Geo-information Science and Remote Sensing

Thesis Report GIRS-2024-22

Estimating apple yield by detecting orchard apples in UAV-acquired RGB images

Lars ten Kate

04-03-2024

WAGENINGEN
UNIVERSITY & RESEARCH

Estimating apple yield by detecting orchard apples in UAV-acquired RGB images

Lars ten Kate

Registration number: 1010951

Supervisors:

C (Chenglong) Zhang

Dr.ir. L (Lammert) Kooistra

A thesis submitted in partial fulfilment of the degree of Master of Science

at Wageningen University and Research,

The Netherlands.

04-03-2024

Wageningen, The Netherlands

Thesis code number: GRS-80436
Thesis Report: GIRS-2024-22
Wageningen University and Research
Laboratory of Geo-Information Science and Remote Sensing

Abstract

Within the field of precision agriculture, yield estimation plays a crucial role in the decision-making of farmers. Traditional methods rely on manual measurements. Computer vision, particularly artificial neural networks, can help automate fruit counting and yield estimation. These newer methods often require manual labour to acquire images which is time-consuming, inaccurate and prone to human errors. UAVs offer a solution by providing cost-effective, flexible and quick access to aerial images. This report investigates apple yield estimation by detecting orchard apples in unmanned aerial vehicle (UAV) acquired RGB images. First, an existing dataset containing UAV images is used to train a faster R-CNN model. During the training phase, the influence of reflections by the sun and shadows in the trees on apple detection are investigated by different setups of the training datasets. Lastly, a prediction by the best performing model is used in a regression to estimate apple yield. An underachieving model with an F1-score of 0.56 and an average precision of 0.45 resulted in a regression with an R^2 of 0.39. Despite these low performances, the results demonstrate that a front to end set-up to estimate apple yield is feasible and deserves further investigation. As a final addition, recommendations were made to improve this front to end set-up including implementing a better annotation strategy, refining image acquisition methods, and incorporating state-of-the-art deep learning algorithms.

Table of Contents

1 Introduction

1.1 Research Background

Precision agriculture is gaining traction in fruit plantations. Planting, cultivating and harvesting are important processes in the management of apple orchards (González-Araya et al., 2015). Estimating the yield of an orchard is key due to its relevance in marketing strategies (Gongal et al., 2015; Tian et al., 2019). It can contribute to a more precise management of apple orchards. The size of fruit is important to assess its quality (Gongal et al., 2018). Yet, information about yield and size are often only available after the fruits are harvested, weighed and sorted (Bulanon et al., 2009). Earlier yield estimation within an apple orchard can be used to improve orchard management. For example, it can be important in the planning of harvesting operations according to the volume stock required at the supply chain level (Chinchuluun et al., 2006). Within the chain: harvesting, packaging, transporting and marketing operations can also be improved with early yield estimations (Zhou et al., 2012; Wang et al., 2013).

Counting the apples in an orchard is a suitable way for yield estimation (Wang et al., 2013). The traditional process of manual measurements of fruit characteristics is based on past experience, and historical data collected by farmers (Rahnemoonfar and Sheppard, 2017). This can be inaccurate, inefficient and subjected to bias (Aggelopooulou et al., 2013; Bargoti and Underwood, 2017a). With the advances in computers, cameras and image analysis technology, new methods for fruit counting and yield estimations have been developed (Gongal et al., 2015). These new methods use computer vision to automate the manual labour in the process of detecting and counting apples. Recent studies about yield estimation make use of artificial intelligence (AI), in specific artificial neural networks (ANNs), to detect fruits (Sa et al., 2016; Kamilaris and Prenafeta-Boldú., 2018; Zhu et al., 2018). ANNs also have proven to be highly successful in yield estimations of apples (Cheng et al., 2017).

Although the counting part is covered by the ANNs used in yield estimation, models still require pictures taken at ground level (Gongal et al., 2018; Koirala et al., 2019a). Frequent manual labour is still required to acquire these images, which is ineffective and less viable in orchards. Unmanned aerial vehicles (UAVs), which have seen increasing usage in precision farming, can be a substitute for acquiring images. Their low cost, flexibility and high repeatability are positive features (Martínez et al., 2017). UAVs are capable of generating large amounts of data in the form of videos and images (Csillik et al., 2018; Ziliani et al., 2018). In Apolo-Apolo et al. (2020a, 2020b) a convolutional neural network (CNN) was used to detect apples and citrus fruits in UAV images. The deep learning model in Apolo-Apolo et al. (2020a) showed promising values and great potential for apple yield estimation.

1.2 Research Needs

While detecting apples in UAV images is possible, several difficulties in the detection process come upfront. Fruits invisible in the image, hidden by leaves or other fruits, are the main problem for deep learning models using object detection (Kamilaris and Prenafeta-Boldú., 2018). Approximately 60-70% of fruits are visible from the outside (Moltó et al., 1992; Jiménez et al., 2000). Images taken closer to the zenith (vertically downward at 90°) specifically, which can be the case for UAV images, complicate acquiring an accurate apple count through deep learning models (Y. Chen et al., 2019). In addition to invisible fruits, disturbances in light conditions can influence apple detection. The position of the sun or camera, different weather conditions and other factors, influence the lightning in images. Outdoor images in specific, might suffer from brightness distortions caused by light conditions (Y. Chen et al., 2019). To allow the deep learning model to be robust for these differences, it is important to train the algorithm on data containing different light conditions (Sabzi et al., 2018). In Apolo-Apolo et al. (2020a), after counting the visible apples in UAV images, linear regression was used to estimate the

number of total apples on each tree. They were able to obtain results greater than 90% in terms of precision.

Object detection has developed rapidly in the world of deep learning (Redmon et al., 2016; Ren et al., 2015a). It often makes use of RGB, NIR, thresholds, shape and convolutional filters for the classification of fruits. Afterwards, shape fitting, bounding boxes or blob segmentation can locate individual objects. Counting the bounding boxes results in the amount of fruits detected in an image (Koirala et al., 2019b). A different approach is semantic segmentation. This method tries to classify each pixel of an image (Cireşan et al., 2012). Combining object detection and semantic segmentation together results in instance segmentation (L.C. Chen et al., 2018). It gives an instance in the image and classifies each pixel. For fruits specifically, the advantage of instance segmentation is the more detailed classification of fruits. It is capable of classifying more complex structures such as fruits hidden by foliage (Santos et al., 2020).

In previous literature several algorithms have been used to detect different types of fruit. Sa et al. (2016) adapted the Faster Region-based CNN (Faster R-CNN) model, through transfer learning, for the task of fruit detection using both colour (RGB) and near-infrared (NIR) images. This led to a novel multi-modal Faster R-CNN model, which achieves state-of-the-art results compared to prior work. Although the model was trained to detect sweet peppers it can be retrained to perform detection of seven fruits, including apples. 61 apple images, split into 80/20 for training and testing, were used. These images were taken from Google Images and contained apples close up in the images. Koirala et al. (2019a) used the already existing you only look once (YOLO) architecture in combination with their own mango dataset, creating the MangoYOLO architecture. The architecture outperformed several other algorithms such as Faster R-CNN ZF, Faster R-CNN VGG and SSD VGG. For training, testing and validating the model: 1300, 300 and 130 images have been used respectively. In addition to their own dataset, the model was also initialised with Common Objects in Context (COCO) pre-trained weights. The pre-trained weights show no significant performance gain over the MangoYOLO model. S.W. Chen et al. (2017) demonstrated a blob detection fully connected layer to segment potential fruit clusters from the background. Within each blob a convolutional network was used to count the apples. Lastly, a linear regression of the count estimate on the ground truth is performed. 71 images of oranges and 21 images of apples were used in the final dataset. In Bargoti and Underwood. (2017b) high-resolution images of apples, mangoes and almonds in orchards were acquired and used in a Faster R-CNN network. Both VGG-16 and ZFNet frameworks were used in separate models of which VGG-16 achieved better results. In addition, these results were superior to the pixel-wise CNN model used in Bargoti and Underwood. (2017a) to detect apples and mangoes.

Yet, none of the previously mentioned fruit detection methods used UAV images. In Apolo-Apolo et al. (2020a) the Faster R-CNN Inception Resnet V2 Altrous Coco model was used to generate yield estimation maps from apple orchards using UAV imagery. The images used were cropped to produce smaller images which resulted in 1000 images ready to train the model. In addition, data augmentation was used to increase the image total to 3000 pictures. In another research, Apolo-Apolo et al. (2020b) used the same Faster R-CNN to detect citrus fruits in UAV imagery of orchards.

1.3 Research Aim

The research objective of this thesis will be the setting up and evaluation of a deep learning algorithm that is able to detect apples in orchards from UAV-acquired RGB imagery and estimate the yield according to its prediction. The connection between deep learning and fruit detection has already been explored in the basics. However, little research on the detection in UAV-acquired RGB imagery has been performed. As a starting point, the existing algorithm of Apolo-Apolo et al. (2020a) will be used. In addition, the practical use of the currently available dataset will be investigated. The relation between image disturbances like, shadows, reflectance, unpruned trees etc., and the algorithms ability to detect apples will be explored. The dataset will be optimised accordingly and insights into the process of acquiring UAV imagery will be explored.

1.4 Research Questions

To get to the objective, the following research questions will be answered.

- How can the existing RGB UAV imagery dataset be used in apple detection?
- How to use deep learning to detect apples in orchards using RGB UAV imagery?
- Which image disturbances contribute negatively towards apple detection?
- Is it feasible to estimate apple yield from RGB UAV imagery?

2 Data and Methods

2.1 Data

2.1.1 Study Site

The orchard field of apples (Malus x Dornestica Borkh. Cv "Elstar") is located in Randwijk (latitude: 51°56'18.5"N; longitude: 5°42'24.8"E) near Wageningen in the Netherlands. The crop field is 0.47 ha with 592 trees allocated distributed into 14 rows with approximately 41 trees per row and a pollinator tree every 10 meters with an average tree height of 3-meters. Rows were oriented NW-SE with a 3 meters spacing between rows and 1-meter spacing between trees. Crop management tasks (fertilisation, thinning, pruning etc.) were performed following conventional farm practices. Only in 2021 pruning was not performed. In Figure 1 the exact location of the orchard is shown.

