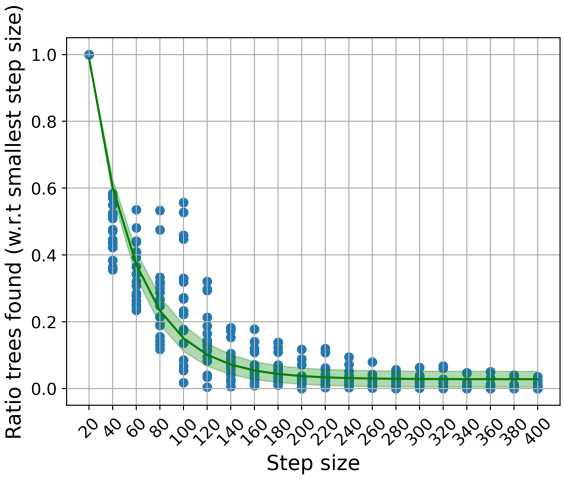


Figure 2: The ratio of trees found w.r.t. the trees found at the smallest stepsize $\tilde{s} = 20$ for AHN2. The blue dots are individual datapoints and the green line shows the fitted function. The green space shows the uncertainty in the fitted function.

5.2 Densities per city

Using the just mentioned methods, the estimations are shown in Table 5 and Table 6. The change in estimated density is shown in Table 7. This shows that for all cities in a whole, the estimated density of trees has gone down from the time of AHN2 to the time of AHN3.

5.3 Visualization

The visualisation was made in the manner outlined in 4.2; a static website was created using Javascript, HTML and CSS, where the Mapbox GL JS libarary was extensively used. Due to time and computational constraints, results were limited to the 7 aforementioned municipalities. A snippet of the visualisation showing various layers is given with figure 4.

6. DISCUSSION

The time window of this project and the size of the datasets were the hardest parts of this project. The methods that were implemented to abide by this time window created a need for an estimation which has a relatively large variance. The size of the datasets were handled by taking a subset of the files and creating an implementation which handles the data in chunks that do not overload the ram storage. These solutions come with their own set of problems, which will be addressed in the following sections.

6.1 Estimation experiments

The uncertainty in the precision and recall are more responsible for the uncertainty in the estimate than the fit uncertainty. This is due to the detection model not being the optimal model for the AHN point clouds. Training a model, or finding a model that's trained on the type of trees in the Netherlands, would have improved the precision and recall. As training a model was not an option, an existing model had to be optimized for the given subset.

	a	b	c	R^2
AHN2	0.961 ± 0.008	0.515 ± 0.016	0.0278 ± 0.0048	0.91441
AHN3	0.925 ± 0.009	0.203 ± 0.006	0.0788 ± 0.0090	0.96255

Table 3: The found parameters with uncertainties and the R-squared value for the fit shown in Equation 4 for both AHN2 and AHN3.

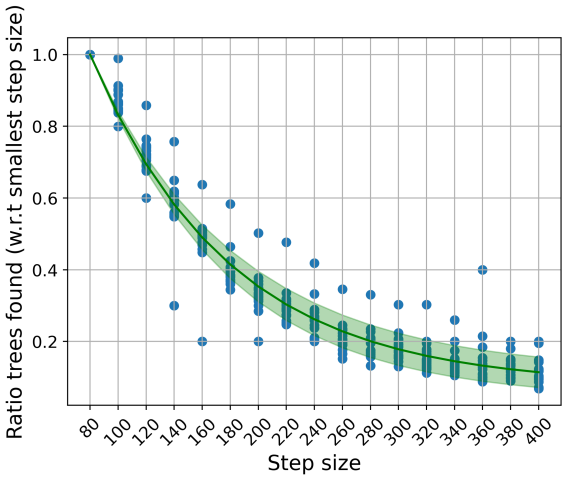


Figure 3: The ratio of trees found w.r.t. the trees found at the smallest stepsize $\tilde{s} = 80$ for AHN3. The blue dots are individual datapoints and the green line shows the fitted function. The green space shows the uncertainty in the fitted function.

