OULUN AMMATTIKORKEAKOULU

Mikko Pajula

BERRY DENSITY ESTIMATION WITH DEEP LEARNING

Estimating Density of Bilberry and Lingonberry Harvest with Object Detection

BERRY DENSITY ESTIMATION WITH DEEP LEARNING

Estimating Density of Bilberry and Lingonberry Harvest with Object Detection

Mikko Pajula
Master’s thesis
Autumn 2022
Data Analytics and Project
Management
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences
Data Analytics and Project Management

Author: Mikko Pajula
Title of the thesis: Berry Density Estimation With Deep Learning – Estimating Density of Bilberry and Lingonberry Harvest with Object Detection
Thesis examiner: Ilpo Virtanen
Term and year of thesis completion: Autumn 2022 Pages: 82

Currently, yield estimation of berry harvest is done manually by calculating berries in a predefined area. Manual counting is labour intensive and thus, the data created lacks scope. This thesis studied the method to estimate berry harvest as berry density value only with a consumer-level mobile phone camera. In addition, the thesis studied if berries detected from a top vegetation image could correlate with the berry density in the area defined by field measurement.

The density estimation process was twofold. Part one was to study the possibility of detecting lingonberry and bilberry from an image in the growth phases: flower, raw and ripe. YOLOv5 object detection was applied for this process. The original image was split into smaller images for deep learning to avert losing pixel information by scaling. Two split methods were tested.

Part two was to define the area for the image. Multiple methods were compared to define the most promising approach to estimating the area for berry density estimation. Field measurement values were used to describe the baseline area and to study the feasibility of determining berry density only from visible berries in the image.

The complete practical process of training the YOLOv5 object detection model was described in the thesis, from annotating datasets through editing and balancing datasets to training. Inadequacies in the dataset were described, and future improvements were suggested.

Keywords: deep learning, lingonberry, bilberry, harvest, neural networks, annotation

CONTENTS

1 INTRODUCTION

Wild berries are essential non-wood forest products for recreational use and other emerging ecosystem usages (Kilpeläinen 2016). Accurate and comprehensive berry harvest statistics benefit the Natural Resources Institute Finland (LUKE) and the Norwegian Institute of Bioeconomy Research (NiBio). Object detection with the machine learning can potentially alleviate harvest inventory currently done by hand.

Berry Machine project aimed to study machine learning to evaluate berry harvest using a consumer-level mobile phone camera. The financier was Interreg Nord fund, and the participating institutes were the Natural Resources Institute Finland (LUKE) and the Norwegian Institute of Bioeconomy Research (Nibio). Frostbit Software Lab at Lapland University of Applied Sciences did the actual implementation of machine learning.

The project aimed to study the possibility of doing berry yield estimation using a mobile phone camera with machine learning object detection in the natural environment. Computer vision combined with a consumer-level mobile phone camera raises the possibility of crowdsourcing and simplifying data collection and promises to bring savings with more consistent data to participating institutes. Evaluation was planned to be done by detecting berries in images with computer vision and estimating berry density with statistical or other means. The thesis focused on the technical process of using object detection to classify and detect berries from images. The number of detected berries was used to study the possibility of estimating berry yield density by estimating area visible in image based on detected berries and rectifying the resulting berry count divided by area (berry density) with linear regression.

The harvest state was observed by counting berries in different stages of development. Currently, LUKE collects data manually for berry yield estimation by counting berries in predetermined places (Kilpeläinen 2016). Counting is done by marking one m^2. Berries are then counted one berry at a time inside a square. One square berry count represents the number of berries per one m^2. Multiple squares compensate for deviation in square meter count values (marjahavainnot.fi 2022). Data collection with field measurement is both laborious and expensive and simultaneously not available for more elaborated studies (Bohlin 2021). Field measurement data is nevertheless valuable information for current, and future studies like this thesis, as results are created with and tested against field measurement data.

Berries were counted in three stages of the growth phase: Flower, raw, and ripe. The study was limited to two berry species: Lingonberry (Vaccinium Vitis-idea) and bilberry (Vaccinium Myrtillus). Bilberry and lingonberry are common berries harvested by households and companies and are abundant in evergreen conifer forests. Both berries act as a significant food source for local fauna. Intensive forest management can harm berry yields. (Kilpeläinen 2016). Bilberry yield is best in mature stands with medium or poor fertility, while lingonberry yield is highest in low tree density environments with poor mineral soil (Ihalainen 2002).

2 THEORETICAL FRAMEWORK

The main techniques were explained in this chapter. The neural network in this thesis was used to detect objects from an image. For more context, Convolutional Neural Network, the technology on which object detection is based, was explained in this chapter, along with different methods of classifying images. Finally, the YOLO Object detection technique used to train detection models was explained briefly in this chapter, along with the other methods used to derive berry density estimation results.

2.1 Convolutional neural network

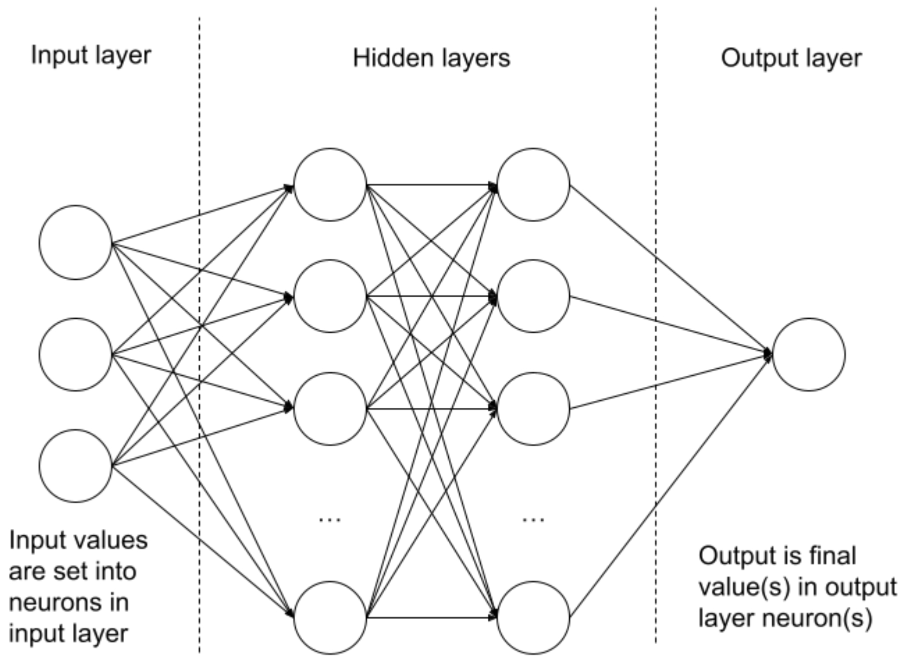

Input layer
Hidden layers
Output layer
...
...
Input values are set into neurons in input layer
Output is final value(s) in output layer neuron(s)

Figure 1. Artificial Neural Network.

An Artificial Neural Network (ANN) is constructed of an input layer, hidden layers and an output layer shown in Figure 1. The processed data is placed as values in input neurons (Gogul 2017). The weight and bias values define the neuron values in the following hidden layers, and as a fully connected network, each neuron value is impacted by all neurons in the previous layer.

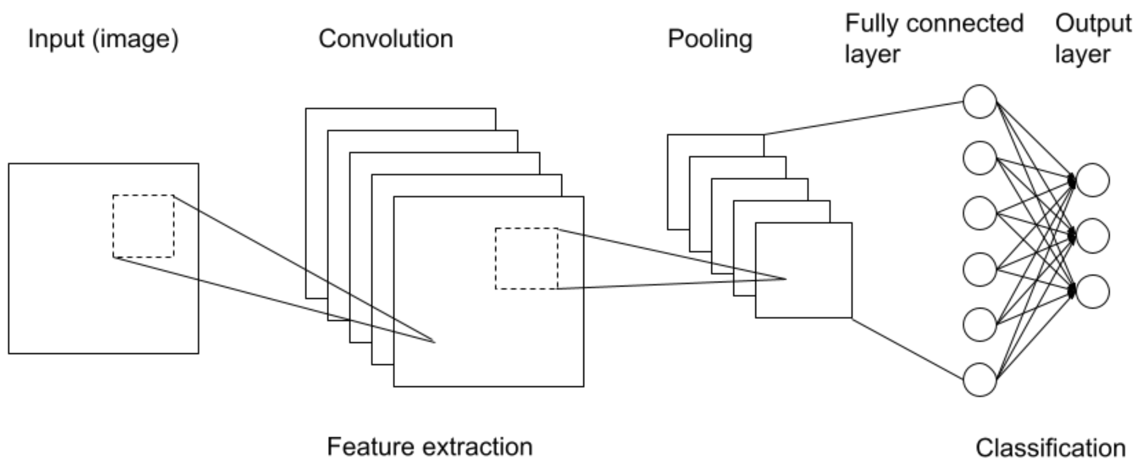


Figure 2. Convolutional Neural Network.

Convolutional Neural Network (CNN) is a breakthrough deep learning technology (Marmanis 2016) and an improvement over ANN that outperformed competing methods on image classification tasks in 2012 (Marmanis 2016). CNN architecture applies local receptive fields as convolution layers that can extract edges, endpoints, and corners (Figure 2). The architecture allows the detection of high-order features from input data. Data sub-sampling or pooling is also used in CNN architecture (Figure 2) to reduce potentially harmful irrelevant precision after convolution and output sensitivity caused by positional shifts and distortions. (Lecun 1998)

Deep neural network architectures like CNN have the potential to achieve significant detection results using purely supervised learning with challenging datasets (Krizhevsky 2012). CNN is capable of human-level or higher accuracy (Galanty 2021).

2.2 Image classification

The image classification process categorises images by labelling images. Image classification is essential for computer vision research (Bayraktar 2020). Convolutional Neural Networks (CNN) succeed in image classification and are the only method used in this thesis context to classify images. Image classification with CNN produces a label for a given input image from a predetermined label list.

2.3 Object detection

A well-written description of object detection is found in the article "Deep Learning for Generic Object Detection: A Survey" by Liu et al. "Object detection, one of the most fundamental and challenging problems in computer vision, seeks to locate object instances from many predefined categories in natural images" (Liu 2018).