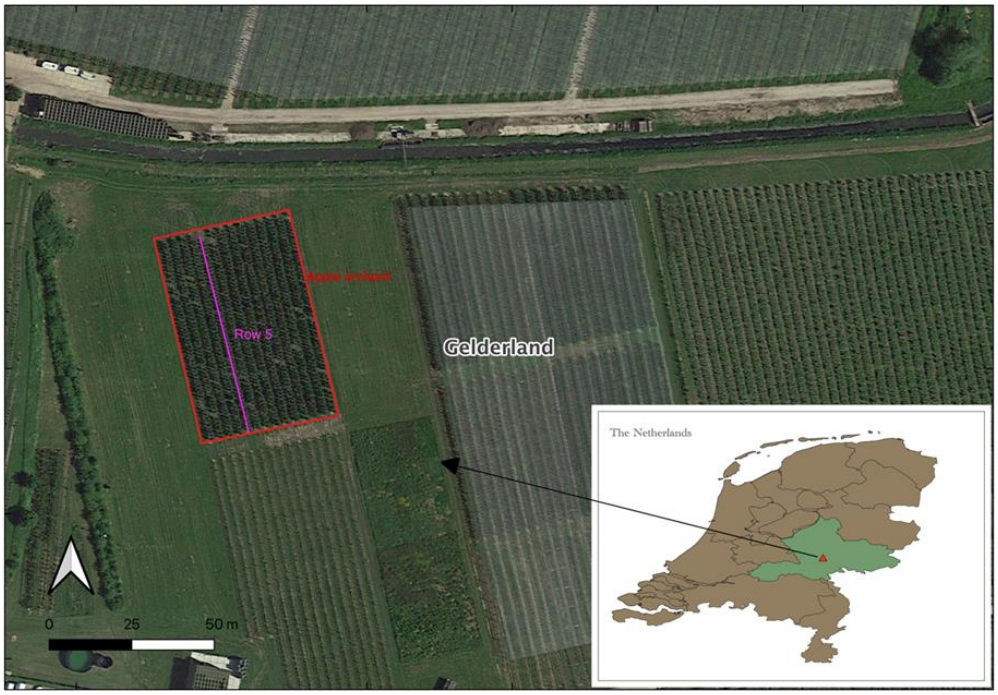

Figure 1. The location of the apple orchard: located at the province of Gelderland in the Netherlands.

2.1.2 Imagery Acquisition

The dataset, used in this report, contains RGB images of an apple orchard. They were acquired via different UAV platforms between 2018 and 2021. All images are stored in .JPG format. More details of the dataset are shown in Table 1. In Figure 2, exemplary images are shown for each year. The images have been cropped to a smaller size to preserve space and maintain structure in the document. For each year one full image is visible in Appendix A.

Table 1: Overview of the dataset per year and the total.

	Images	Size (GB)	Min Image Size (KB)	Max Image Size (KB)	Resolution (pixels)
2018	667	3,17	4,66	5,235	4000 x 3000
2019	353	2,79	7,816	9,04	5472 x 3648
2020	103	1,03	9,676	11,229	6016 x 4008
2021	396	3,15	8,031	8,749	5472 x 3648
Total	1519	10,14	-	-	-

Figure 2A. Image from 2018. Some brightness disturbances on the trees due to intense sunshine.

Figure 2B. Image from 2019. A viewing angle closer to the zenith (90° vertical) shows less apples.

Figure 2C. Image from 2020. Cloudy day which results in very dark shadows on the trees.

Figure 2D. Image from 2021. Unpruned trees which makes it hard to see apples.

All drones and cameras used to acquire images were made by DJI technology (SZ DJI Technology Co., Ltd., Shenzhen, China). During the four years in which images have been acquired, different UAV platforms, flying altitudes and cameras have been used. An overview of these differences can be found in Table 2. More specific drone information can be found in the URL referenced in the specifications column.

Table 2: Overview of the different UAVs, camera and specifications per year.

	UAV platform	Flying Altitude (m)	Camera	Specifications
2018	DJI Phantom 3 Pro	10	Onboard	SZ DJI Technology co., Ltd.
2019	DJI Phantom 4 Pro	15	Onboard	SZ DJI Technology co., Ltd.
2020	Matrice 210 RTK V2	25	Zenmuse X7	SZ DJI Technology co., Ltd.[1]
2021	Matrice 210 RTK V2	10	Zenmuse X7	SZ DJI Technology co., Ltd.[2]

[1] *Reference to URL of Matrice 210 RTK V2 specifications*

[2] *Reference to URL of Zenmuse X7 specifications*

The flight plan was designed in a grid shaped pattern using the DJI Ground Station Pro (SZ DJI Technology Co., Ltd., Shenzhen, China) iPad application, via which automatic flights for DJI aircrafts can be controlled and planned. The flights were made two days before the first harvest. All years had similar sun and wind conditions, except for 2020 where it was cloudy. The imagery from 2018 was obtained with a forward overlap of 85% and a sideways overlap of 75%. The UAV flight made in 2018 had to be made over a portion of the trees because the rest of the field had already been harvested by the farmer.

2.2 Apple Detection with Faster R-CNN

Training a neural network from scratch requires large datasets for learning and takes considerable computational power. Transfer learning can be used to overcome both problems (Gu et al., 2018). The main benefit of this technique is transferring the weights and therefore the knowledge of one model trained on a large dataset to another model. The weights have been trained on large datasets such as ImageNet (Deng et al., 2009) or COCO (Lin et al., 2014). Faster R-CNN will be used since this network can use several architectures, such as ResNet, Inception and Altrous. This heavily benefits the flexibility of the model and the eventual efficiency and precision of fruit detection (Dias et al., 2018).

Apolo-Apolo et al. (2020a) found that it is possible to detect the number of fruits in apple trees from UAV-acquired imagery. They used transfer learning and a Faster R-CNN and their model showed very promising values. In addition, the analysis in this report works with the same apple orchard. The dataset is partly the same, extended with newly acquired UAV imagery. These factors combined make it a logical choice to start with their deep learning algorithm of which they supplied the code as supplementary material. The algorithm consists of three steps. Faster R-CNN extracts feature maps from the image using a CNN. The feature maps go through a region proposal network (RPN), which returns object proposals. Lastly, the maps are classified and bounding boxes are visualised in the images. In Figure 3 the steps are visualised.

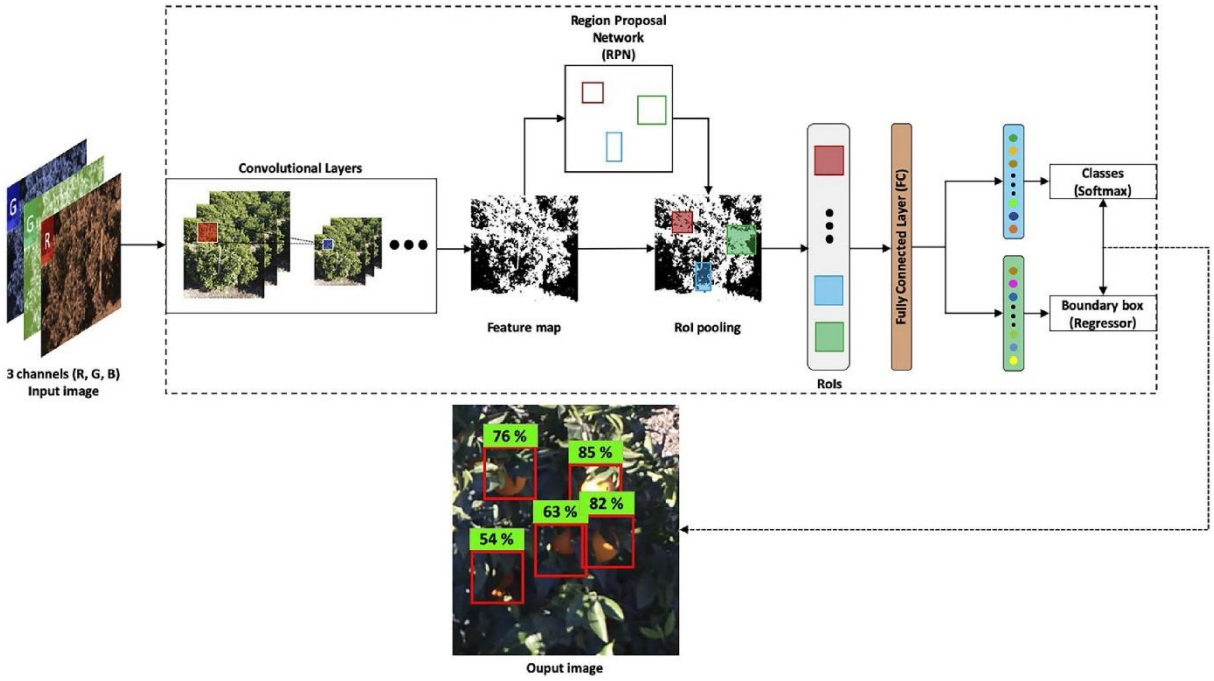

Figure 3. Faster R-CNN architecture

To ensure the algorithm will achieve the best result, the possibility of splitting up the dataset will be investigated. Subsections will still contain enough images to use for training and analysis of the model. An example of the exclusion of data could be the images from 2021. The orchard was not pruned

(Figure 2D above) by management and therefore these images might not be suitable for further use. Since multiple years of data are available, it is possible to divide the dataset into subsections if necessary. The eventual approach towards the selected data will be explained in section 2.3.1 Image Selection.

The construction of the algorithm should have been a simple copy from Apolo-Apolo et al. (2020a). However, after trying to access the supplied code to the algorithm, it turned out to be inaccessible. After contacting Apolo-Apolo about this problem for a solution, an unhelpful response was received and the data remained inaccessible. Therefore, no opportunity to copy their exact approach in regard to algorithm and labelling was possible. Since steps to work with faster RCNN were already made, the current approach was continued. The main objective was to get an instance of Faster RCNN running. After exploring multiple repositories, a mix of two sources was combined to end up with the used algorithm. The general coding structure for loading and preparing data and the algorithm was retrieved from PseudoLab (PseudoLab Tutorial Team., 2020). On top of this the training and validating process and extra functions were retrieved from debuggercafe (Ranjan Rath, 2021). This was mainly done because this better visualised the training process and made it easier to store the loss values and best performing models for later evaluation. The pre-trained Faster R-CNN model with a ResNet-50 backbone and Feature Pyramid Network was used as a starting point.

2.3 Pre-processing

2.3.1 Image Selection

The initial approach of this research was based on multiple years. Since the dataset consists of multiple years of data, it is important that every year is present in the training, validation and test data. Therefore, stratified sampling is necessary to ensure that each subset contains the same ratio of data from each year. In this case, the data will be split into 80% training, 10% validation and 10% test data. Since the drone flew over the orchard in a grid shaped pattern, all trees are visible in different images under slightly different angles. Ideally all images containing a single tree should be included during the training process. However, this quickly leads to a large increase in the workload necessary to prepare the images. During the annotation process, it quickly became clear that this approach was not feasible.

This led to two major changes. At first, only a single year was used in the continuation of this research and secondly, a subset spatially covering the whole orchard was selected. An example of a set of images still covering the whole orchard can be seen in Figure 4. The choice for the year 2020 over the other years was simply made because annotation was already done for a part of these images. The final dataset consists of five images taken in 2020.