	Precision	Recall
AHN2	0.818 ± 0.167	0.562 ± 0.285
AHN3	0.885 ± 0.298	0.422 ± 0.207

Table 4: The mean precision and recall with their respective variances for both AHN2 and AHN3.

The precision and recall experiment was done on a random sample of subtiles from the subtiles created after the pre-processing. These randomly chosen subtiles were not limited to the cities subtiles. Therefore, the precision and recall values are based on how well the model performs on different kinds of area's, for example areas with a higher tree density, which makes it more difficult to distinguish individual trees. An improvement to this experiment would be to classify the subtiles on which the detection model will run, and find a precision and recall per class, such that the found values correspond well to the type of subtile. This classification could even be done on cities, where for example one distinguishes a subtile from dense city centre to less dense suburb.

The differences in the subtiles might have influenced the step size experiment. As seen in Figures 2 and 3, there are multiple outliers, for which the optimal step size might be different. The classification of the subtiles might be a partial solution to this problem, as one would be able to couple the step size to a certain subtile class.

6.2 Densities per city

The boundaries of the cities were created by drawing a square around all trees mentioned by each municipality. However, some cities have trees within their data that are far from the urban area of the city. Especially in the case of Eindhoven, which was the city with the largest size in the evaluation, the tree data gives a wrong impression about the actual size of the city. Together with the fact that city borders always change and the AHN datasets are (at least) two years apart, using the same city borders for both the analysis on AHN2 and AHN3 can come with inaccuracies. For future research when the focus is also on cities alone, it would give a better indication of the deforestation within the city itself if the official city boundaries, of the year the specific AHN dataset was created, were used.

The selection of LAZ files for the specific cities, based on if they contain at least a single point within a city, was executed on the unprocessed LAZ files. Because the unprocessed AHN3 LAZ files cover a distance of 6.25 km by 5.0 km and AHN2 only 1.0 km by 1.0 km, this helped to create the significant difference between the city sizes in the analysis on AHN2 and the analysis on AHN3. This difference can be seen in Table 5 and 6, which have a significant difference between the size of each city. Although this difference is taken into account when estimating the density and when creating the visualization, it still creates a shift in which area is analysed when looking at the whole city. For future research this can be fixed in multiple different ways. Firstly, if the the selection of LAZ files is done before and after processing the LAZ files, the number of files that is processed is minimized and the number of points used for the estimation is also minimized. Secondly, a separate table containing meta data about each LAZ file could be made during the processing. This way, there is only a need to read that specific table to know which LAZ files to use.

Due to the difference in sizes of the processed LAZ files of the AHN2 and AHN3 dataset there are multiple files overlapping in coordinates for each of dataset. Therefore, if an analysis on one of the datasets fails because of a problem, e.g. a corrupt file, this will only overlap partly with the files from the other dataset. This creates a situation where the analysis of the change sees zero trees where an error has occurred. Therefore, the ratio of subtiles which were completed without issue (as shown in Table 5 and 6) has a very important role in the estimation of the density. To make sure the estimate is not influenced by these corrupt files, future analyses should either make the LAZ files of the different datasets overlap fully, or continue the analysis until there is no corrupt estimation left.

6.3 Visualization

The visualisation was limited to the output of the model. Unsurprisingly, as the model was not applied across the full data, the scope of the project to be restricted to selected cities. However - The results obtained do give an interesting insight into the number of trees across the cities included. What was evident from the visualisation was that the claim

City	Trees detected	Size (km^2)	Completed ratio	Trees estimated	Density estimated (tree / km^2)
Amsterdam	16071	159.87	1.00	$(5.4 \pm 3.0) \times 10^6$	$(3.3 \pm 1.8) \times 10^3$
Amersfoort	8386	80.78	0.70	$(2.8 \pm 1.5) \times 10^5$	$(3.5 \pm 1.9) \times 10^3$
Den Haag	15853	162.15	0.99	$(5.3 \pm 2.9) \times 10^5$	$(3.3 \pm 1.8) \times 10^3$
Nijmegen	25475	119.80	1.00	$(8.6 \pm 4.7) \times 10^5$	$(7.1 \pm 3.9) \times 10^3$
Utrecht	23760	193.54	1.00	$(8.0 \pm 4.3) \times 10^5$	$(4.1 \pm 2.3) \times 10^3$
Eindhoven	53008	240.51	1.00	$(1.7 \pm 1.0) \times 10^6$	$(7.4 \pm 4.1) \times 10^3$