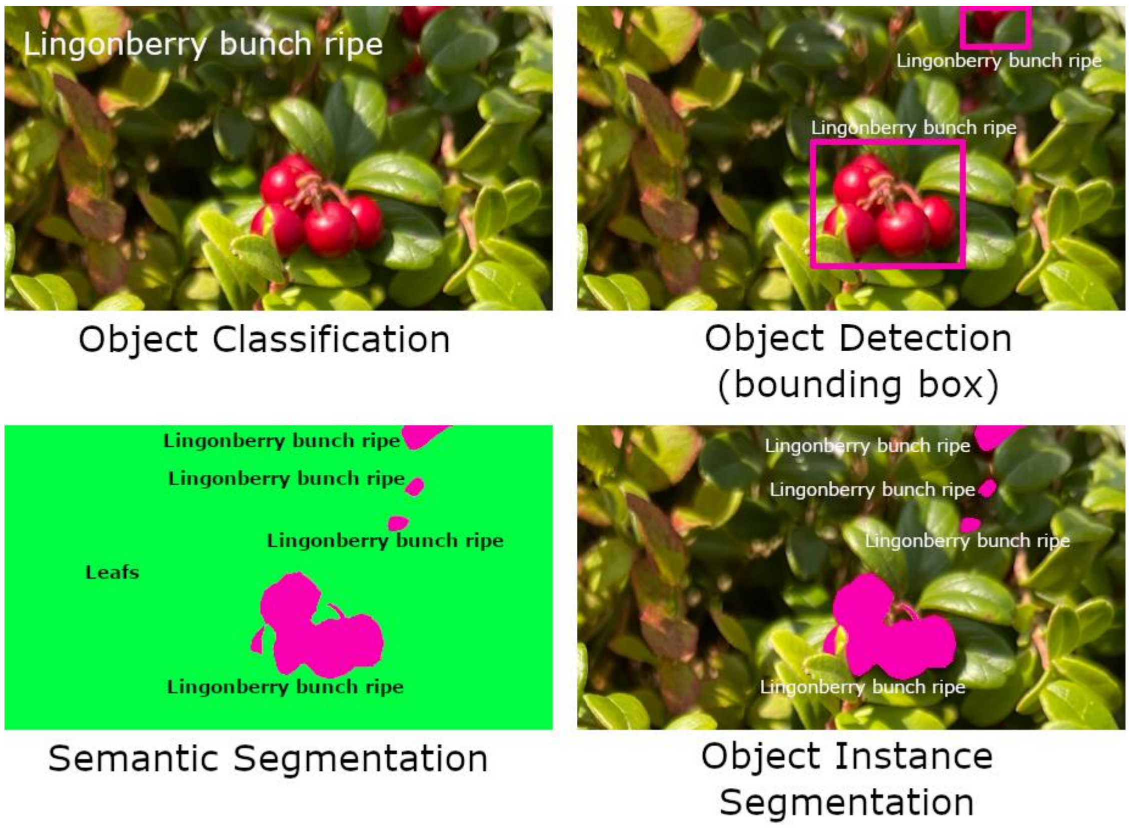


Figure 3. Ways to classify or segment images in computer vision.

Multiple ways to spatially detect object location in an image exist, as Figure 3 shows. Object detection with object bounds marked with a bounding box is one detection technique (Liu 2018). Object detection with bounding boxes is used in this thesis context. Object detection returns the object pixel coordinate position in the input image and object classification label (Bayraktar 2020).

2.4 YOLO object detection

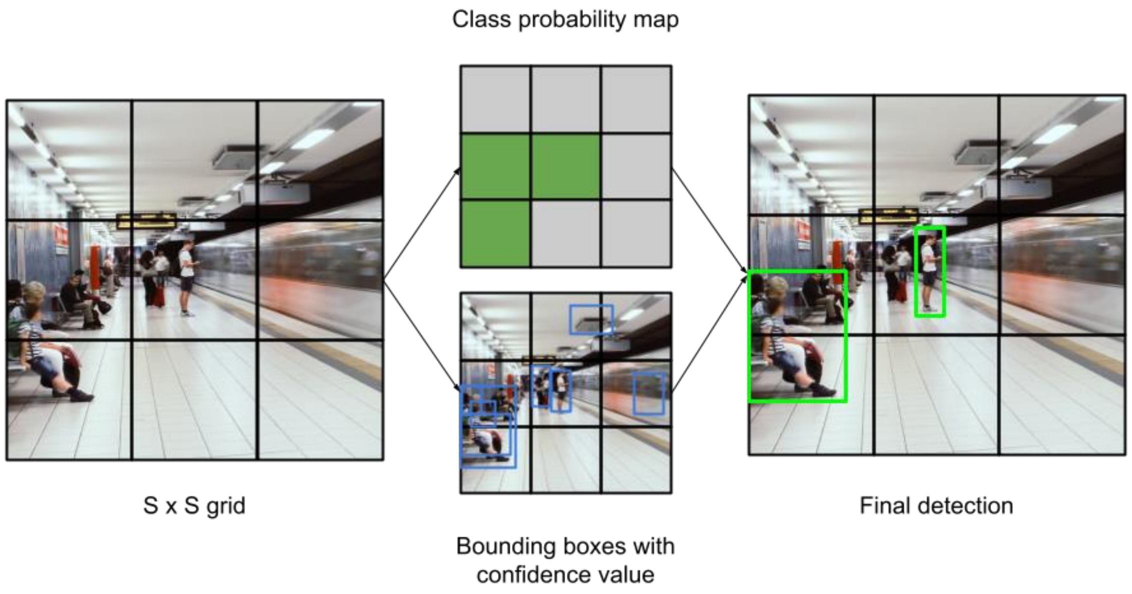


Figure 4. YOLO (You Only Look Once) object detection visualization.

Before YOLO (You Only Look Once), object detection from images was done with techniques like the sliding window approach that ran image classification multiple times in part of the image so that the detection area moved an equal amount each time. Other methods like R-CNN (Regions with CNN) existed but had complex optimising pipelines (Redmon 2015). YOLO object detection uses a single neural network for detection by dividing an image into a grid (Figure 4), and each grid predicts the object and its bounding box. Finally, multiple bounding boxes are joined with classification probabilities for each grid cell (Redmon 2015). YOLO Object detection was designed to be simple, fast and accurate (Redmon 2015). Originally YOLO was not effectively capable of detecting small objects 2021 (Ni 2021). However, Recent improvements in newer YOLO object detectors improve YOLO small object detection capabilities while retaining the YOLO signature speed property (Benjumea 2021).

Berry detection is done using the YOLOv5 Object detector. YOLOv5 promises to be faster and more accurate than previous YOLO iterations (Solawetz 2020). A critical aspect of the YOLOv5 detector is that it is built on the PyTorch framework (Solawetz 2020). PyTorch makes YOLOv5 deployment easier in some platforms like IT Center for Science (CSC) supercomputer with preinstalled PyTorch and limited user privileges. YOLOv5 also offers embedded data augmentation to data processing, like rotation and translation of images (Dongjian 2021).

2.5 Linear regression

"Linear regression is a fundamental building block of statistical data analysis. It amounts to estimating the parameters of a linear model that maps input features to corresponding outputs" (Gast 2019). With simple linear regression, observation data of two variables can be used to create a model for predicting the second value based on other values. The linear regression model can be evaluated with the Pearson correlation coefficient. (Hackeling 2017)

2.6 Intersection over union (IoU)

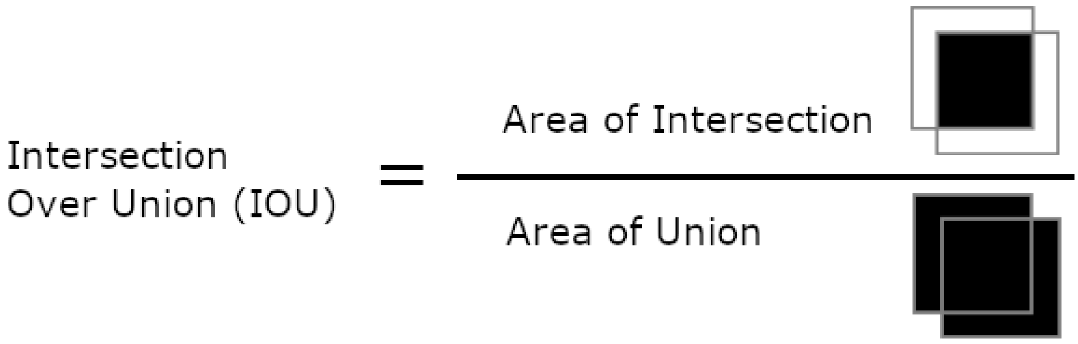


Figure 5. Visually explained formula to calculate Intersection Over Union.

Intersection over union (IoU) is a standard method to benchmark object detection labels with rectangle bounding boxes (Rezatofighi 2019). IoU is calculated as a relation between two areas by dividing the intersection area by the union area (Figure 5).

$$IoU = \frac{A \cap B}{A \cup B}$$

Equation 1. Calculating the intersection of the union

IoU is calculated with Equation 1 (Rezatofighi 2019). With complete overlap, when an area of union and the area of intersection are the same, Intersection over Union (IoU) is one. When the object position shifts to a different position, the IoU value drops closer to zero. Typically, a 0.5 threshold is used as the limit to determine the prediction as a correct result (Zitnick C Lawrence. Dollár 2014).

2.7 Homography

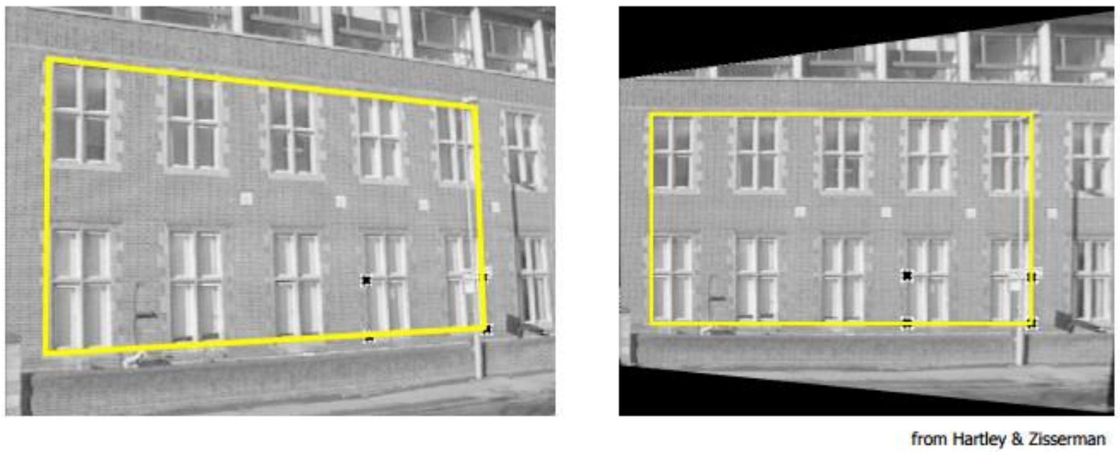


Figure 6. Homography perspective correction (OpenCV 2022)

2D Homography is a projective transformation and a fundamental task in computer vision (DeTone 2016). Performing homography projective transformation requires at least four matching points in the original image and resulting coordinates (Malis 2007). These points form a homography 3X3 transformation matrix (Malis 2007). The homography matrix is used to transform, as visualised in Figure 6. Use cases, for example, are perspective removal or image correction (OpenCV 2022).