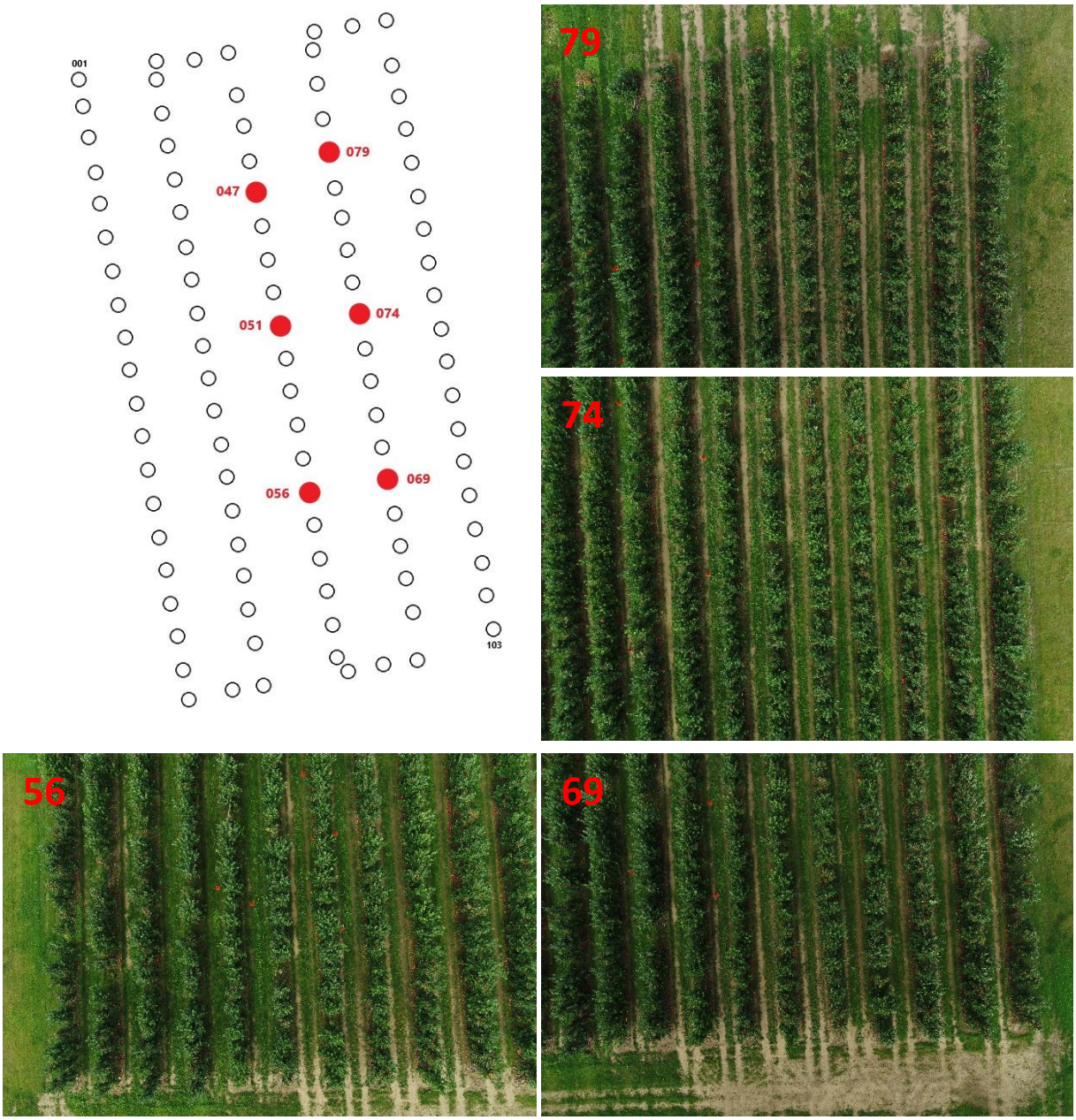

Figure 4. Spatial distribution of images taken over the orchard with a subset of images that spatially still covers the whole orchard, deemed sufficient coverage for continuation.

2.3.2 Annotation

Training a CNN requires a large amount of annotated images with the coordinates of each fruit on the images in the training dataset (Rahnemoonfar and Sheppard, 2017). Images will be labelled in the same way as Apolo-Apolo et al. (2020a). The free open-source labelling tool called LabelImg (v1.8.3) (Tzutalin, 2015) will be used. This process has to be done manually and very carefully to prevent mislabelling. This is a time-heavy process. Once all fruits have been labelled with a bounding box, an Extensible Markup Language (XML) file in PASCAL Visual Object Classes (VOC) format will be generated with the use of Roboflow.

Efforts to get the exact annotation strategy used in Apolo-Apolo et al. (2020a) turned out to be fruitless. This led to a new annotation strategy. Since the annotation part was a combined effort, a multiple-class annotation approach was chosen. This makes the resulting dataset universally useful. The annotation strategy consists of four classes: apple, apple on the ground, occluded apples and

difficult. A more detailed meaning of the labels can be seen in Table 3. A visual example can be seen in Figure 5. The model used in this research needs to detect apples. Classification can overcomplicate the data and is unnecessary. Therefore, all four labels were combined into a single class, apple.

Table 3 Further explanation of labels.

Label	Criteria	Examples, See *Figure 5*
apple	Clearly visibly apple ≥50%	Purple
apple on the ground	Apples laying on the ground	Red
occluded apples	Apples "hidden" by other apples or leaves	Yellow
difficult	Not sure if it is an apple.	Pink

Figure 5. Screenshot of small part of original image and the corresponding labelled image. Purple boxes for apple, red boxes for apple on the ground, yellow boxes for occluded apples and pink boxes for difficult.

2.3.3 Augmentation

To improve the performance of the deep learning algorithm, a dataset needs to be augmented. This results in a more robust model. The augmentation can also be done with the use of Roboflow. Initially, the images were going to be rotated by 90°, 180° and 270° degrees. However, after the first few try-outs training the algorithm, augmenting data turned out to increase the training time too much. Especially considering the planned augmentation steps in regard to image disturbances. These augmentations are with a specific reason and are explained in section 2.4 Image Disturbances. Besides these specific augmentations, no general augmentation steps have been performed except for the cropping and resizing of images. Firstly, due to the original image size being 6016 x 4008, the images were cropped into 54 images of 668 x 668 pixels to improve calculation speeds. Where the initial idea was to mimic the size used in Apolo-Apolo et al. (2020a), which would have been 416 x 416 pixels, the images were resized to 640 x 640 pixels instead. This was done since the creation of the dataset was a combined effort. The other party is using the dataset with the YOLO algorithm which needs the specific 640 x 640 pixels size and this also works for the Faster RCNN algorithm.

2.4 Image Disturbances

2.4.1 Training Datasets

To simulate image disturbances within the dataset, four different datasets will be created. The four datasets will be called Base, Reflection, Shadow and Combined. The base dataset contains just the cropped original images without any augmentation. The other three datasets are augmented in Roboflow. The reflection dataset will be augmented to make images brighter to simulate reflections from the sun. The Shadow dataset will be augmented to make images darker to simulate (dark) shadows on the trees. Lastly, the combined dataset will be augmented with both darkened and brightened images to see if a combination of both augmentations improve the results. During the augmentation process, the initial number of images is multiplied by three resulting in larger amounts of data. An overview of the specific augmentation steps per dataset can be seen in Table 4. The algorithm will be trained on each training dataset resulting in four separate models.

Table 4: Overview of the four datasets used to train the four models.

Dataset	Augmentation Steps	Number of Images	Resembles
Base	-	188	Original Orchard Images
Reflection	Brightness: Between 0% and +85%	562	Reflection of the sun
Shadows	Brightness: Between -85% and +0%	560	Shadows within trees
Combined	Brightness: Between -85% and +85%	562	Reflection and Shadows

2.4.2 Testing Datasets

To test the four trained models, four test datasets have been created following the same steps used in the training process. At first the base test dataset, which has been augmented three times to create a reflection, shadow and combined test dataset to end up with four test datasets. In Table 5 an overview of the specific augmentation steps per dataset can be seen. All four models will be tested on all four test datasets resulting in 16 different performances which can be compared to see which model performs best and give a basic insight into the influence of reflection and shadows on the detection of apples.

Table 5: Overview of the four test datasets used to train the four models.

Dataset	Augmentation Steps	Number of Images	Resembles
Base	-	21	Original Orchard Images
Reflection	Brightness: Between 0% and +85%	63	Reflection of the sun
Shadows	Brightness: Between -85% and +0%	63	Shadows within trees
Combined	Brightness: Between -85% and +85%	62	Reflection and Shadows

To illustrate the augmentation steps used in the creation of the training and test datasets, two images close to the maximum and minimum augmented brightness value can be seen in Figure 6. Both images come from different image tiles. Therefore, they only contribute in regard to the visualisation of brightness augmentation. The exact brightness augmentation value is unknown, but visual inspection of the datasets showed that both are close to the maximum value of +/- 85%. In the darkened image only the brightest leaves are somewhat visible and several apples can be seen in the tree tops. In the brightened image more features of the trees and apples are still visible, but several apples seem to lose their distinct red colour.

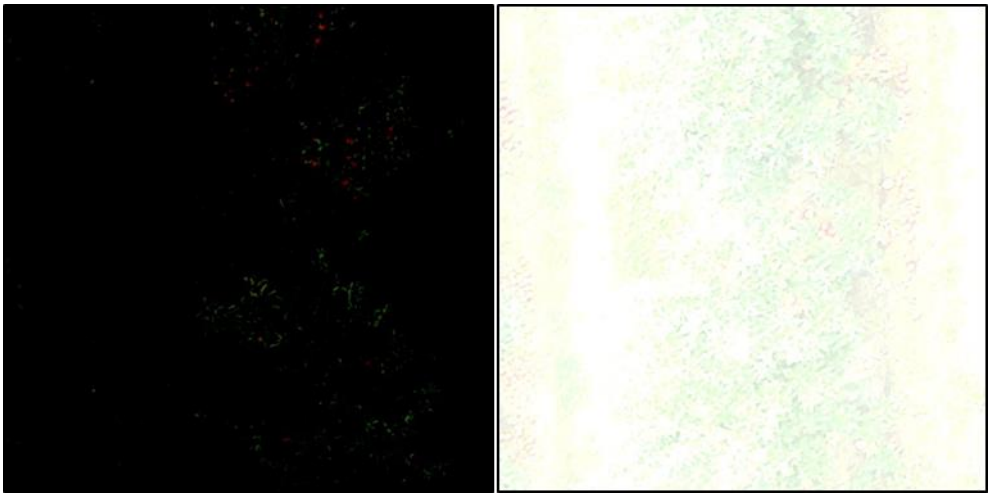

Figure 6. Example images of brightness augmentation close to the maximum values of +/- 85%.

2.5 Evaluation

2.5.1 Computable Parameters

The models will be evaluated via performance metrics. The precision, recall, average precision and F1-score will be calculated. These metrics are defined as shown in Equation 1, 2, 3 and 4 respectively:

$$\text{Precision (P)} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall (R)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$\text{Average Precision (AP)} = \int_{\text{R}=0}^{1} \text{P(R)dR} \tag{3}$$

$$\text{F1 Score (F1)} = 2\text{x}\frac{\text{PxR}}{\text{P} + \text{R}} \tag{4}$$

Where TP stands for true positives, i.e., when an apple is identified correctly, FP stands for false positives, i.e., when an apple is identified in a location where no apple is present, and FN stands for false negatives, i.e., when an apple is not identified.

Depending on your stance against FPs or FNs all metrics can be used to select the preferred model. The F1-score is a machine learning metric that is a proposed improvement of two simpler performance metrics, precision and recall. This works best when FPs or FNs are not specifically interesting. In addition, it is commonly used as a metric to evaluate deep learning algorithms as shown in Koirala et al. (2019b). Therefore, the model can be compared with other similar models via the F1-score. If FPs are a problem for your research, it is best to go for a higher precision. If FNs are a problem, recall is the preferred metric. For this research the focus was on selecting the best performing model during training. Therefore, the average precision was used since this results in the more stable and consistent model.