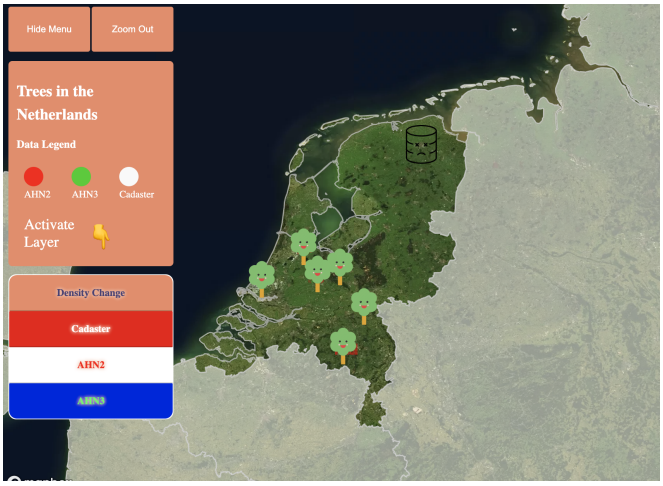
Table 5: The results of the tree detection per city for AHN2. The trees detected are the output of the detection model. The size of the city is the total size of the subtiles on which the detection is executed. The completed ratio shows how many subtiles of the cities subtiles have been used for detection, the residue subtiles are unused due to corrupt processed files. The trees estimated and density estimated are the results after the stepsize and precision and recall estimation.

City	Trees detected	Size (km^2)	Completed ratio	Trees estimated	Density estimated (tree / km^2)
Amsterdam	58610	298.61	0.88	$(6.2 \pm 3.8) \times 10^5$	$(2.0 \pm 1.3) \times 10^3$
Amersfoort	45230	263.56	1.00	$(5.3 \pm 3.2) \times 10^5$	$(2.0 \pm 1.2) \times 10^3$
Den Haag	66661	300.97	0.96	$(7.7 \pm 4.6) \times 10^5$	$(2.5 \pm 1.5) \times 10^3$
Nijmegen	50837	280.31	1.00	$(6.0 \pm 3.6) \times 10^5$	$(2.1 \pm 1.3) \times 10^3$
Utrecht	72630	371.71	1.00	$(8.5 \pm 5.2) \times 10^5$	$(2.3 \pm 1.4) \times 10^3$
Eindhoven	76064	434.89	0.76	$(9.0 \pm 5.4) \times 10^5$	$(2.1 \pm 1.2) \times 10^3$

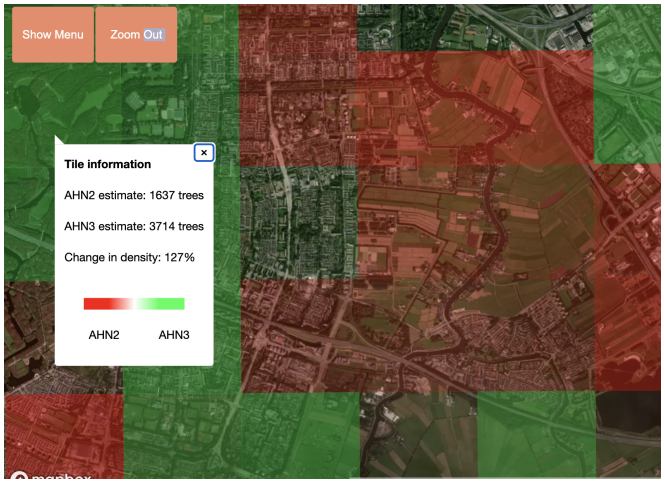
Table 6: The results of the tree detection per city for AHN3. The trees detected are the output of the detection model. The size of the city is the total size of the subtiles on which the detection is executed. The completed ratio shows how many subtiles of the cities subtiles have been used for detection, the residue subtiles are unused due to corrupt processed files. The trees estimated and density estimated are the results after the stepsize and precision and recall estimation.