3 DATA COLLECTION

During the summer and autumn of 2021, three groups took berry pictures in the plants natural habitat. The pictures were collected during the summer and autumn to gather material from different growth phases (flower, raw and ripe). Multiple locations were used to give variation. Also, bilberry and lingonberry grow differently in different environments (Ihalainen 2002). Multiple consumer-level mobile phones and a single digital SLR (single-lens reflex camera) were used to take pictures.

Groups were instructed to count berries using the current method of manual counting in a square meter. Pictures were taken from multiple distances. The first distance was to include counting squares in the picture. The second distance was closer to the ground to include multiple berries in the image. Close-up images of berries were also taken. Pictures were taken from eight central locations around the Rovaniemi region. Images were taken six times on every two berries in three growth phases. One separate location image set was taken afterwards later in autumn to get a ripe lingonberry dataset where these red berries were against a reddish autumn background.

Two groups took approximately 10000 pictures in multiple locations. These groups installed and counted a large number of squares and collected images. Six phones were used, belonging to three models and one SLR. One group used three phone models in only one location, taking only bilberry images. Additional devices were used to add a variation to the device spectrum.

Device type	**Number of devices**	**Group**
Apple iPhone 6s	1	3
Apple iPhone 11	1	3
Apple iPhone 12 pro max	2	1 and 2
Nokia 7 plus	2	1 and 2
Nokia 3.4	1	3
Samsung a40	2	1 and 2
SLR	1	1 and 2
Total	**10**	

Table 1. Devices used in data collection

Two groups took most of the images and used similar phone sets. These two groups comprised mostly agrologist students (groups 1 and 2) with iPhone 12 pro max, Nokia 7 plus, Samsung a40 and SLR. The complementary group used iPhone 6s, Nokia 3.4 and iPhone 11 (group 3). Data collected by the complementary group has the limitation that three models were only used in one location and only took bilberry images. A complementary group, formed chiefly of information technology students, was created to increase the number of devices used, create quick access data, and acquaint developers with the current manual harvest estimation technique.

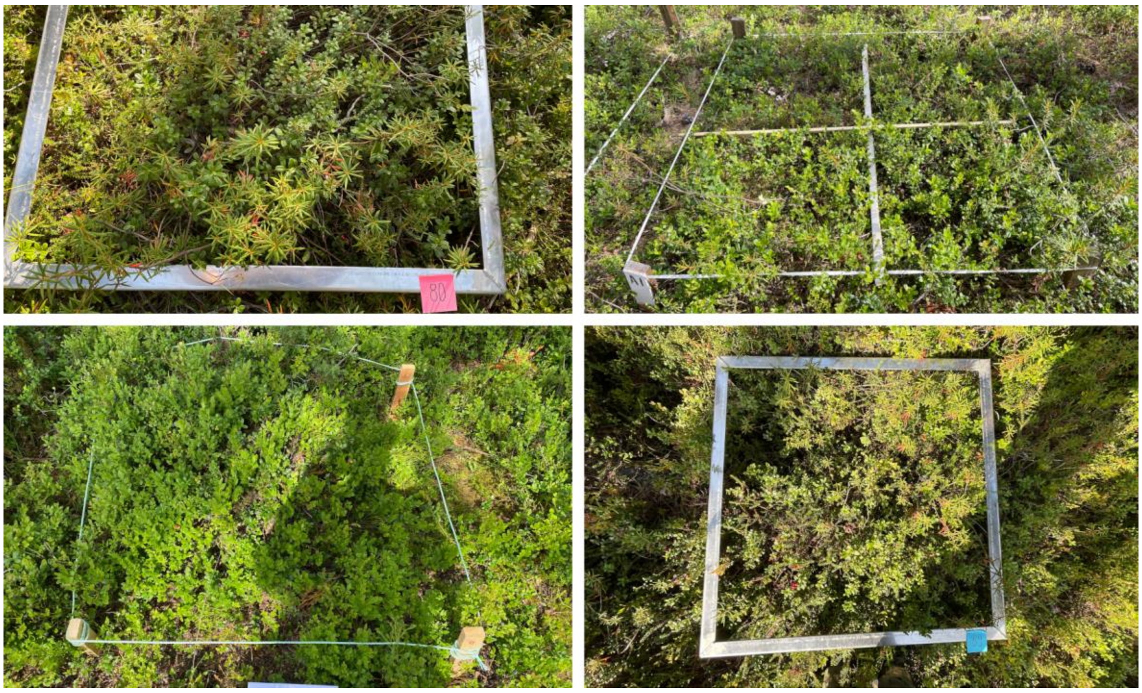


Figure 7. Counting frames.

One square meter counting squares were custom-made aluminium frames or wooden poles in each corner, and sides formed with wires running from one pole to another (Figure 7). Manual calculation results were written on paper, and this paper was arranged next to the frame so that the count value would be visible in the image taken from the frame. Typically, images following full frame images were taken closer and inside the frame, but this was by no means certain. An extensive set of other berries images was also taken from around the frames from multiple distances. Groups were instructed to take a varying range of berry images for the training dataset

Images were stored, so the device and berry phase target area or location was visible in folder naming. For example, "Tervola_26_7/iphone12/IMG_1107.JPG". 26.7.2021 in the Tervola region target berry phase was ripe lingonberry. Thus, all images had the following meta information: Location, target berry species and phase and device.

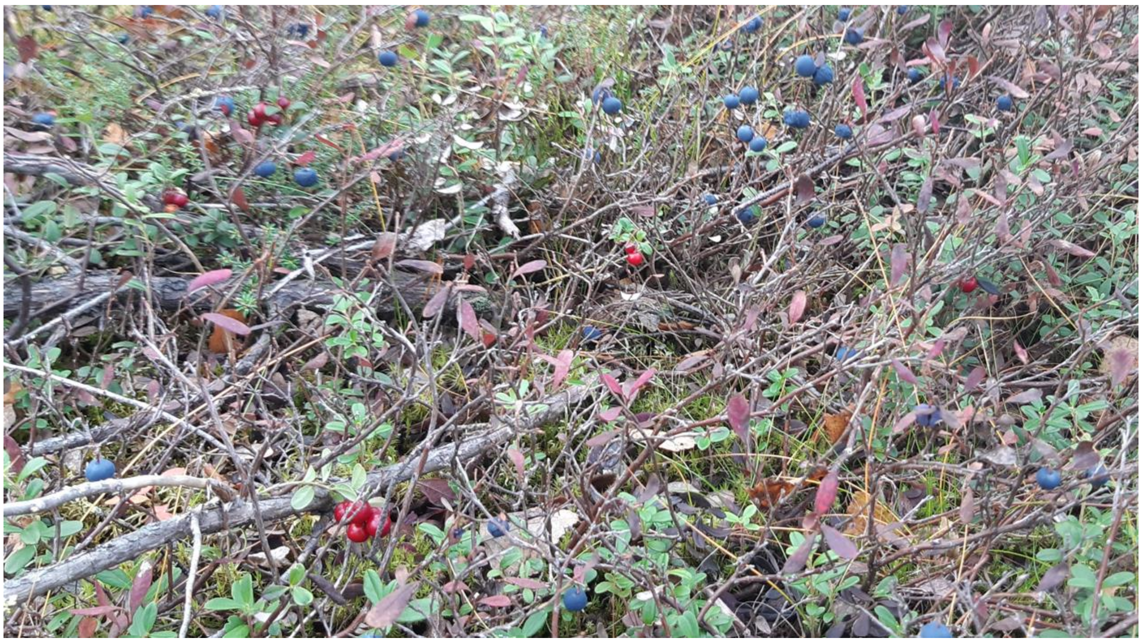


Figure 8. Example of the autumn dataset from Kivitaipale.

After the planned field measurement data collection was finished, an additional dataset with a more colourful background was collected. The assumption was that deep learning would detect ripe berries primarily based on colour, as red and blue colours are rare in the collected dataset background. However, some rare colours from measurement group material were visible in the image dataset. Autumn colours were a concern. How would deep learning separate a red leaf from a red berry? Therefore a separate autumn dataset of 168 images was created and annotated fully. Dataset was collected from Kivitaipale, Rovaniemi, using Samsung a40 and different SLR (Canon 1100D). Different SLRs were used to add variation in devices, but the original SLR was unavailable in autumn. The example image from the autumn dataset is visible in Figure 8.

	Bilberry			**Lingonberry**		
Location	**Flower**	**Raw**	**Ripe**	**Flower**	**Raw**	**Ripe**
Pöyliövaara	X			X		
Hirvas	X	X	X			
Juotasniemi	X	X	X			
Käyrämö	X	X				
Levi		X	X			
Pisa				X		
Tervola	X	X	X	X	X	
Ounasvaara	X	X	X			

Unknown		X
Kivitaipale		X

Table 2. Image dataset locations

The image dataset was collected from several locations (Table 2). Ounasvaara images were collected by Group 3. Kivitaipale images are an additional autumn dataset.

3.1 Annotation

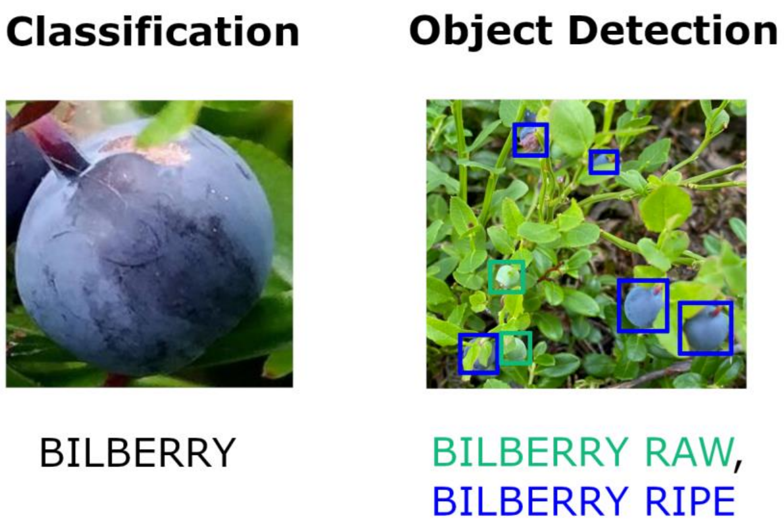


Figure 9. The difference in classification and object detection.