2.5.2 Specified Parameters

To calculate the performance metrics of the models, some parameters have to be defined. The intersection-over-Union (IoU) is an important factor in all the performance metrics for object detection. The IoU is the ratio of the intersection of the bounding boxes to the union of the bounding boxes, visualised in Figure 7. One box for the prediction and one box for the ground truth. A value > 0.5 is generally considered "good". It is difficult to preliminary decide on an IoU value before the research. During analysis, the IoU was set at a value of 0.6. In this case, values above 0.6 are considered correct.

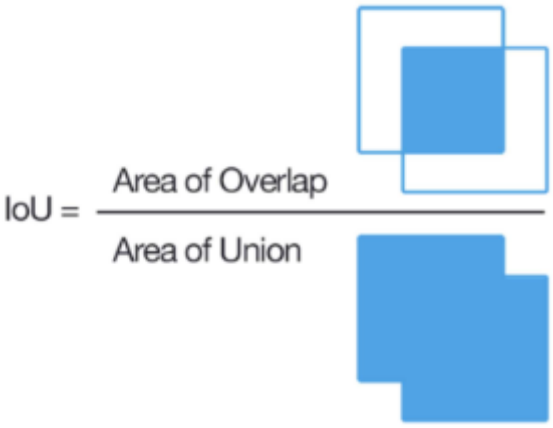

Figure 7. Intersection-over-Union visualised for two bounding boxes.

A second input parameter that has to be defined is the confidence threshold. The model needs guidance on the base of which it can decide to include or discard certain predictions. This threshold was also set manually and was checked for several values. A high confidence interval results in a lower amount of prediction, but should result in better predictions. Since there are no acceptable ranges of precision and recall beforehand only a visual inspection was done in relation to setting the threshold. For this analysis a confidence threshold of 0.25 was used.

2.6 Apple Yield Estimation

2.6.1 Prediction

After testing the 4 models, the best performing model will be selected to continue towards yield prediction. Analysis towards the relation between by the model detected apples and the true apple count will be investigated. Since the model is trained only on images in 2020, only images of 2020 will be predicted. The ground truth consists of hand-harvested and counted apples. For the year 2020 apple counts for 24 trees are available. These 24 trees will be identified in the original images. For each tree one manual image cut-out will be made. Due to the cropping and resizing steps in the pre-processing of the images used for training, it is important to work with the same dimensions for the

individual tree images used here. The individual trees are already smaller than cropped images. Therefore, a black border up to 668 x 668 pixels was added. After this step, they were rescaled to 640 x 640 pixels with the help of Roboflow. Now the images are ready for the algorithm to predict the amount of apples for each tree.

2.6.2 Regression

As a last step towards yield estimation, a relation between the ground truth and the amount of predicted apples will be investigated. With the prediction of the amount of apples for the 24 trees available and the corresponding ground truth, regression analysis can be used to analyse the relation between the two counts. The regression results in a metric the shows how strong the relationship is between the two variables, in this case hand-counted apples and algorithm predicted apple. A simple linear regression will output a formula and the statistical measure R-Squared (R^2) that quantifies the estimative power of the model. In general the R^2 value falls between 0 and 1. Higher values indicate a model with a higher predictive value for apples and thus apple yield.

2.7 Flowchart

As a summary, a flowchart of all steps taken during analysis and research performed for this report is added in figure 8. The figure was created after carrying out research and analysis. Therefore, some initial steps as described in the methodology are not included.

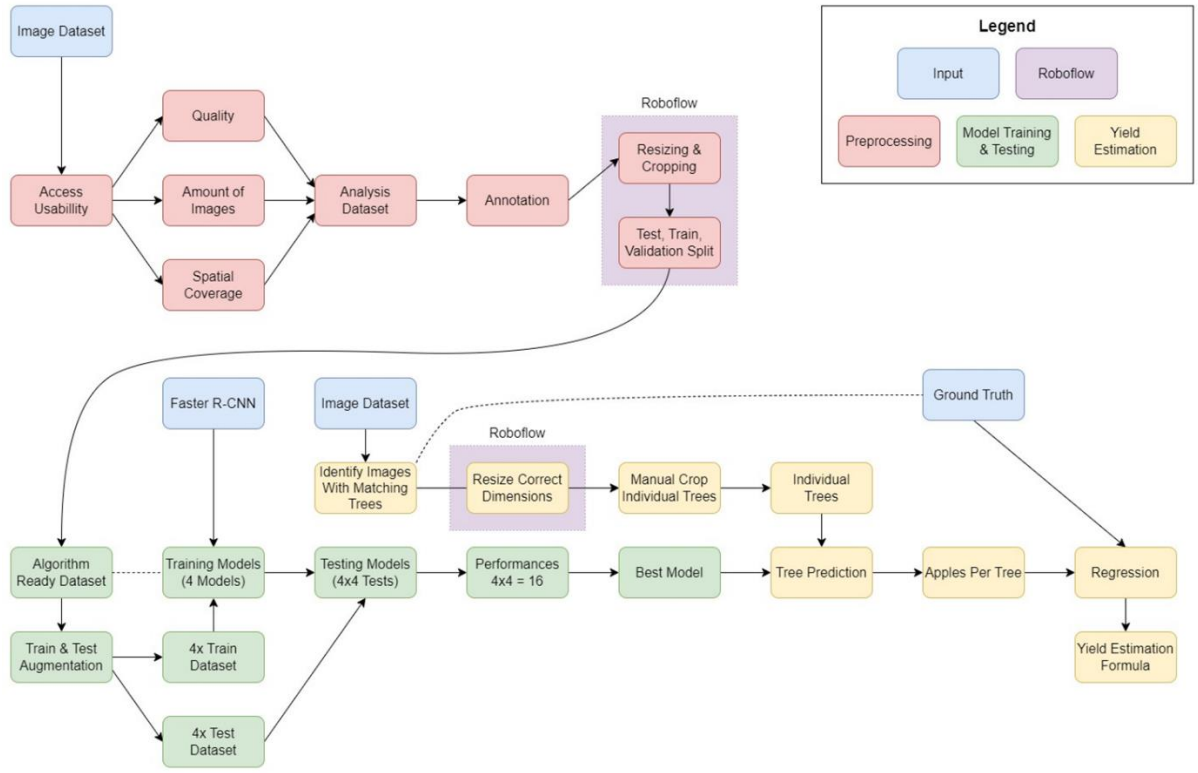

Figure 8. Schematic overview of performed steps during analysis and research.

3 Results

3.1 Image Selection

With deep learning datasets, the problem is often a lack of data for training. This is not the case for the dataset used in this report. With more than 1500 total images there should be plenty of data to be used in apple detection. On top of this the dataset contains years with different circumstances which can help improve the overall robustness of a trained model. The dataset exists of RGB UAV images of an apple orchard. This is what makes the dataset interesting to analyse, but also brings a problem with it. Due to the distance to trees and more importantly, distance towards the individual apples, a high spatial resolution is needed to capture details. This results in original images containing a lot of information. A large number of total pixels, but also many apples within a single image. Part of the information of interest, in this case apples, is only visible zoomed in, reducing resolution. This complicates apple detection. Performing cropping and resizing before annotation is an easy workaround to prevent zooming in, but does not solve the amount of pixels per apple.

The original images on their own are not suitable to be used in training an algorithm for apple detection. Before the images are suitable they have to be annotated and cropped. The cropping process links back to the resolution of the images. High resolution on orchard level, low resolution on apple level. The annotation process turned out to be a very time-heavy process. As explained in section 2.3.1 Image Selection, only 2020 has been included in this analysis. This does not hurt the use of the dataset for apple detection on its own, but resulted in less data being taken into account during the analysis in this report. This resulted into an analysis ready dataset of 270 image tiles. These 270 image tiles are coming from only 5 original images from 2020.

3.2 Annotation

Annotation was done with the help of LabelImg, which has become part of the Label Studio community. For each apple a bounding box was drawn over the apple. The bounding box should be as precise as possible, touching the edge of the apple on each side of the box. The ability of LabelImg to zoom in to great depths, helped with minimizing oversized bounding boxes. LabelImg was easy to use and fast at the start. On top of this, it gives multiple formats as options to export your labels. Since Faster-RCNN is used in the analysis, the labels were exported in PASCAL VOC. However, due to the image sizes and the total amount of apples to be labelled in a singular image, the software started to slow down the more labels in an image. When using LabelImg, consider the use of smaller images to increase labelling speed.

As mentioned, annotation was one of the main reasons to reduce the amount of images used in the analysis. Partly due to the software slowing down, but mainly due to the large amount of labels in a single image. The final subset of images consists of five images and these contain a total of 18,754 labels. Labels within a single image, ranging from two to four thousand. The total amount of labels used in the analysis is 18,754. The distribution over the four classes can be seen in Table 6. Before analysis started it was decided to combine all labels into a singular class. This was done because this report is focused on the first stage which is apple detection and only at the second stage, classification becomes relevant.

Table 6: Distribution of labels used in analysis.

Label	Number of labels	Percentage
apple	1902	10
apple on the ground	10286	55
occluded apples	5056	27
difficult	1510	8
total	18754	100

3.3 Algorithm Performance

The Faster R-CNN algorithm was trained four separate times resulting in four different models; base model (a), reflection model (b), shadow model (c) and combined model (d). Model specifics can be found in the methodology, Table 4: Overview of the four datasets used to train the four models.. An overview of the loss values during the training process for the four models can be seen in Figure 9. The training and validation loss per epoch was calculated and visualised. The red dots in the figures are the best performing models at that current moment. Since a pre-trained algorithm was used, it was expected to see quick improvements in the first few epochs. The quick improvement is clearly visible for all four models. After several epochs, the training improves in small steps for all four models. However, after 40 to 50 epochs the models stop improving and the optimal performing model is reached. The models all show strong signs of overtraining, especially clear after epochs 40 to 50. The overtraining can be due to a limited amount of data compared to the complexity of the data. However, model (b), (c) and (d) all contain augmented images and therefore, triple the amount of total images. This suggests that this is not the problem for this dataset and model. Another reason for the overtraining could be a case of complex data. A third reason could be a suboptimal labelling strategy. Since only 10% of the labels are clearly visible apples it might be that the other labels are too complex or not useful at all.