City	Estimated density change (%)
Amsterdam	-38.4 ± 19.3
Amersfoort	-41.9 ± 19.9
Den Haag	-22.2 ± 14.1
Nijmegen	-70.0 ± 17.1
Utrecht	-44.1 ± 20.1
Eindhoven	-72.1 ± 16.4

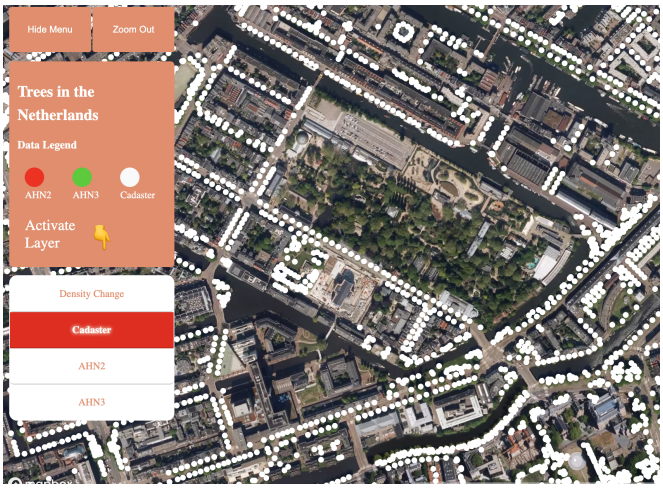
Table 7: The change in estimated density per city in percentages from AHN2 to AHN3.



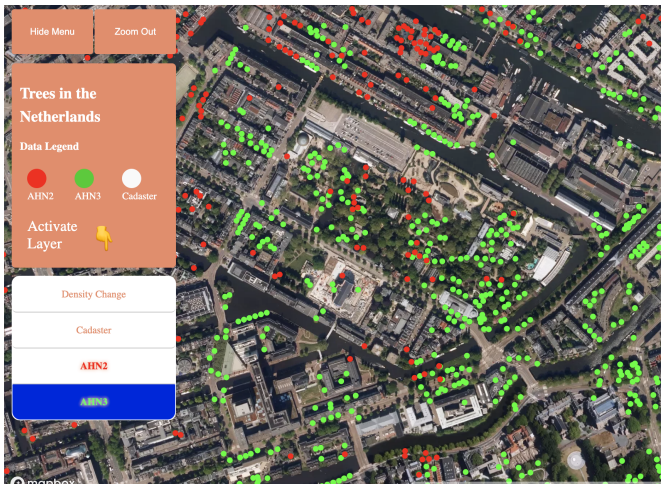
(a) Visualization of the full map.



(b) Portion of the visualization of the density difference.



(c) Municipality cacdaster tree data.



(d) AHN2 and AHN3 detected trees

Figure 4: Four different views of the visualization.

made in the introduction - that the municipal cadasters are limited and do not contain all trees in a particular city seems to be supported by the result.

As seen in the visualization, the points of trees detected from the AHN3 dataset show some vertical gaps in the data. This is due to the preprocessing done per column of subtiles, and if this preprocessing fails - for example due to the cluster stopping or the workers crashing - the files from this column of subtiles might become corrupt. The detection model will then skip over these files and not detect any trees for the corrupt subtiles, creating these vertical gaps in the data. If time allowed, an improvement afterwards would be to create an algorithm that searches these corrupt files and recreates the subtiles to allow detection on them.

7. CONCLUSION

The main goal of this project was to find out if it is possible to make a probable estimate of the number of trees in the Netherlands using a deep learning algorithm on Point Cloud Data. It is shown that with the methods used the estimate has a variance of a size that makes it not probable. As mentioned above, this variance can be brought down using methods that take more time which could make the estimate more probable in the future.