High-quality and large-scale datasets are essential for training CNN-based object detection methods (Z.-Z. X. Wu 2022). Therefore, the aim was a fully supervised instance-level annotated dataset despite being labour-intensive and time-consuming (Z.-Z. X. Wu 2022). Instance level is referred to object detection differentiating from image classification. Image classification is referred to the process of labelling whole images. Object detection is the process of labelling instances in images (Russakovsky 2015). This difference is shown in Figure 9.

Predesigned object detection neural network scales images to "native" resolution, which depends on the selected predesign. For example, the first step of the YOLO object detection framework's native resolution is 416x416 (Borel 2019). Yolov5 is available in two resolutions: 640x640 and 1280x1280 (Jocher 2022). In the collected dataset, mobile phone and SLR original image sizes are, for example, 4032x3024 pixels (Nokia 7 plus). Scaling a single image to object detection resolution would cause severe data loss. Especially berries are small objects to detect. Our challenge was to decide on data rescaling or/and slicing images for annotation before choosing object detection technology. No scaling or slicing was done before and during annotation to avoid possible issues with wrong image size or data loss. Instead, annotation was done directly to the original image sizes.

Annotation was done via premade open-source annotation tool – Label studio (Label Studio 2021). Label Studio is a web application with premade backend and frontend. The application was deployed on our server and was made available in the local area network. Label Studio, as a web tool, allowed annotation to be done efficiently to the same dataset by multiple persons without overlapping work.

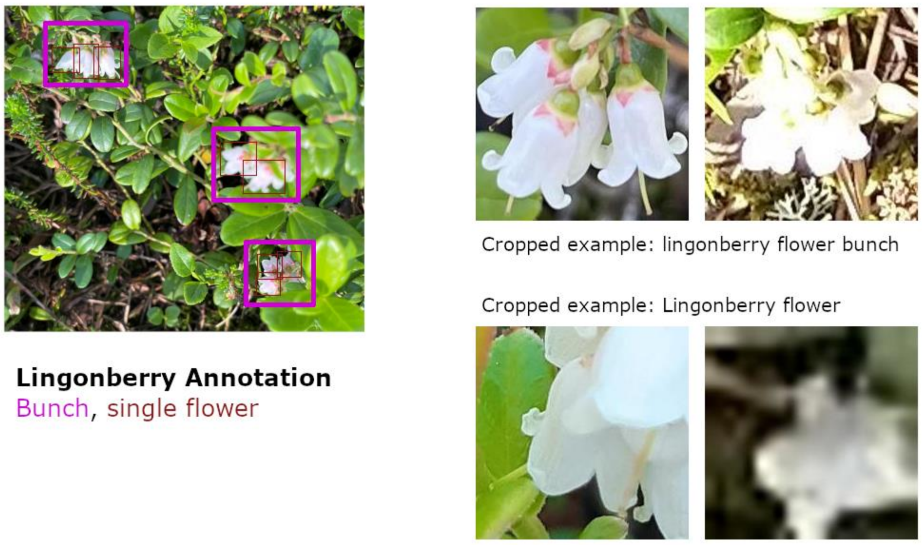

Lingonberry Annotation
Bunch, single flower
Cropped example: lingonberry flower bunch
Cropped example: Lingonberry flower

Figure 10. Examples of annotations of single lingonberry flower and lingonberry flower bunch.

The annotation process was done using rectangle labels. The main goal is to differentiate bilberry flowers, raw bilberries, ripe bilberries, lingonberry flowers, raw lingonberries, and ripe lingonberries. These berries and their growth phases were the first annotation labels. Three additional labels were added: Lingonberry bunch of flowers, ripe and raw. These were added due to the difficulty annotating single lingonberry phases from each other in a bunch (Figure 10). Single lingonberry phase annotation labels were kept alongside bunch labels. Both were annotated to enable as many possible ways to use annotation data in future.

Phase	**Species**	**label**
Flower	Bilberry	bilberry flower

Raw	Bilberry	bilberry raw
Ripe	Bilberry	bilberry ripe
Flower	Lingonberry	lingonberry flower
Raw	Lingonberry	lingonberry raw
Ripe	Lingonberry	lingonberry ripe
Flower	Lingonberry	lingonberry bunch flower
Raw	Lingonberry	lingonberry bunch raw
Ripe	Lingonberry	lingonberry bunch ripe

Table 3. Dataset labels in annotation

The primary dataset was created with nine labels (Table 3), including the single lingonberry and bunch sets. Annotation was initiated with six labels. Therefore, some previously annotated images needed to be relabelled by the final annotation directions.

Three thousand eighty-one (3081) images from approximately 10000 images dataset were selected for the annotation process. No more resources were available to annotate more images. The image selection process was mainly affected by timetables. Group 3 bilberry dataset was always annotated as this dataset was the first to be available. Also, the group 3 dataset was smaller and more critical as it was created with different devices, thus diversifying the final dataset. As soon as groups 1 and 2 datasets were available, flowers and raw lingonberries were added to the annotation process. Ripe lingonberries and ripe bilberries were available later as both berry phases of the data collection came available later in the summer. The initial dataset in annotation was supplemented with the berry phase with the least number of annotations. Raw and ripe lingonberries were added later in autumn and annotated from autumn to the start of winter 2021.

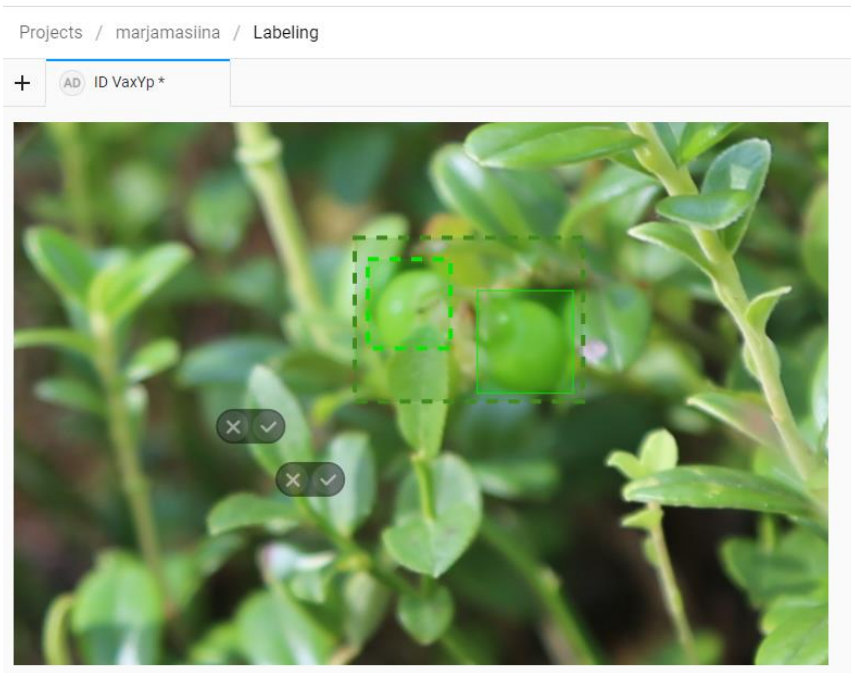


Figure 11. Machine learning pre-annotation results in Label Studio.

Label Studio application also supports machine learning pre-annotation (Label Studio 2022). Machine learning pre-annotation was used to help annotation after the first preliminary object detection was trained. Pre-annotation was used more with lingonberry raw -additional annotation. Automatic pre-annotation in practice received mixed reactions from personnel annotating whether pre-annotation was wanted depending on the berry type and the person in question. Pre-annotation made locating some berry types, like raw ones, easier, but several false positives in pre-annotation diminished pre-annotation value, especially with easier-to-spot berry phases like ripe ones. Label Studio's pre-annotation user interface is shown in Figure 11.

3.2 Primary annotated dataset

The primary annotated dataset was 61269 annotated labels consisting of 2855 annotated images. The number is smaller than the original annotated dataset size (3081). However, 226 images had no annotations. Images without annotations were due to two reasons: One is that the image did not have any visible berries, or the image was unusable in quality and thus had nothing to annotate, for example, due to camera shake.

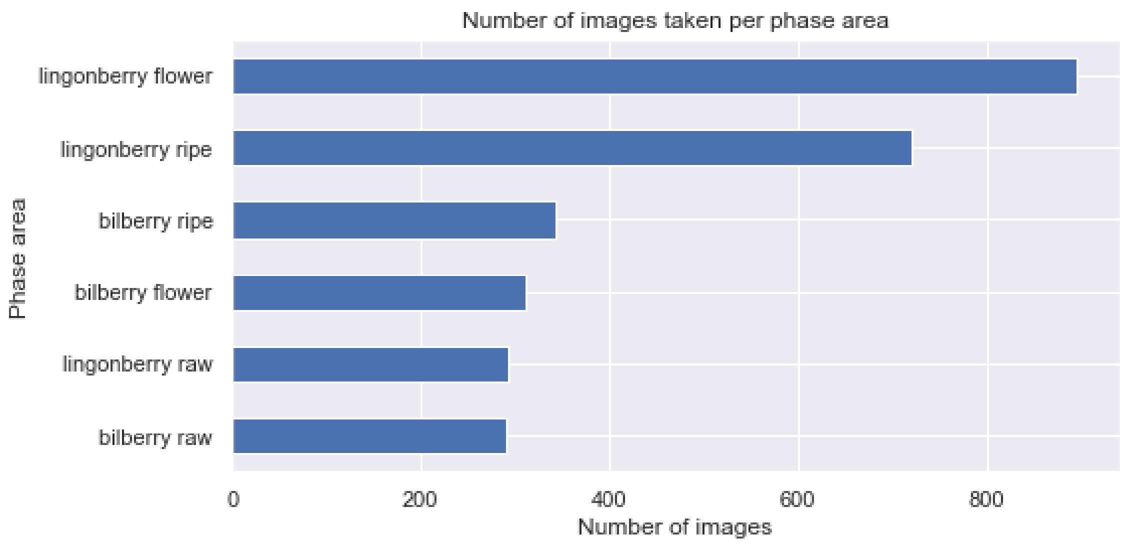


Figure 12. Number of images taken per phase area in the annotated dataset

The annotated dataset was unbalanced, considering images taken per phase area (Figure 12). The phase area is a predefined target phase area where the goal is to take specific berry species and phase images. Although the goal was to take, for example, ripe lingonberry images, some images taken in that area may contain more other annotations. The target berry phase in an image is defined as the phase area. Target may be a single berry phase, but results may include other phases. Phase area does not mean dataset annotated labels will end up similarly unbalanced as berry density may vary in images taken from different areas.