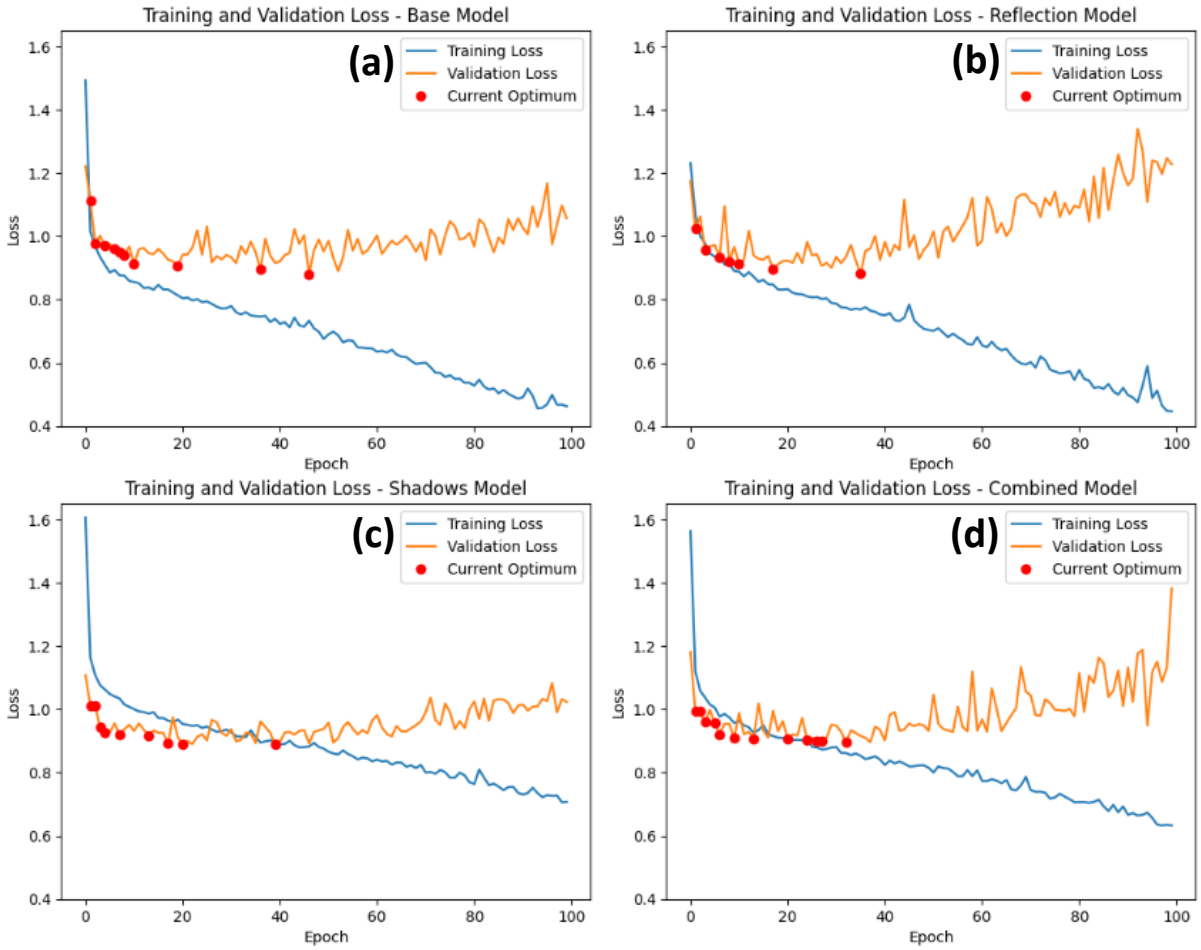

Figure 9. Overview of the four trained models and their respective loss values during training. The Red dots show the best performing models at that current moment.

Another interesting difference between the behaviour of the models is the faster increase in difference between training and validation loss for model (b). Training loss seems to decrease and validation loss seems to increase quicker than the others model. On top of this, model (d) has larger fluctuations,

similar to model (b) and the distance between loss values is closer to model (c). This could be due to the combination of both augmentation factors.

The problem with overtraining does not mean that analysis stops here. For all four models, the optimal performing model has been selected and tested on four test datasets. The result of the performances expressed in average precision can be seen in Figure 10. A first more general look at the figure shows that all four models get their best result on the base test dataset with model (a) outperforming the other models. Augmenting the datasets and training the separate models does not result in a better model. Since the models are already prone to overtraining, most likely due to the data being too complex, the augmentation of data might add to the complexity and result in a lesser overall performance of the models. Secondly, it can be seen that model (d) structurally outperforms models (b) and (c) on any test dataset. It also outperforms model (a) on the three augmented test datasets, but not the base test dataset. This is interesting because, even though augmenting the data seems to complicate the data in a way that hurts performance, it still gives insight into the benefit of augmentation. A model improves in performance if the opposite image disturbance is added in the training process. E.g., a model trained with only shadow augmented data benefits from adding reflection augmented data and a model trained with only reflection augmented data benefits from adding shadow augmented data. The fact that it works in both directions suggests that it is indeed the combination of data that has a beneficial effect on model performance.

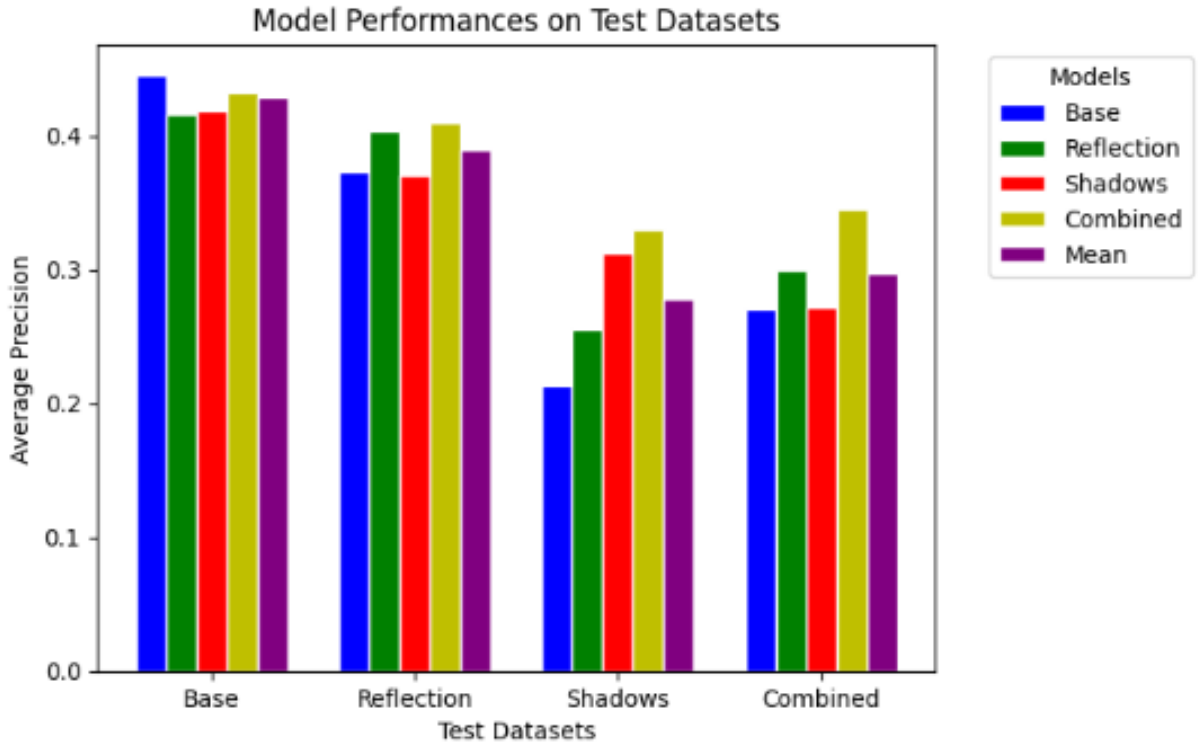

Figure 10. Model Performances expressed in average precision. Each models vs each test datasets.

The best model is model (a) on the Base dataset. The performance statistics of this model can be seen in Table 7. Since it is important to be sure about the amount of apples predicted, it is nice to have a higher precision. For the continuation towards yield estimation, a precise amount of apples is more relevant than recall. The statistical performance metrics are calculated for each model based on each test dataset. An overview of all these metrics is added in Appendix B. Were the selection based on average precision makes sense beforehand, other performance metrics could have been used to make the selection. F1-Score or Precision might have been the better choice of metric in hindsight.

Table 7: Performance statistics of the base model, model (a). The best performing model.

Precision	Recall	F1-Score	Average Precision
0.64	0.50	0.56	0.45

To demonstrate what this model is capable of doing, two images out of the test dataset have been predicted by the model and have been visualised in Figure 11. In these two images, the model has predicted locations of apples. In the first image on the left, blue area (1) highlights an inaccuracy around an orange pylon hidden behind the tree. The yellow area (2) in the second image on the right shows occluded apples. The model is unable to identify these apples even with a confidence interval of 0,25. Also in the right image, in the pink area (3), several apples have not been identified even though they have a rather high visibility. Lastly it can be seen, mainly in the right figure that the model predicts many apples on the ground. These have been included in the annotation and thus learning process, but are not apples that are going to be harvested by a farmer and will not be counted towards yield. Even though this is the best-performing model, a quick assessment of images shows there is room for improvements in relation to apple detection.

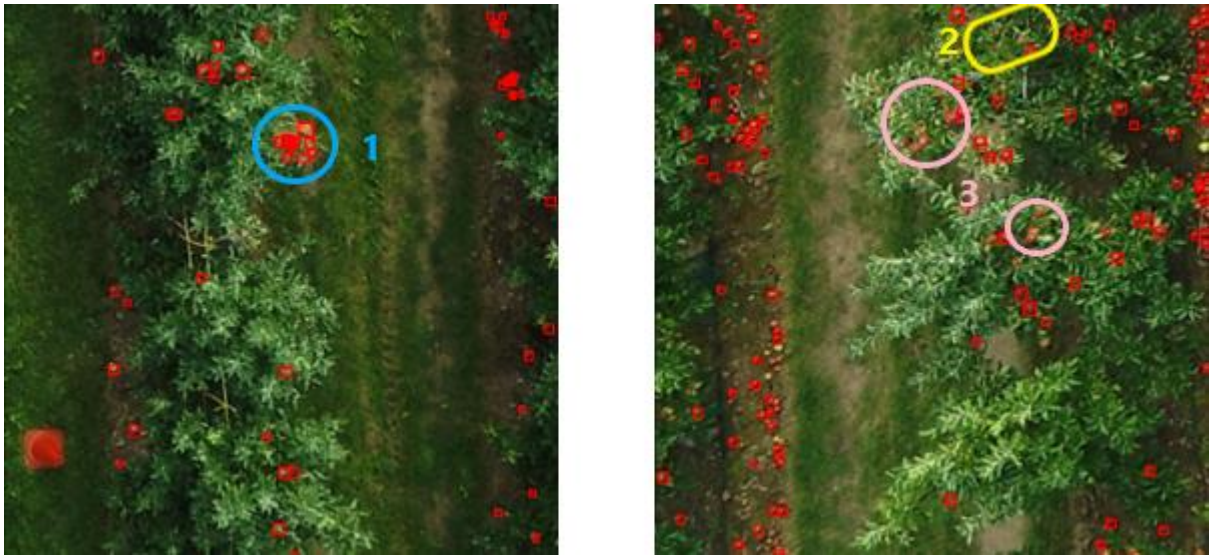

Figure 11. Two images in which the boxes are places were the model predicts an apple. The prediction was done with the aforementioned confidence interval of 0,25.

3.4 Image Disturbances

Focussing on apple detection in relation to reflection of the sun or dark spots due to shadows, shadows seem to have a bigger negative impact on apple detection. All models except model (c), which has it the other way around, have their lowest performance score for the shadow test dataset and their second lowest score for the combined test dataset. These performance scores are on average more than 25% lower than the performance scores on the base and reflection test dataset. This information can be used to pick a day of acquiring images for apple detection. Considering the performances visualised in Figure 10 above, the sun should be at an angle that minimises shadows and solar reflections should be accepted. On top of this it would be a good idea to make sure the apple trees are pruned properly to minimise shadows by leaves and other objects in the orchard.