Furthermore, this means that the change in the number of trees over time using different versions of the point cloud data also does not have a probable estimate with the methods used. However, if the only interest is whether the number of trees has decreased or increased, this estimate is more relevant.

8. CONTRIBUTION

The contributions to this project are shown in Table 8.

Nadav Levi	Jacco van Wijk	Tjerko Kieft
Investigated relevant research on the detection of trees	Investigated relevant research on the detection of trees	Investigated relevant research on the detection of trees
Created visualization	Did tests with removing using BAG dataset	Created detection model
Investigated relevant research on social relevance	Did estimation experiment on precision and recall	Created and tested pre-processing scripts
Wrote introduction	Ran pre-processing script AHN2	Ran pre-processing script AHN3
Wrote relevant research	Did fit experiment on AHN2	Ran detection model on both AHNs
Wrote method on visualization	Found fit parameters for estimation	Provided data for visualization
Wrote results and discussion on visualization	Wrote results on estimation experiments	Did fit experiment on AHN3
Collected and processed municipal data(cadaster)	Wrote results and discussion on city densities	Wrote method on project setup
	Wrote conclusion	Wrote discussion on estimation experiments

Table 8: The contributions to the project per group member.

9. REFERENCES

[1] Basisregistratie Adressen en Gebouwen (BAG). https://data.overheid.nl/en/dataset/basisregistratie-adressen-en-gebouwen--bag-.

[2] Reporting tree felling and obligatory reforestation. https://business.gov.nl/regulation/reporting-tree-felling/. Accessed: 2022-10-22.

[3] AHN. Algemeen Hoogtebestand Nederland. http://www.ahn.nl/index.html. Accessed: 2022-10-22.

[4] J.-F. Bastin, Y. Finegold, C. Garcia, D. Mollicone, M. Rezende, D. Routh, C. M. Zohner, and T. W. Crowther. The global tree restoration potential. *Science*, 365(6448):76–79, 2019.

[5] R. W. Cameron, T. Blanuša, J. E. Taylor, A. Salisbury, A. J. Halstead, B. Henricot, and K. Thompson. The domestic garden–its contribution to urban green infrastructure. *Urban forestry & urban greening*, 11(2):129–137, 2012.

[6] D. A. Doroski, M. S. Ashton, and M. C. Duguid. The future urban forest–a survey of tree planting programs in the northeastern united states. *Urban Forestry & Urban Greening*, 55:126816, 2020.

[7] A. S. for Photogrammetry and R. Sensing. https://www.asprs.org/divisions-committees/lidar-division/laser-las-file-format-exchange-activities. Accessed: 2022-10-22.

[8] V. Hermoso, S. Carvalho, S. Giakoumi, D. Goldsborough, S. Katsanevakis, S. Leontiou, V. Markantonatou, B. Rumes, I. Vogiatzakis, and K. Yates. The eu biodiversity strategy for 2030: Opportunities and challenges on the path towards biodiversity recovery. *Environmental Science & Policy*, 127:263–271, 2022.

[9] V. Hermoso, A. Regos, A. Morán-Ordóñez, A. Duane, and L. Brotons. Tree planting: A double-edged sword to fight climate change in an era of megafires. *Global Change Biology*, 27(13):3001–3003, 2021.

[10] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz. Review on convolutional neural networks (cnn) in vegetation remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 173:24–49, 2021.

[11] R. Kippers, L. Moth, S. O. Elberink, C. Deltas, and W. Rivers. Automatic modelling of 3d trees using aerial lidar point cloud data and deep learning. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43:B2–2021, 2021.

[12] H. H. Ku. Notes on the use of propagation of error formulas. 2010.

[13] D. M. Kurn, S. E. Bretz, B. Huang, and H. Akbari. The potential for reducing urban air temperatures and energy consumption through vegetative cooling. Technical report, Lawrence Berkeley Lab., CA (United States), 1994.