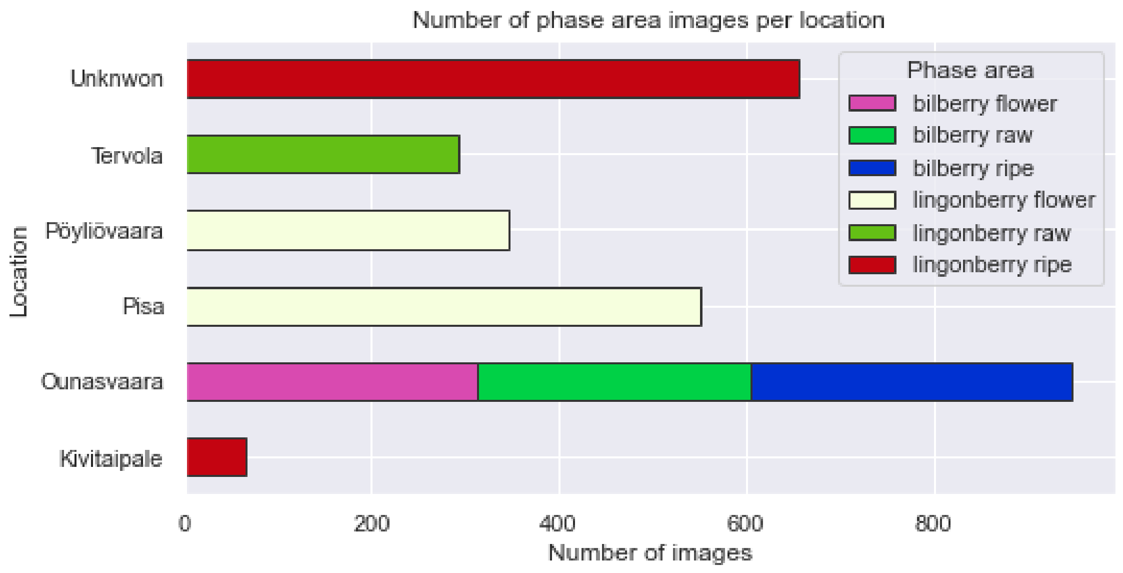


Figure 13. The number of phase area images from different locations.

Observing annotated dataset per location (Figure 13) reveals that location diversity was ignored while balancing species phase numbers. Presumably, having species phase images from multiple locations would create a more comprehensive dataset. The location itself was not the primary criterion for data diversity. Still, different locations can produce more varied backgrounds due to different weather conditions, soil types and field personnel actions.

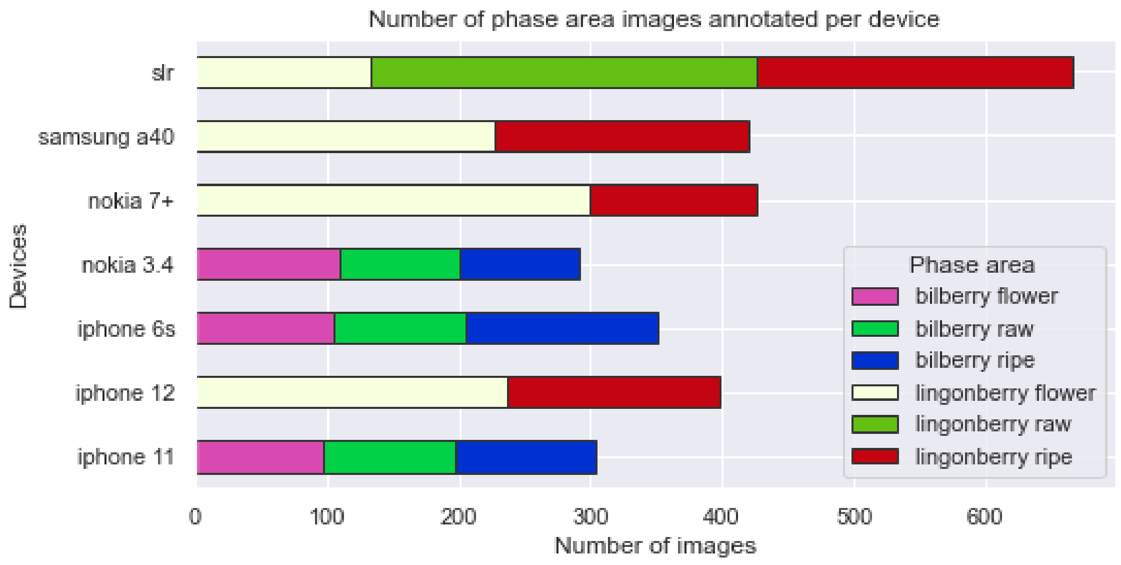


Figure 14. Number of phase area images per device in the annotated dataset

Images selected for annotation per device are numbering between 300 to 400, except for the SLR camera, which had 665 images taken (Figure 14). A possible significant issue with the dataset collecting devices is that some devices take only certain types of berries, as shown in Figure 14. The dataset structure is two-fold as it contains results from two different types of field measurement groups. Group 1 and 2 with devices SLR, Samsung a40 and Nokia 7+. Group three with devices Nokia 3.4, iPhone 6s and iPhone 11. The annotated dataset was supplemented in autumn by annotating missing lingonberry raw and ripe images. Raw lingonberry images were only from SLR data (Figure 14).

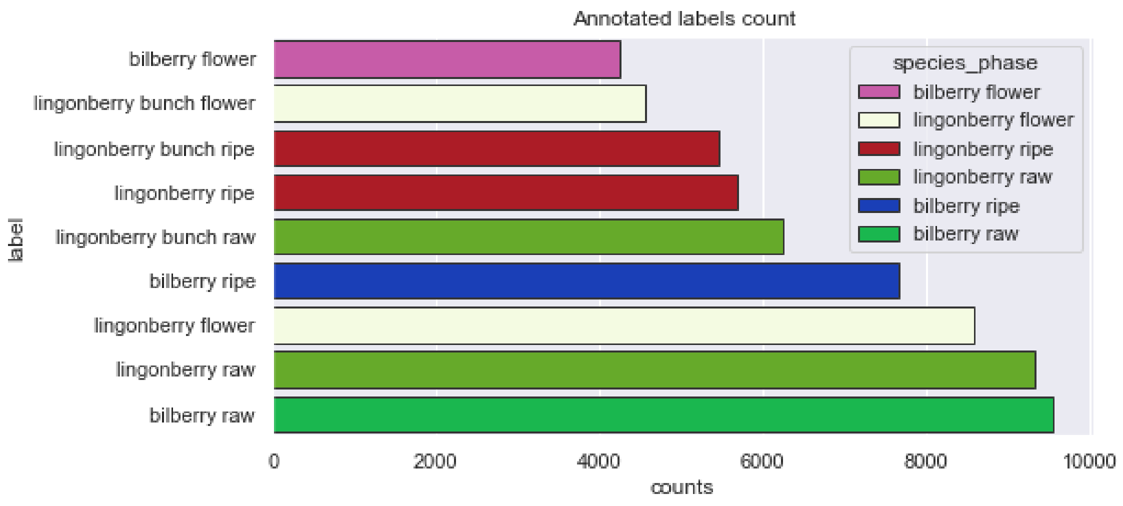


Figure 15. Annotated labels count

Phase area (Figure 13) was not the main factor when balancing the annotation dataset. The main factor was the annotation label count (Figure 15). After the first annotation status checks, the dataset was quite uneven. More images were added to balance the dataset—mostly raw lingonberries. After balancing, the number of raw lingonberries increased more than initially intended. The increase is visible in Figure 15. The inability to accurate balance datasets was due to how the annotation process was organised. With group 3, the bilberry dataset was entirely annotated as it was first available, smaller in image count and added three new devices for variety. Group 1 and 2 datasets also contained many bilberry images, but the immediate target was to add lingonberry images to the dataset to cover all berry species and phases fully.

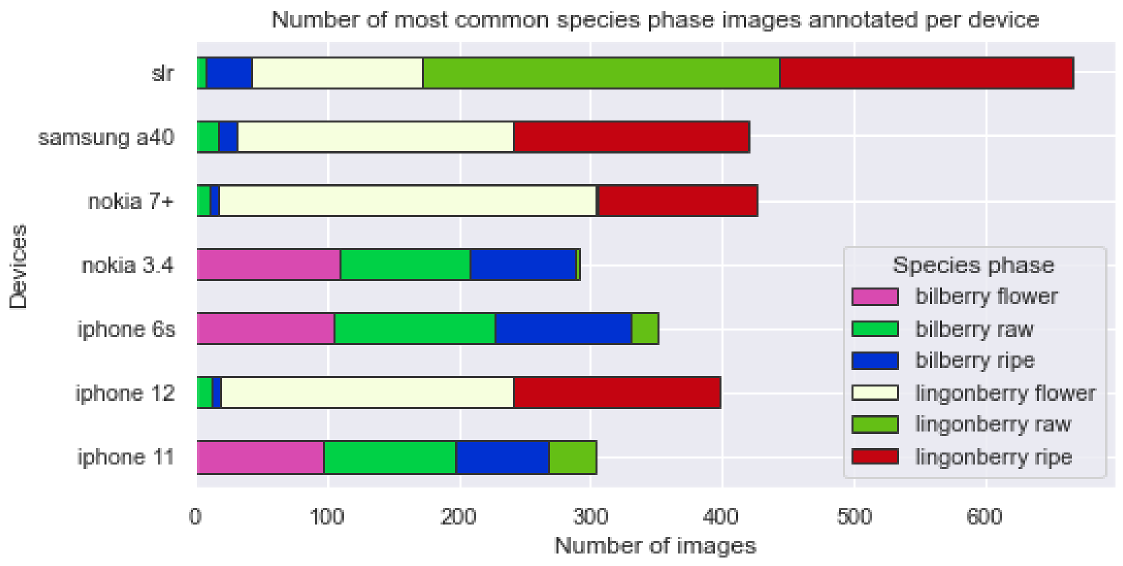


Figure 16. The number of most common species phase images per device in the annotated dataset.