3.5 Apple Yield Estimation

3.5.1 Tree Cropping

Before the amount of apples can be predicted, the trees of which ground truth is available had to be identified in the original images. Afterwards the trees had to be cropped out of the image and resized

to properly fit into the algorithm. Now apples can be predicted within the images. In Figure 12 the tree identification is highlighted. In Figure 13 the three steps of cropping a singular tree to an image with apple predictions are visualised. In total this process was performed for 24 trees.

Figure 12. Tree identification in original image on the background. In the front is the true scale of the image with the tree ids.

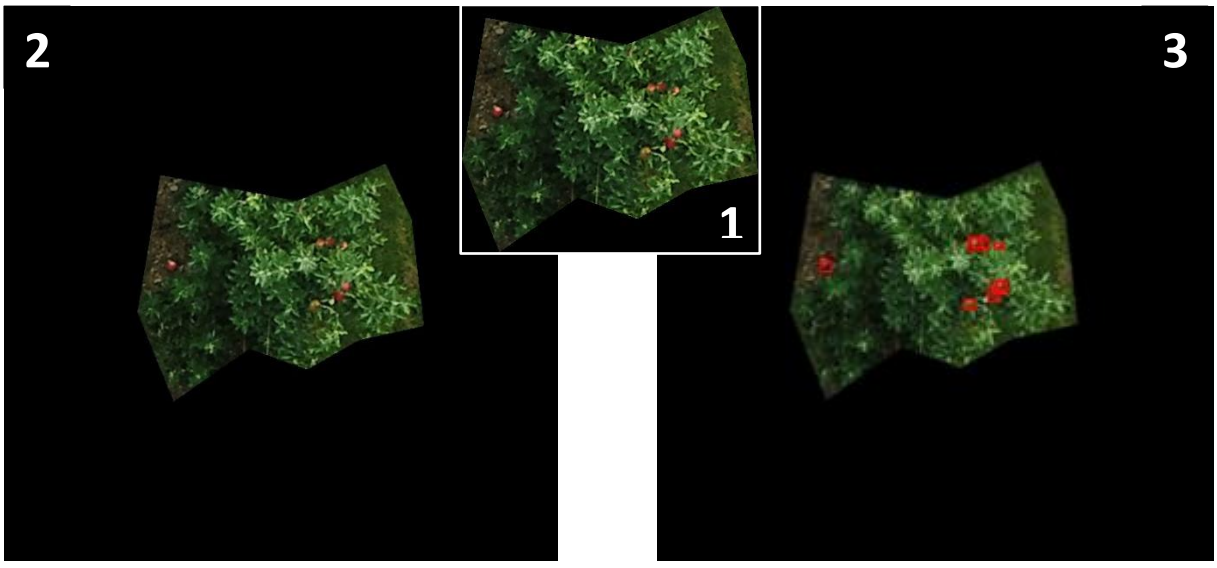

Figure 13. Cropping of a single tree. First step is a crop out of the original image resulting in 1. Image 2 is the result of adding the black boarder to ensure correct sizing in regard to the algorithm. Image 3 is the prediction of the apples.

3.5.2 Regression

With the best performing model, which is model (a) the base model, the amount of apples is predicted for the orchard. Only for trees with an available ground truth, the amount of apples is predicted. The

regression can be seen in Figure 14 below. The amount of apple counted in the trees as a function of the model predicted amount of apples. The regression line is also visualised within the graph. As can be seen, the pattern of the regression line clearly follows an upwards trend which means the more apples the model detects the more apples can be expected when harvesting fruit. However, the R^2 of the regression is only 0.3861 which indicates there is some value in the prediction of fruits harvested based on the apple detection by the model, but it is low. Depending on the necessity of a high accuracy the prediction might be insufficiently accurate. There is most likely too much room for error to plan harvesting operations within the orchard solely based on the prediction of this model. Another interesting thing to highlight is the difference between the actual fruits vs predicted fruits. The model can only detect a few apples resulting in a difference of one order of magnitude. This is not necessarily a problem, but combining the findings together suggests a model able to detect a higher percentage of apples will likely lead to a direct increase in the regression and predictive value.

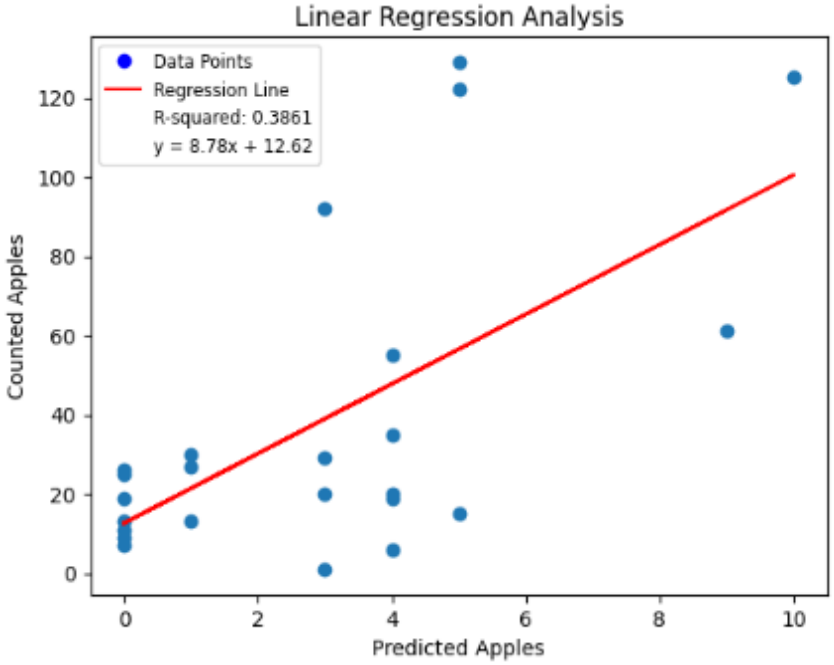

Figure 14. Linear Regression Analysis. The amount of counted apples vs the amount of predicted apples.

4 Discussion

4.1 Dataset

4.1.1 Images

For the use of the existing RGB UAV imagery dataset in apple detection the dataset was investigated manually to assess usability. This quickly led to the fact that the dataset as a base can indeed be used in apple detection. The dataset contained enough images and contains multiple years adding to the possibilities with the dataset. This is also in line with literature. This dataset contains 1519 images, whereas other papers used fewer images in their analysis. Sa et al. (2016) used 61 images and S.W. Chen et al. (2017) used 71 and 21 images. This dataset is more in line with the amount of images of Koirala et al. (2019a) who used 1730 images. One major flaw in this comparison is the usability of images and the actual use of the images. For the analysis in this report, only 5 images of the dataset have been used. This would not be sufficient, but these are very large images resulting in 270 cropped images that make up the eventual trainable dataset which is a sufficient amount of data. It is clear that the number of images is not strictly comparable with another number of images.

Due to the fact that only a few images from a single year were used in the analysis in this report, the speculative side of the dataset or images did not arise during the analysis. Most trees are visible in multiple images and under different angles. It is not clear if this is a positive or a negative. Does this mean only a few images that cover the whole orchard are enough to mimic the whole orchard or do all angles for every individual tree have to be included. Given that acquiring images with UAVs is quick and the model is meant to focus on this method of acquiring data, it might be perfectly fine to use the current approach of only using a tiny subset of the dataset to train a model. This plays right into the positive benefits of UAV-acquired images, but leaves room for possible overfitting towards singular trees that just happen to have a better angle in the image to detect apples. A solution for this problem would be the inclusion of all images in the dataset. This ensures that each tree is available in multiple angles. This would prevent a bias to specific angles or positions in images. A nice idea, but this would take an unreasonable amount of time for a speculative improvement of the eventual apple detection.

Lastly, only the year 2020 was used in the analysis. In hindsight this is possibly not the most ideal choice. As mentioned in the methodology Table 2: Overview of the different UAVs, camera and specifications per year. the images of 2020 have been acquired from a higher altitude than the other years. Even though the resolution is also slightly higher, the resolution for individual apples is lower. On top of this the results in relation to image disturbances might suggest that a lighter condition during image acquisition would be beneficial. It could also have been worse since the orchard was not pruned when images of 2021 were taken. This most likely results in all images being useless for apple detection. At the point of writing this, the images of 2018 are most easy to work with. An example of the raw images of each year can be found in Appendix A

4.1.2 Annotation

Were the amount of data is the first step in a usable dataset for apple detection, the second step is the preparation of data. Annotation was the most time-heavy process during the creation of this report. Annotation did not get the attention it needed before starting this process. The main reason for this oversight was the initial approach to copy the annotation strategy of Apolo-Apolo et al. (2020a). On top of this, inexperience with annotating data also contributed to this. Due to the switch to a self-made annotation approach several questions remain connected to the performance of the models. Most notably is the question about the usefulness of the classes. For this analysis, apples that will end up as yield are relevant. Other apples are not. Therefore, two of the classes, apple and occluded apples are logical classes to be included into the analysis. However, apples on the ground are not interesting at all towards the eventual aim of yield estimation. Therefore, it does not make sense that this class was

included into the combination of classes for this analysis. Lastly, the difficult class was used as a leftover class for when it was too difficult to say if something was an apple or leave discolouration or something else. The use of this class could be the possibility of the algorithm being able to correctly recognise if it was indeed an apple.

Another point of discussion is the subjective nature of annotation. Even though the same annotation strategy can be used, some cases will be assessed differently. As an example the apple class consists of apples with a visibility equal or larger than 50%. Were most apples will clearly fall in or outside this class, some cases might end up in different classes depending on the person annotating. Due to this subjective nature of annotating, it is important to compare variance between operators. In the case of large differences, possible changes in annotation strategy have to be done or part of the annotation has to be reassessed.

Individual assessment of classes was not performed during this analysis. During the set-up of the algorithm however, one small trail run was performed to see if the model worked from front to end. In this trial four separate classes were used. These results have not been investigated due to the decision to combine all four classes into one due to the interest in detection and not classification during this analysis. These very raw results showed the following detection accuracy for the apple (70%), occluded apples (23%), apple on the ground (29%) and difficult (0%) classes.

A better approach to the annotation strategy would have been a better selection criteria of which apples were going to be labelled. Were apples and occluded apples are relevant, apples on the ground is not and the difficult class seems to contribute nothing. With this in mind, the annotation strategy could have been limited to only apples and occluded apples resulting in a 63% decrease in labels, calculation based on Table 6: Distribution of labels used in analysis.. The time save here could have been used to test detection accuracy for a different year or additional images could have been annotated.

4.2 Algorithm

4.2.1 Faster R-CNN

The use of deep learning and more specific Faster R-CNN is not new within the world of object detection. There are many models to pick from. For this report Faster R-CNN was used. Faster R-CNN was considered state-of-the-art 9 years ago when it was released, it is now considered deprecated. Detectron is its follow up which is also already deprecated and followed up by Detectron2 (Ren et al., 2015b; Girshick et al., 2018; Wu et al., 2019). The main reason to use Faster R-CNN was again the following of Apolo-Apolo et al. (2020a). After it became clear the use of Detectron2 was investigated, but due to inexperience with setting up a complete algorithm, the choice for Faster R-CNN was chosen. Were the set-up of Detectron2 resulted in problems not directly solvable, the Faster R-CNN set-up was done within a few days. Faster R-CNN being deprecated is not a problem. The existence of better models does not make older models futile. Of course, it would be nice to use better models, but in this case the use of Faster R-CNN was quicker and results in a one-on-one comparison to the results of Apolo-Apolo et al. (2020a). On top of this the annotation part was a combined effort. The other party used the dataset with a version of the YOLO algorithm. Here also lies an easy comparison since it is done on the same dataset. Faster R-CNN is considered more accurate and the general YOLO structure is considered faster (Tan et al., 2021). The trade-off of these two factors is interesting.