[14] X. Li, W. Y. Chen, G. Sanesi, and R. Lafortezza. Remote sensing in urban forestry: Recent applications and future directions. *Remote Sensing*, 11(10):1144, 2019.

[15] C. Lucas, W. Bouten, Z. Koma, W. D. Kissling, and A. C. Seijmonsbergen. Identification of linear vegetation elements in a rural landscape using lidar point clouds. *Remote Sensing*, 11(3):292, 2019.

[16] J. E. Lucien Martijn, Stijn Meijerink. Detect illegal buildings based on lidar point cloud data. https://event.cwi.nl/lsde/2019/results/group02.pdf.

[17] E. G. McPherson and J. R. Simpson. Potential energy savings in buildings by an urban tree planting programme in california. *Urban forestry & urban greening*, 2(2):73–86, 2003.

[18] M. Meijer, F. Rip, R. van Benthem, J. Clement, and C. van der Sande. Boomkronen afleiden uit het actueel hoogtebestand nederland: kwaliteitsaspecten rondom het geautomatiseerd in kaart brengen van bomen op basis van het ahn2-bestand. Technical report, Alterra, Wageningen-UR, 2015.

[19] T. Montaigu. laspy: Python library for lidar las/laz io. 2022.

[20] D. J. Nowak, S. Hirabayashi, A. Bodine, and R. Hoehn. Modeled pm2. 5 removal by trees in ten us cities and associated health effects. *Environmental pollution*, 178:395–402, 2013.

[21] L. Pesola, X. Cheng, G. Sanesi, G. Colangelo, M. Elia, and R. Lafortezza. Linking above-ground biomass and biodiversity to stand development in urban forest areas: A case study in northern italy. *Landscape and Urban Planning*, 157:90–97, 2017.

[22] S. Pincetl, T. Gillespie, D. E. Pataki, S. Saatchi, and J.-D. Saphores. Urban tree planting programs, function or fashion? los angeles and urban tree planting campaigns. *GeoJournal*, 78(3):475–493, 2013.

[23] C. Qi, L. Yi, H. P. Su, and L. P. Guibas. Deep hierarchical feature learning on point sets in a metric space. arxiv 2017. *arXiv preprint arXiv:1706.02413*.

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[25] J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

[26] J. A. Salmond, M. Tadaki, S. Vardoulakis, K. Arbuthnott, A. Coutts, M. Demuzere, K. N. Dirks, C. Heaviside, S. Lim, H. Macintyre, et al. Health and climate related ecosystem services provided by street trees in the urban environment. *Environmental Health*, 15(1):95–111, 2016.

[27] S. Schmohl, A. Narváez Vallejo, and U. Soergel. Individual tree detection in urban als point clouds with 3d convolutional networks. *Remote Sensing*, 14(6):1317, 2022.

[28] M. Soilán Rodríguez, R. Lindenbergh, B. Riveiro Rodríguez, A. Sánchez Rodríguez, et al. Pointnet for the automatic classification of aerial point clouds. *ISPRS Annals of Photogrammetry Remote Sensing and Spatial Information Sciences*, 2019.

[29] V. F. Strîmbu and B. M. Strîmbu. A graph-based segmentation algorithm for tree crown extraction using airborne lidar data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104:30–43, 2015.

[30] J. B. Turner-Skoff and N. Cavender. The benefits of trees for livable and sustainable communities. *Plants, People, Planet*, 1(4):323–335, 2019.

[31] L. Windrim and M. Bryson. Forest tree detection and segmentation using high resolution airborne lidar. *CoRR*, abs/1810.12536, 2018.

[32] L. Windrim and M. Bryson. Detection, Segmentation, and Model Fitting of Individual Tree Stems from Airborne Laser Scanning of Forests Using Deep Learning. *Remote Sensing*, 12(9):1469, May 2020.

[33] L. Windrim and M. Bryson. Forest 3D App: inventory from pointcloud data. https://github.com/lloydwindrim/forest_3d_app, 2021.

[34] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. Apache Spark: a unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, Oct. 2016.