The annotated dataset was more varied in label-device relation (Figure 16) than the image phase area suggests (Figure 14). The main berry species phase was formed by counting all annotations in an image and selecting the most common species phase as the primary berry type. Figure 16 shows, in comparison with Figure 14, that the phase area contains a variety of other berry species and phases. The difference between phase area and the most common species phase annotated is 9%. Images produced in the phase area mainly contained expected berry species phases but not always. Different labels in images positively add variety to the annotated dataset.

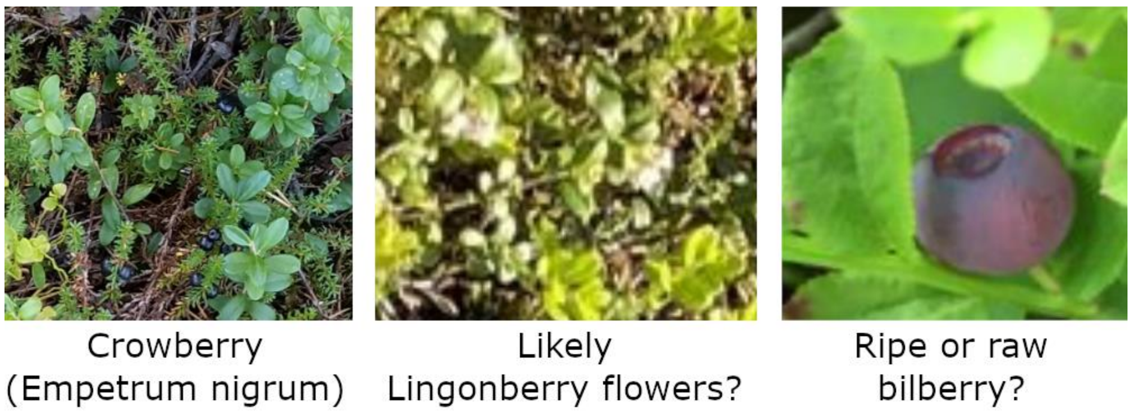


Figure 17. Examples of annotation difficulties: Berry species excluded from annotation object in the edge of visibility and berry phase borderline case.

Bog bilberry (Vaccinium Uliginosum) and Crowberry (Empetrum Ingrum) were present on annotated images, and it is possible that in 60 000 annotations are, an unknown percentage of labels added as the wrong species. Crowberries are a common sight but easily distinguished due to their sprig-like leaves (Figure 17). Bog bilberry is hard to distinguish from bilberry, but the assumption was made that bog bilberry mislabelling is assumed not to cause issues due to species assumed rarity in images. Therefore, bog bilberries annotated are not likely to cause issues in deep learning due to similarity with bilberry, thus adding to the capability to detect bilberries.

The annotation process's main issue with personnel annotating was the difficulty of deciding whether to annotate barely visible assumed berries or not, for example, in Figure 17, middle image. Images were instructed to be taken from multiple distances and angles, so many images contained edge cases of berry phases on the edge of visibility. The decision to annotate is dependent on the person annotating. As a rule, annotating personnel were directed to annotate in cases when the berry was assumed recognisable as a separate cropped image. Although human intuition suggests that a white spot between lingonberry-like leaves is most likely a lingonberry flower bunch, it also may not be. The human decision can be based on overall picture properties, not on features in the object. Annotating too mysterious objects were assumed to cause exaggerative sensitivity in deep learning model results.

Berry changes its phase from flower to raw and finally ripe. This process happens gradually, and a borderline state exists between phases (Figure 17, last image). Converting continuous pixel value image data to nominal annotation label data is bound to cause difficulties in borderline cases. During the annotation process, this borderline issue was nominal. Most likely due to the timing when data was collected. Field measurements were timed to happen when berries in the area were almost entirely in the target phase, minimising the risk of unnecessary field trips. The timing was adjusted based on summer 2021 conditions. Berries, mainly in the target phase, have the following result: the dataset does not contain many borderline cases based on observations during the annotation process. The limited number of borderline cases has the unwanted consequence that insufficient data on how the deep learning model will behave with borderline cases is unavailable.

3.3 Benchmarking dataset

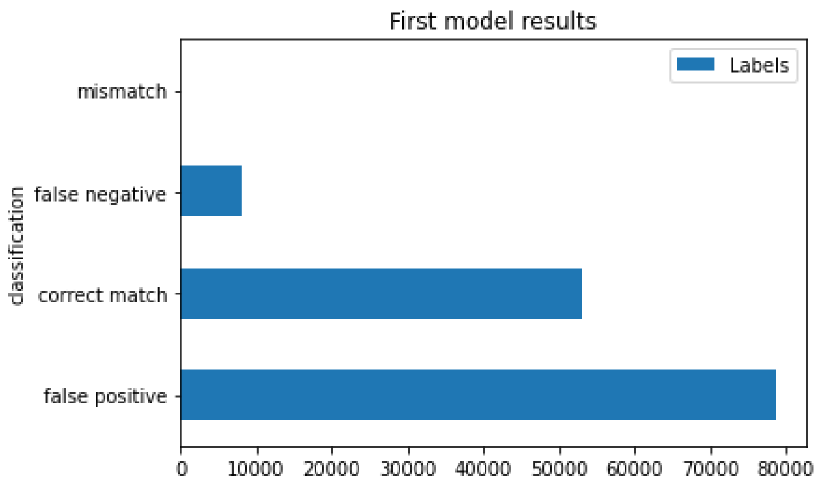


Figure 18. The first model compared the annotated dataset with a 50% IOU Threshold.

The first model with nine classes trained resulted in a large amount of false positive detection (Figure 18). The possibility of mismatches was found to be nominal. The number of false negatives was low compared to false positives. The main issue was to solve false positives and find possible reasons for such a high number.

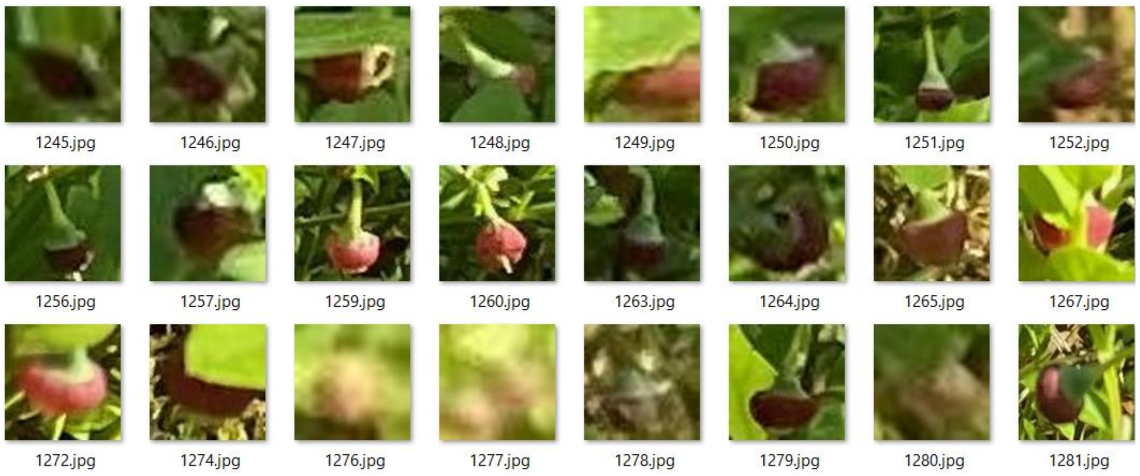


Figure 19. Sample of initial bilberry flower detections classified as false positive.

For determining the nature of false positives, detected labels were cut into separate images divided into folders mismatch, false negative, correct match and false negative for visual inspection. Visual inspection of false positives revealed many correct matches (Figure 19). It is unclear why such a large number of berries was not annotated, but an assumption was made based on personnel experiences. In annotating large images, f. ex., 4032x3024 pixels, human concentration was insufficient to find accurately every berry in the image. In many cases, this can be compared to the allegory of finding a needle in the haystack. Therefore, the assumption was made that the original annotation data is inaccurate for benchmarking. Two options remained: 1) Fix the original dataset or 2) Annotate a second smaller but accurate dataset. Option two was selected. Fixing the original dataset was deemed too time-consuming as revisiting every image would likely take as much time as the original annotation. The option to create a specialised application to accept or deny machine learning findings quickly was considered. Still, this solution was abandoned for development time and possibly caused unknown issues.

The second dataset for benchmarking was created. Two changes were made to enable more accurate annotation better to prevent similar issues from emerging in this dataset. 1) All images were 640x640 pixels and thus were much faster to annotate. Smaller images could fit on a computer screen as size 1:1 without zooming. A glance was usually enough to determine all berries in the image. 2) Lingonberries were annotated only as bunches. If only one lingonberry phase were visible, it was instructed to be annotated as a bunch. Both changes simplified the annotation process, and a simple annotation process was assumed to lead to more accurate results. Also, personnel were directed to invest in accuracy detriment to speed. In addition, the newest version of annotating program Label Studio was deployed. Some glitches were experienced with the initial version, but smaller image sizes with version updates are assumed to make annotation more graceful.

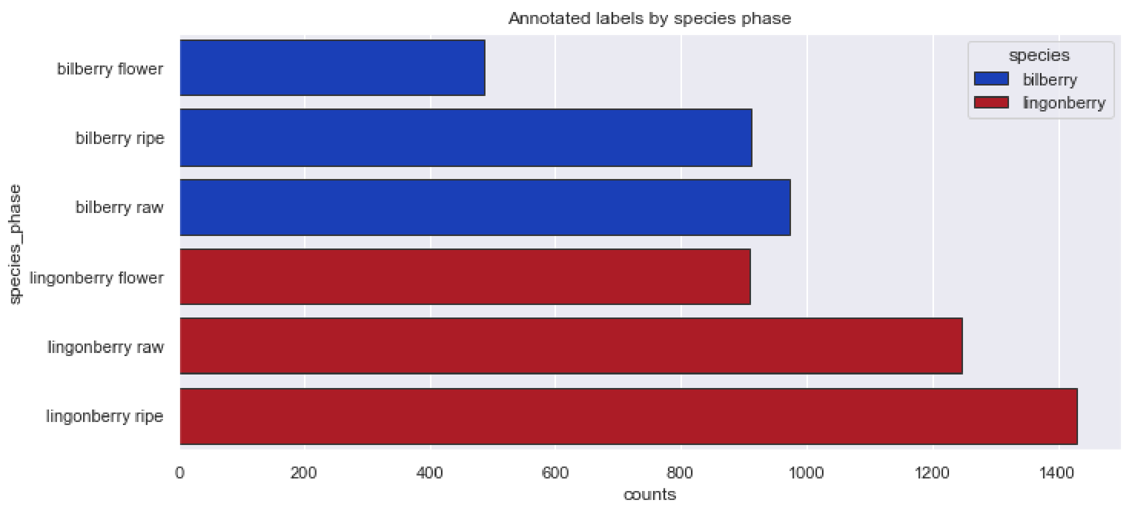


Figure 20. Benchmark dataset number of labels.