4.2.2 Performance

In comparison to Apolo-Apolo et al. (2020a), the metrics of the model used in this analysis are a lot lower. Apolo-Apolo et al. (2020a) managed to get a precision of 0.93, a recall of 0.90 and an F1-of 0.91. This is in contrast to the values of this report: precision 0.64, recall 0.50 and an F1-of 0.56 found in

Table 7: Performance statistics of the base model, model (a). The best performing model.. There are multiple explanations for a difference this large. One very clear difference between Apolo-Apolo et al. (2020a) and this paper is the year out of which images are used. As mentioned earlier in the discussion, this report used 2020 and in hindsight 2018 has multiple factors which seem beneficial to apple detection. Apolo-Apolo et al. (2020a) did use images from 2018. The image is closer, light conditions are better and more apples are visible in the trees. The difference between the years is so large that this could very well be the cause of the underperforming model in this report.

Secondly, the annotation strategy in this paper contains a lot of labels not useful for the model. Apolo-Apolo et al. (2020a) does not explain their labelling strategy very well except for the software used which was copied for this report that they did it manually and carefully. They also did not want to have overlapping bounding boxes which is an interesting choice. The reasoning behind this is not explained, but apples close or overlapping each other, will need an overlapping bounding box in some cases. This can influence the performance in benefit of Apolo-Apolo et al. (2020a), but this is unlikely to have a large influence on these performance statistics. In a picture that shows one annotated image, no apples on the ground have been labelled and several hidden apples also have not been labelled. This also works in the benefit of Apolo-Apolo et al. (2020a). Excluding harder to find apples will naturally push your performance metrics higher.

In comparison with other apple or fruit detection models using faster R-CNN or other models as a base for their models, the achieved F1-score of 0.56 is low. With Faster R-CNN as a starting point, Sa et al. (2016) reached an F1-score of 0.838 on sweet pepper detection and Bargoti and Underwood. (2017b) reached an F1-score larger than 0.9 for both apple and mango detection. With YOLO as a starting point Koirala et al. (2019a) reached an F1-score of 0.968 on mango detection. Bargoti and Underwood. (2017a) started with a CNN to detect apples and reached an F1-score of 0.791 with a pixel-wise approach and an F1-score of 0.861 with a segmentation approach. Even though the model in this report is outperformed by many and is lacking in its performance, the continuation towards yield estimation is valid. A model does not have to be perfect in its detection. If a model consistently predicts 20% of apples in a tree, linear regression will result in a very high relation between detected apples and eventual yield. Given that UAV images, which will not have a clear angle to each individual apple, this is also a very realistic approach.

4.3 Image Disturbances

To inspect the contribution or influence of image disturbances towards apple detection, the data has been augmented for two common disturbances. A first augmentation that brightens the images to simulate the reflection of the sun. A second augmentation that darkens images to simulate shadows within images. Lastly, a combination of both augmentations was carried out resulting in a third extra dataset and model. The choice to augment brightness between -85% and +85%, used in Table 4: Overview of the four datasets used to train the four models. & Table 5: Overview of the four test datasets used to train the four models, to simulate both disturbances is purely done visually. The +85% brightness includes the most bright spots in images and the -85% brightness includes the most dark spots in the images. This method visually makes sense, but there is no scientific backing of this method. There also is no easy way of simulating these disturbances which is exact. Although this method is not exact and it is difficult to say how precise the augmentation mimics the true characteristics of reflections and shadows, the lack of a scientific approach leaves the used method on which visual assessment leads to values the look sufficiently similar to the real images. Besides this, the augmentation of images results in far more bright spots and far more dark spots than is realistic in the original images. At a first glance this sounds like a negative, but this leads to the image disturbance influencing the detection process in larger quantities. Thus, resulting in a stronger connection between

disturbance and performance of the model. A different response of the model is more likely to be allocated to the disturbance.

In literature no direct research towards the shadows and brightness in regard to apple or fruit detection was found. Research towards night vision suggests that data with different illuminations should be modelled separately, to prevent interference during training (Xiao et al., 2020). This suggestion is a problem for apple detection in orchards based on UAV-acquired images because this guarantees a mix of illumination conditions within images. Another area that could yield a possible improvement is research towards object detection and illumination changes. This is often related to moving cameras or a moving surrounding (Cheng et al., 2011; Choi et al., 2012; Qu et al., 2019). Since the images used in this report are static, this area has not been investigated further to look for possible improvements.

4.4 Yield Estimation

To investigate the feasibility of apple yield estimation based on apples detected in RGB UAV imagery, the best performing model was used to detect apples of image crops containing one single tree of which the ground truth was known. This resulted into a relatively simple regression formula that can be used to predict the amount of apples in the tree. If left at this point, it can be used to estimate apple yield from RGB UAV imagery. However, to assess whether a regression formula will be sufficiently accurate to be of any use to a farmer is more difficult. The exact use to a farmer is unknown. For this model, with an R^2 value of 0.3861 from Figure 14. Linear Regression Analysis. The amount of counted apples vs the amount of predicted apples., the predicted regression formula will not be used by farmers. It simply is too inaccurate. As an example, a completely empty orchard will still get an estimated apple count of 12.62 apples for each tree in the orchard which is completely useless to a farmer. The 12.62 apples are based on the formula of the regression line in Figure 14. Linear Regression Analysis. The amount of counted apples vs the amount of predicted apples.. Even though the correlation is low, the approach used in this report saves time over using the method based on one orthophoto used by Apolo-Apolo et al. (2020a).

The detection model in this report has an insufficient detection rate of apples to be used in yield estimation. However, the fact that a rather poor model is still able to get somewhat of a relation heavily suggests that the concept of training a model, detecting apples and ending up with a formula that is able to estimate yield is far from useless. Combining the results in relation to the dataset, the algorithm, the image disturbances and apple yield together. The possibility of ending up with a front to end set-up to estimate apple yield seems feasible. However, key parts of the process have to be improved and investigated at their own before it is realistic that a farmer will get any use out of it. To name a few possible key parts, think about the tree/orchard conditions, image acquiring strategies specifically aimed towards apple/fruit detection in orchards, usage of state-of-the-art deep learning algorithms or the precision of apple detection in relation to apple yield estimation.

5 Conclusion

After concluding the research towards the development and evaluation of a deep learning algorithm that detects apples in orchards from UAV-acquired RGB imagery and using said algorithm to estimate yield, the following can be said.

5.1 Dataset

For the question how the existing RGB UAV imagery dataset can be used in apple detection, it became clear that the dataset is sufficiently large. There are plenty of images to be used in apple detection algorithms. On top of this several different years, with different circumstances leave room for multiple angles and investigate the differences between them. To get to a stage where the dataset is actually useful however, quite a lot of work has to be done. Especially in the annotation, part this dataset requires a lot of work before it can be used in apple detection with Deep Learning. Besides this, the selection of images is also important. Depending on the need, some data is entirely useless, most notably, images from 2021, were pruning was not carried out which results in a lack of visible apples to detect.

5.2 Algorithm

For the question how deep learning can be used to detect apples in orchards using RGB UAV imagery, it became clear that Faster R-CNN can be used to train a model that is capable of detecting apples. With an F1-score of 0.56 however the best-performing model did disappoint. Especially compared to previous research of Apolo-Apolo et al. (2020a), who manage to reach an F1-score of 0.91. During the training process, signs of overtraining were clearly visible. After several epochs of training all models showed signs that heavily suggest issues with overtraining. Since Faster R-CNN is deprecated by detectron and detectron2, this seems like a logical improvement. Even though detectron2 most likely will result in a better-performing model, the results presented in this report are more in line with a dataset that is too complicated.

Improvements have not been investigated in this report. However, using the knowledge about the dataset used in this report, a different annotation strategy can possibly improve performance. Using a different year of data, in this case 2018 similar to Apolo-Apolo et al. (2020a), will also likely improve performance. If these models will also perform better on the current will be an important test to assess if the improvements are structural.

5.3 Image Disturbances

To investigate the contribution of image disturbances towards apple detection, both shadows and reflection within images seem to negatively impact a models ability of apple detection. Out of the four trained models, the model trained based on darkened images, mimicking shadows, had the lowest performance scores. On top of this all four models performed best on the dataset without augmentation, supporting the negative influence of shadows and reflections on apple detection.

Testing all models on augmented test datasets that mimicked image disturbances showed that reflection has a small negative impact on apple detection and shadows have quite a large impact on apple detection. Since the approach taken during analysis is not exact, it is unclear to say in what magnitude shadows are an issue. What can be said is that shadows should be minimised over reflection.

Lastly, in combination with the algorithm training it is interesting to note that a model trained on purely shadow or reflection augmented data showed improvement when both factors were combined. The combination model performed best on the augmented training datasets, but was still outperformed by the base dataset on the base test dataset.

5.4 Apple Yield Estimation

To assess the feasibility of apple yield estimation based on RGB UAV Imagery, the best-performing model was used to detect apples which was then used in linear regression. With a resulting R^2 value of 0.3861 the result does not seem useful. Even though this is not valuable on its own, combining these findings with the previous results, the concept of estimating apple yield based on RGB UAV imagery deserves another look. The whole process explored in this report leads to multiple questions ready for further investigation.

6 Recommendation

Even though this paper on its own does not give conclusive proof of anything related to the concept of apple yield estimation based on RBG UAV imagery, it can serve as a base to explore multiple angles that can result in valuable contributions to this concept. The findings about the dataset, model performances and image disturbances all show signs of an overcomplicated dataset. Due to the strong signs of overtraining, it is likely that the limited performance of the model can be somewhat attributed to image characteristics linked to the process of acquiring images. This together can lead to recommendations in relation the process of acquiring images and several areas in which more detail focused research can contribute to the concept

Areas in which more detailed research would be interesting are mostly focused on the acquisition of images. In the comparison towards Apolo-Apolo et al. (2020a), the main difference was the use of the data from 2018 instead of 2020, which year was used in this report. At first a small difference is the difference in weather conditions and conditions of the orchard. In 2018 the orchard is pruned much better which improves apple visibility. On top of this, the weather was sunny which is beneficial over darker days which is supported by the result about image disturbances in this paper. A more difficult to assess difference is the settings of the UAV during image acquisition. Even though the resolution of 2020 is larger compared to 2018, the fly altitude is 2.5 higher, 10 to 25 meters. This results in a lower resolution per apple which hurts the detection rate. The fly height lends itself perfect to a focused research to fly height in relation to apple detection in an orchard. This is especially interesting since the benefit of UAV-acquired images is the speed and ease of acquiring images. If there is a set plan of circumstances and UAV settings that is suitable for apple detection a large jump towards apple yield estimation can be made.