A random selection of 640x640 images with a balanced number of labels was selected for benchmark dataset annotation. The data set was formed from already annotated images to balance labels. Selected images were split into smaller images. Annotated benchmark dataset consisted of 2034 640x640 images with 5958 labels. The final size was formed based on personnel time available for annotation. Benchmark dataset quality was reviewed using the latest deep learning model and comparing results to annotations. Labels were cut into separate images. Label images were divided into folders mismatch, false negative, correct match and false negative for visual inspection like previously with original annotations. Only a tiny number of annotation errors were found, and these errors were corrected in the annotation. This dataset is used to benchmark deep learning models. The number of species phases in the benchmark dataset is significantly unbalanced, as shown in Figure 20.

3.4 Count frame image dataset

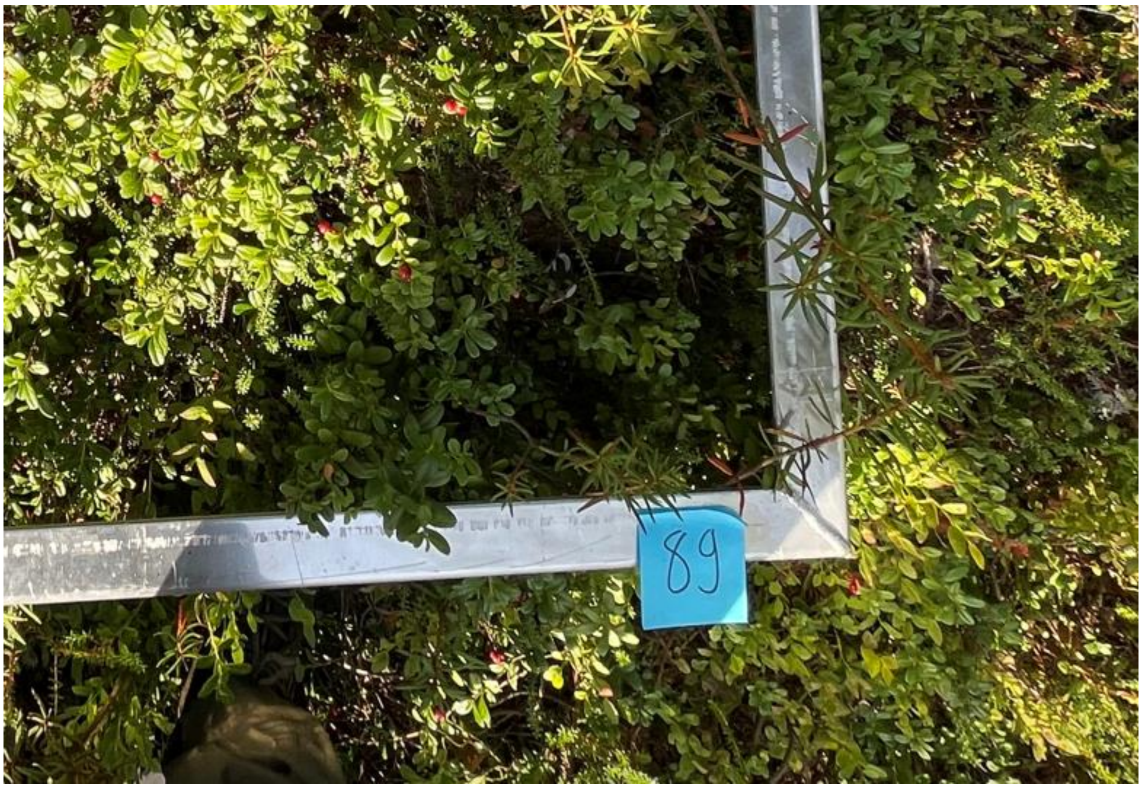


Figure 21. Berry count result documented in field measurement. 89 berries counted inside the frame.

From count frame images, two datasets were created. The first dataset table was to link field measurement count number (Figure 21), species counted, and phase to file path. This data is to label the image with the actual berry density value for final comparison. The second dataset was frame geometry annotations to enable more accurate area estimation through cropping or ground plane estimation.

All frames in prediction dataset images (expanded dataset including annotated dataset) were collected in a data table. Collected information for a single image row was:

- File path
- berries per m² from post-it paper visible in image or image location with known berry density
- Filename
- Count label. Berry species and phase counted to post-IT paper
- Frame visibility. Is the frame entirely or mainly visible in the image? This Boolean value is to separate close-up images from "full frame" images

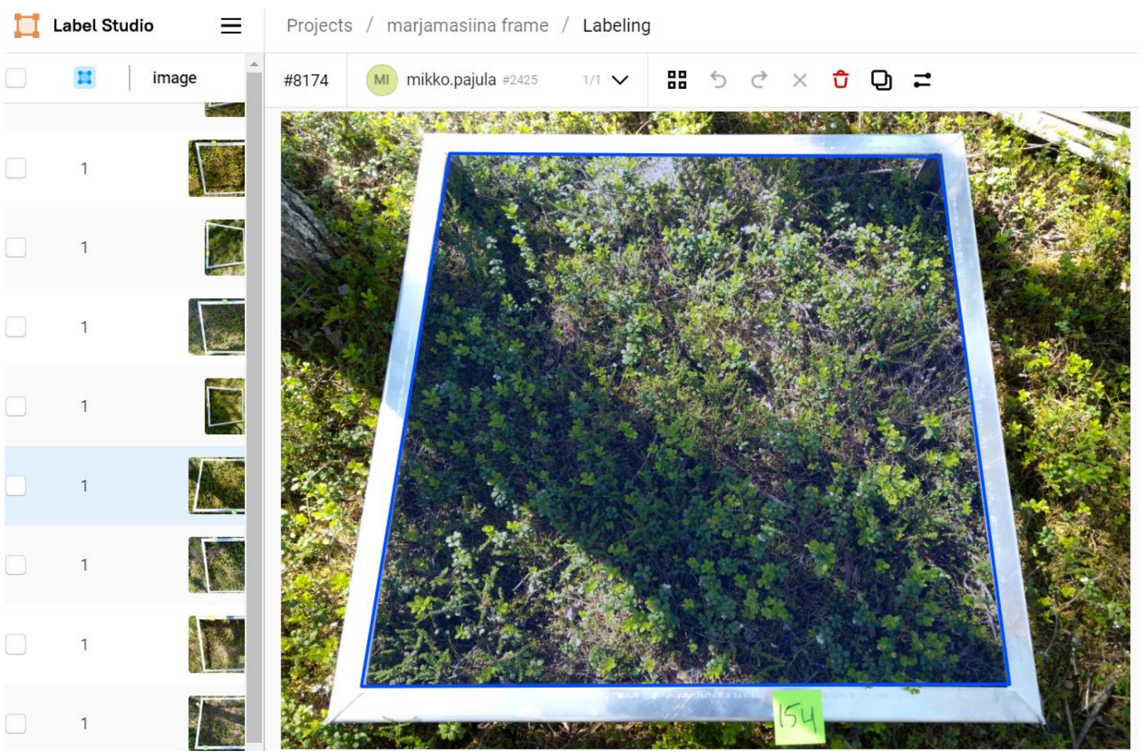


Figure 22. Frame annotation with Label Studio Segmentation mode.

Berry frames are annotated as a polygon in image coordinates to enable more accurate image cover area estimation in berry density analysis. Label Studio segmentation was used as an annotation tool as this tool was already familiar, segmentation labelling is the same as polygon draw, and Label Studio format parsing tools were already developed within the project (Figure 22). Label Studio segmentation results were frames as polygons. This data was not used as machine learning segmentation training but with cropping berries from inside and homography transformation. Frame annotation process directions were to annotate frames with four edges only. If a frame was not entirely inside the image, the frame was not annotated.

4 DATA PROCESS PIPELINE

After data collection and annotation, created dataset format was standardised and flattened into a single folder. Images are also split into smaller images for training to prevent loss of detail. These and other steps require an image processing pipeline to be implemented. Pipeline was described in chapter 4.1 Data formats in and from label studio.

Before the first trained model, it was decided not to scale images to preserve all available pixels when detecting small objects like berries. However, as object detections have a native resolution, writing a script to split images into smaller ones was necessary before using an image for training or inference. The image split process was described in chapter 4.2 Splitting images.

4.1 Data formats in and from label studio

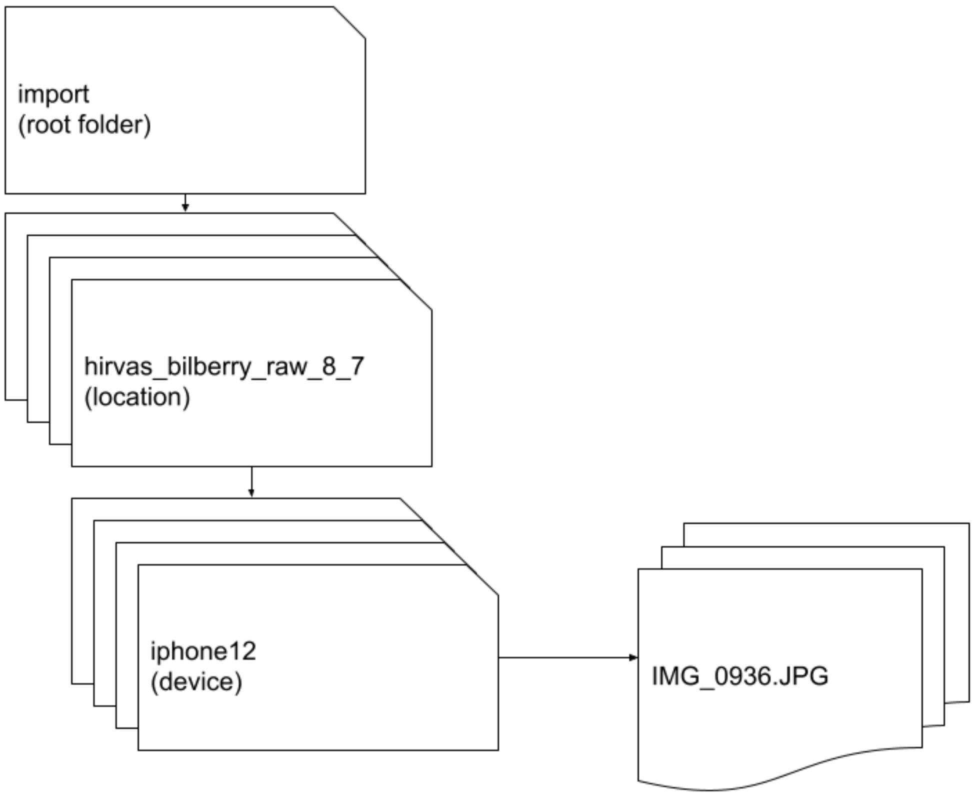


Figure 23. Annotated dataset folder structure.