Another area which is interesting to explore is the model precision necessary for accurate yield estimation. Were the yield estimation formula from this report shows problems, a tree without any detected apples still gets 12.6 expected yield. It is very inconsistent and a farmer will never use this to estimate yield. It would be interesting to see at what precision a model can detect apples consistently enough were a yield estimation regression ends up with a high enough accuracy for farmers to be interested. Theoretically, a model that is extremely precise and consistently predicts 20% of the apples in a tree, will perfectly translate to yield estimation. The higher the precision of a model, the less important its recall becomes. The relation between the precision and recall is very interesting to investigate. The aim can be to set a base threshold for model performances over which they become useful towards yield estimation.

Lastly, the visibility of apples is a known problem with UAV-acquired images. Especially since all the images have been taken from an angle close to the zenith, it can be interesting to investigate at which camera angle most apples are visible. This can serve as another base guideline when acquiring UAV images meant for yield estimation based on apple detection.

References

Aggelopooulou, K., Castrignanò, A., Gemtos, T., & De Benedetto, D. (2013). Delineation of management zones in an apple orchard in Greece using a multivariate approach. *Computers and electronics in agriculture*, *90*, 119-130.

Apolo-Apolo, O. E., Pérez-Ruiz, M., Martínez-Guanter, J., & Valente, J. (2020). A cloud-based environment for generating yield estimation maps from apple orchards using UAV imagery and a deep learning technique. *Frontiers in plant science*, *11*, 1086.

Apolo-Apolo, O. E., Martínez-Guanter, J., Egea, G., Raja, P., & Pérez-Ruiz, M. (2020). Deep learning techniques for estimation of the yield and size of citrus fruits using a UAV. European Journal of Agronomy, 115, 126030.

Bargoti, S., & Underwood, J. P. (2017). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, *34*(6), 1039-1060.

Bargoti, S., & Underwood, J. (2017). Deep fruit detection in orchards. In 2017 IEEE international conference on robotics and automation (ICRA) (pp. 3626-3633). IEEE.

Bulanon, D. M., Burks, T. F., & Alchanatis, V. (2009). Image fusion of visible and thermal images for fruit detection. *Biosystems engineering*, *103*(1), 12-22.

Chen, L. C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., & Adam, H. (2018). Masklab: Instance segmentation by refining object detection with semantic and direction features. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4013-4022).

Chen, S. W., Shivakumar, S. S., Dcunha, S., Das, J., Okon, E., Qu, C., ... & Kumar, V. (2017). Counting apples and oranges with deep learning: A data-driven approach. IEEE Robotics and Automation Letters, 2(2), 781-788.

Chen, Y., Lee, W. S., Gan, H., Peres, N., Fraisse, C., Zhang, Y., & He, Y. (2019). Strawberry yield prediction based on a deep neural network using high-resolution aerial orthoimages. Remote Sensing, 11(13), 1584.

Cheng, F. C., Huang, S. C., & Ruan, S. J. (2011). Illumination-sensitive background modeling approach for accurate moving object detection. IEEE Transactions on broadcasting, 57(4), 794-801.

Cheng, H., Damerow, L., Sun, Y., & Blanke, M. (2017). Early yield prediction using image analysis of apple fruit and tree canopy features with neural networks. *Journal of Imaging*, *3*(1), 6.

Chinchuluun, R., & Lee, W. (2006). Machine vision-based citrus yield mapping system. In *Proceedings of the Florida State horticultural society* (Vol. 119, pp. 142-147). Florida State Horticultural Society.

Choi, J., Chang, H. J., Yoo, Y. J., & Choi, J. Y. (2012). Robust moving object detection against fast illumination change. Computer Vision and Image Understanding, 116(2), 179-193.

Cireşan, D., Giusti, A., Gambardella, L., & Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. Advances in neural information processing systems, 25.

Csillik, O., Cherbini, J., Johnson, R., Lyons, A., & Kelly, M. (2018). Identification of citrus trees from unmanned aerial vehicle imagery using convolutional neural networks. *Drones*, *2*(4), 39.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

Dias, P. A., Tabb, A., & Medeiros, H. (2018). Apple flower detection using deep convolutional networks. Computers in Industry, 99, 17-28.

Dwyer, B., Nelson, J. (2022). Solawetz, J., et. al. Roboflow (Version 1.0) [Software]. Available from **https://roboflow.com**. **computer vision**.

Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., & He, K. (2018). Detectron. In GitHub repository. GitHub. https://github.com/facebookresearch/detectron.

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., & Lewis, K. (2015). Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, *116*, 8-19.

Gongal, A., Karkee, M., & Amatya, S. (2018). Apple fruit size estimation using a 3D machine vision system. *Information Processing in Agriculture*, *5*(4), 498-503.

González-Araya, M. C., Soto-Silva, W. E., & Espejo, L. G. A. (2015). Harvest planning in apple orchards using an optimization model. In *Handbook of operations research in agriculture and the agri-food industry* (pp. 79-105). Springer, New York, NY.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., ... & Chen, T. (2018). Recent advances in convolutional neural networks. Pattern recognition, 77, 354-377.

Jiménez, A. R., Ceres, R., & Pons, J. L. (2000). A survey of computer vision methods for locating fruit on trees. Transactions of the ASAE, 43(6), 1911-1920.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and electronics in agriculture*, *147*, 70-90.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019). Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. *Precision Agriculture*, *20*(6), 1107-1135.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019). Deep learning–Method overview and review of use for fruit detection and yield estimation. Computers and electronics in agriculture, 162, 219-234.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

Martínez, J., Egea, G., Agüera, J., & Pérez-Ruiz, M. (2017). A cost-effective canopy temperature measurement system for precision agriculture: A case study on sugar beet. *Precision Agriculture*, *18*(1), 95-110.

Moltó, E., Pla, F., & Juste, F. (1992). Vision systems for the location of citrus fruit in a tree canopy. Journal of Agricultural Engineering Research, 52, 101-110.

PseudoLab Tutorial Team, Ahn, S., Kang, M., Kim, H., & Park, J. (2020, October). Object Detection, Detecting Medical Masks, 5. Faster R-CNN (Kim, H., & Kim, L., Trans.). https://pseudo-lab.github.io/Tutorial-Book-en/chapters/en/object-detection/Ch5-Faster-R-CNN.html

Qu, Y., Ou, Y., & Xiong, R. (2019, December). Low illumination enhancement for object detection in self-driving. In 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 1738-1743). IEEE.

Rahnemoonfar, M., & Sheppard, C. (2017). Deep count: fruit counting based on deep simulated learning. *Sensors*, *17*(4), 905.

Ranjan Rath, S. (2021, October 25). Custom Object Detection using PyTorch Faster RCNN. Retrieved from: https://debuggercafe.com/custom-object-detection-using-pytorch-faster-rcnn/

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). py-faster-rcnn. In GitHub repository. GitHub. https://github.com/rbgirshick/py-faster-rcnn.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *sensors*, *16*(8), 1222.

Sabzi, S., Abbaspour-Gilandeh, Y., García-Mateos, G., Ruiz-Canales, A., & Molina-Martínez, J. M. (2018). Segmentation of apples in aerial images under sixteen different lighting conditions using color and texture for optimal irrigation. Water, 10(11), 1634.

Santos, T. T., de Souza, L. L., dos Santos, A. A., & Avila, S. (2020). Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association. Computers and Electronics in Agriculture, 170, 105247.

SZ DJI Technology Co., Ltd., Shenzhen, China. Matrice 210 RTK V2. Retrieved October 25, 2022, from https://www.dji.com/nl/matrice-200-series/info

SZ DJI Technology Co., Ltd., Shenzhen, China. DJI Phantom 3 Pro. Retrieved October 25, 2022, from https://www.dji.com/nl/phantom-3-pro/info

SZ DJI Technology Co., Ltd., Shenzhen, China. DJI Phantom 4 Pro. Retrieved October 25, 2022, from https://www.dji.com/nl/phantom-4-pro/info

SZ DJI Technology Co., Ltd., Shenzhen, China. Zenmuse X7. Retrieved October 25, 2022, from https://www.dji.com/nl/zenmuse-x7

Tan, L., Huangfu, T., Wu, L., & Chen, W. (2021). Comparison of YOLO v3, faster R-CNN, and SSD for real-time pill identification.

Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E., & Liang, Z. (2019). Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Computers and electronics in agriculture*, *157*, 417-426.

Tkachenko, Maxim, Malyuk, Mikhail, Holmanyuk, Andrey, & Liubimov, Nikolai. (2020). Label Studio: Data labeling software. Retrieved from https://github.com/heartexlabs/label-studio

Tzutalin (2015). LabelImg. Git code, Available at: https://github.com/tzutalin/labelImg.

Wang, Q., Nuske, S., Bergerman, M., & Singh, S. (2013). Automated crop yield estimation for apple orchards. In *Experimental robotics* (pp. 745-758). Springer, Heidelberg.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2. In GitHub repository. GitHub. https://github.com/facebookresearch/detectron2.

Xiao, Y., Jiang, A., Ye, J., & Wang, M. W. (2020). Making of night vision: Object detection under low-illumination. IEEE Access, 8, 123075-123086.

Zhou, R., Damerow, L., Sun, Y., & Blanke, M. M. (2012). Using colour features of cv.'Gala'apple fruits in an orchard in image processing to predict yield. *Precision Agriculture*, *13*(5), 568-580.

Zhu, N., Liu, X., Liu, Z., Hu, K., Wang, Y., Tan, J., ... & Guo, Y. (2018). Deep learning for smart agriculture: Concepts, tools, applications, and opportunities. *International Journal of Agricultural and Biological Engineering*, *11*(4), 32-44.

Ziliani, M. G., Parkes, S. D., Hoteit, I., & McCabe, M. F. (2018). Intra-season crop height variability at commercial farm scales using a fixed-wing UAV. *Remote Sensing*, *10*(12), 2007.

Appendices

Appendix A

Full images of the images used in

Figure A1. Full image of 2018. Cropped version used in figure 2A

Figure A2. Full image of 2019. Cropped version used in figure 2B

Figure A3. Full image of 2020. Cropped version used in figure 2C

Figure A4. Full image of 2021. Cropped version used in figure 2D

Appendix B

Statistical performance metrics for all four models tested on all four test datasets.

Table B1. Overview of the calculated statistical parameters. The optimal model during each of the four training sessions was tested on the four separate sets of test data. During this report only Average Precision (AP) was taken into account.

Test Dataset	Metric	Base Model	Reflection Model	Shadows Model	Combined Model
Base	Precision	0.6401	0.6738	0.6816	0.7022
	Recall	0.4993	0.4785	0.4729	0.4849
	F1-score	0.5610	0.5596	0.5584	0.5737
	AP	0.4454	0.4165	0.4183	0.4326
Brightness	Precision	0.6017	0.6509	0.6424	0.6739
	Recall	0.4329	0.4698	0.4266	0.4675
	F1-score	0.5035	0.5457	0.5127	0.5521
	AP	0.3730	0.4034	0.3708	0.4100
Shadows	Precision	0.4144	0.5530	0.6114	0.6368
	Recall	0.2807	0.3241	0.3647	0.3792
	F1-score	0.3347	0.4086	0.4569	0.4753
	AP	0.2139	0.2556	0.3120	0.3305
Combined	Precision	0.4937	0.5711	0.5721	0.6297
	Recall	0.3316	0.3669	0.3720	0.3950
	F1-score	0.3968	0.4468	0.4509	0.4855
	AP	0.2710	0.3001	0.2723	0.3453