Label Studio annotation application can read image datasets with images in subfolders. The image dataset was annotated in a subfolder structure (Figure 23). Label Studio annotations are exported in Label Studio custom format JSON file.

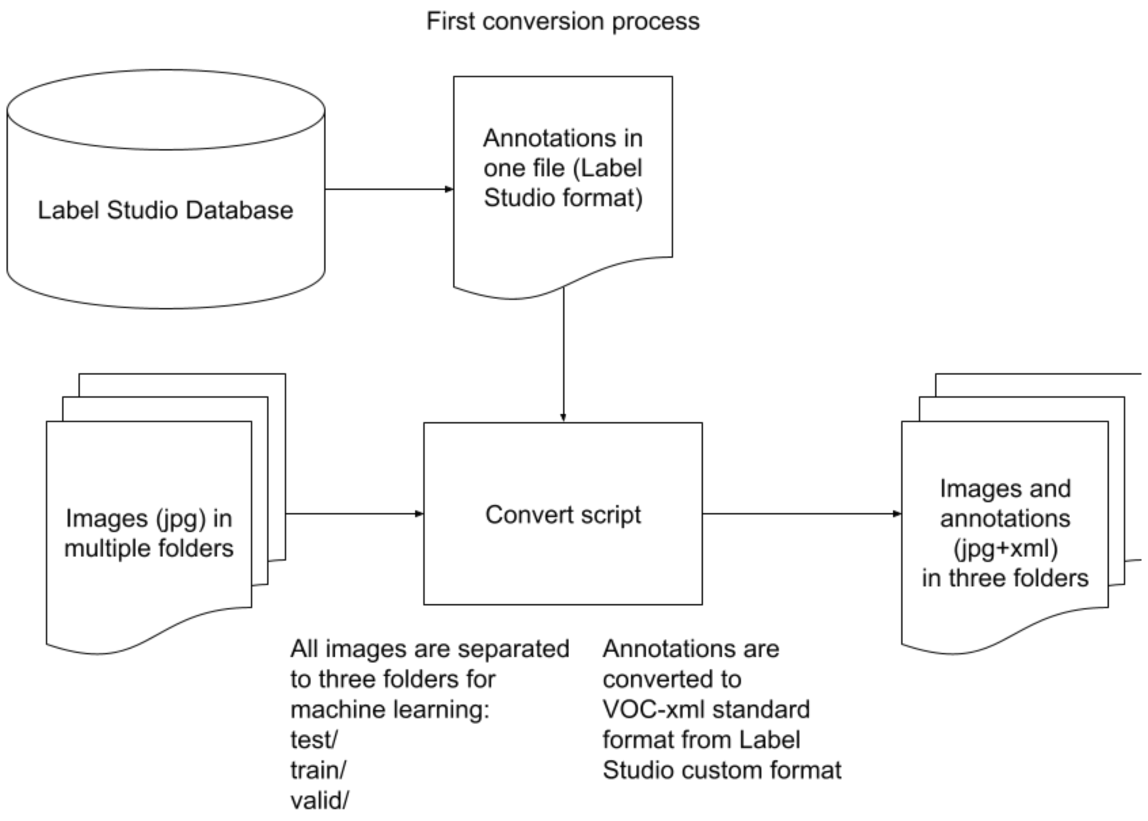

First conversion process
Label Studio Database
Annotations in one file (Label Studio format)
Images (jpg) in multiple folders
Convert script
Images and annotations (jpg+xml) in three folders
All images are separated to three folders for machine learning: test/ train/ valid/
Annotations are converted to VOC-xml standard format from Label Studio custom format

Figure 24. The file conversion process to a more standard format.

For training the deep learning model, a subfolder structure is needed to flatten into a single folder level. Training data with Yolov5 is split into three folders: train, test and validate. Data is split into three folders randomly. No image must have the same filename to be able to split image data in folders without issues randomly. The dataset is flattened into one single folder by renaming files based on the original filename and folder names added to the filename prefix to prevent filename collision. The conversion process (Figure 24) changes the file from hivas_bilberry_raw_8_7/ iphone12/IMG_0936.JPG into hivas_bilberry_raw_8_7_iphone12_IMG_0936.jpg. Location and device information is preserved in the filename. The covert script also creates the hivas_bilberry_raw_8_7_iphone12_IMG_0936.xml -file in the same folder with the image.

Label Studio format was used when exporting annotations from the Label Studio annotation application. Label Studio -format stores annotation coordinates relative to the image width and height. PASCAL VOC is an XML-based format for annotation data (Russell 2008). Pascal VOC

XML -files are stored in the same folder with images. XML files are given the same filename as the corresponding image file but with the .xml filename extension. VOC format benefit is that it is widely supported in different annotation tools (Puertas 2022). Widely supported format enables annotated dataset visual inspection with a tool, like LabelImg (Tzutalin 2015) and assumed ease to change neural network framework. decision was made to use Pascal VOC as default annotation data format.

4.2 Splitting images

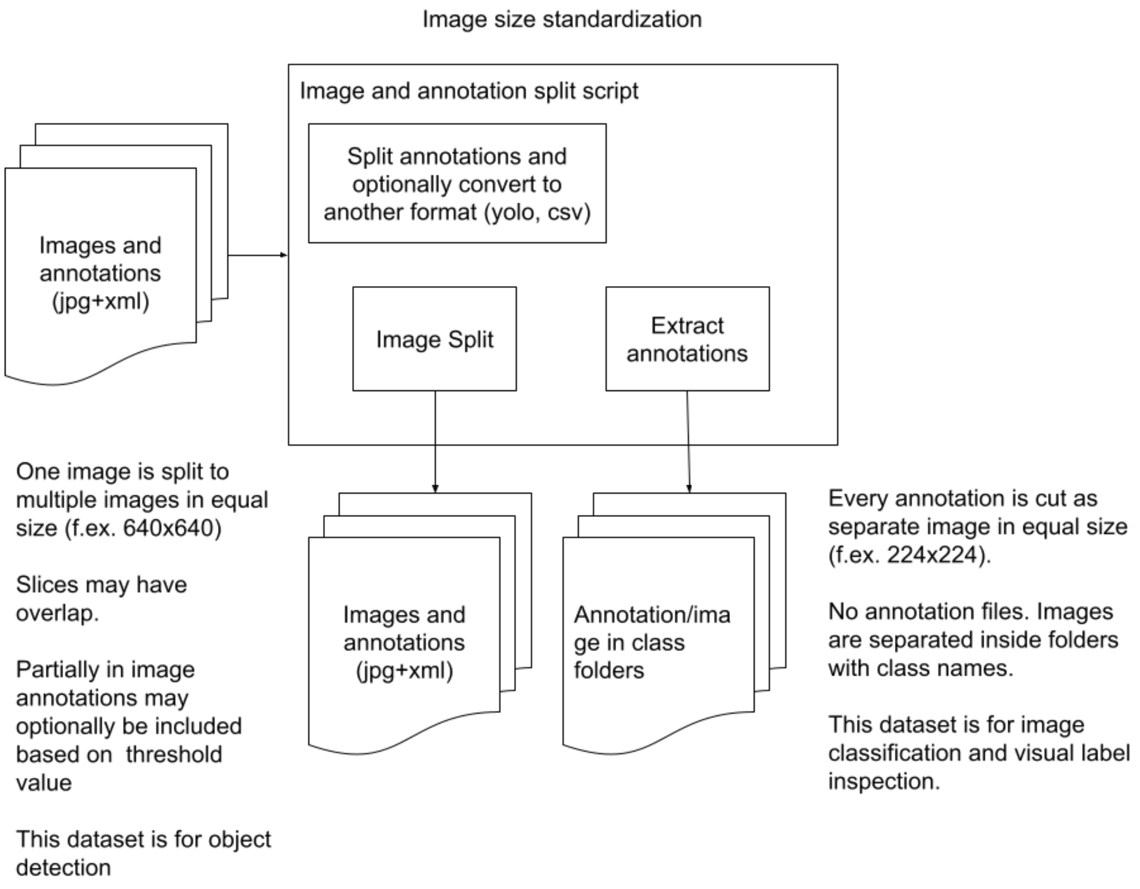


Image size standardization
Image and annotation split script
Split annotations and optionally convert to another format (yolo, csv)
Images and annotations (jpg+xml)
Image Split
Extract annotations
One image is split to multiple images in equal size (f.ex. 640x640)
Slices may have overlap.
Partially in image annotations may optionally be included based on threshold value
This dataset is for object detection
Images and annotations (jpg+xml)
Annotation/image in class folders
Every annotation is cut as separate image in equal size (f.ex. 224x224).
No annotation files. Images are separated inside folders with class names.
This dataset is for image classification and visual label inspection.

Figure 25. Splitting image process

Split process (Figure 25) input is an image and Pascal VOC format annotations. Output in the format defined in script options. The main task was to split image data into neural network native resolution. With annotation data, this includes several tasks: 1) Select labels for each split image based on object location being inside split image bounds. 2) covert annotation coordinates to the split image extend 3) save split image corresponding annotation in the format defined in options.

Annotated dataset after the converter was formed from images and Pascal VOC -files in a flat folder structure. Depending on the selected framework, the data format may not be suitable for machine learning training. In the case of Yolov5 large, 640x640 pixel image size is the native resolution, and label format is YOLO-format. YOLO-format is a txt file with the same filename as the corresponding image. Inside the file, each text row is a single bounding box -annotation:

1 0.160938 0.085938 0.025000 0.040625

On each row, data cells are separated by space. The file has no header row, and the format follows the header: object-id x-centre y-centre width height (Hatab 2020). Coordinate units are relative to the image width and height. With Yolov5 image split script output was 640x640 pixel images with YOLO-format annotations.

The split script can also extract annotations in separate images inside a folder name based on a label. This option allowed producing datasets for image classification and was an easy way to inspect annotations visually.

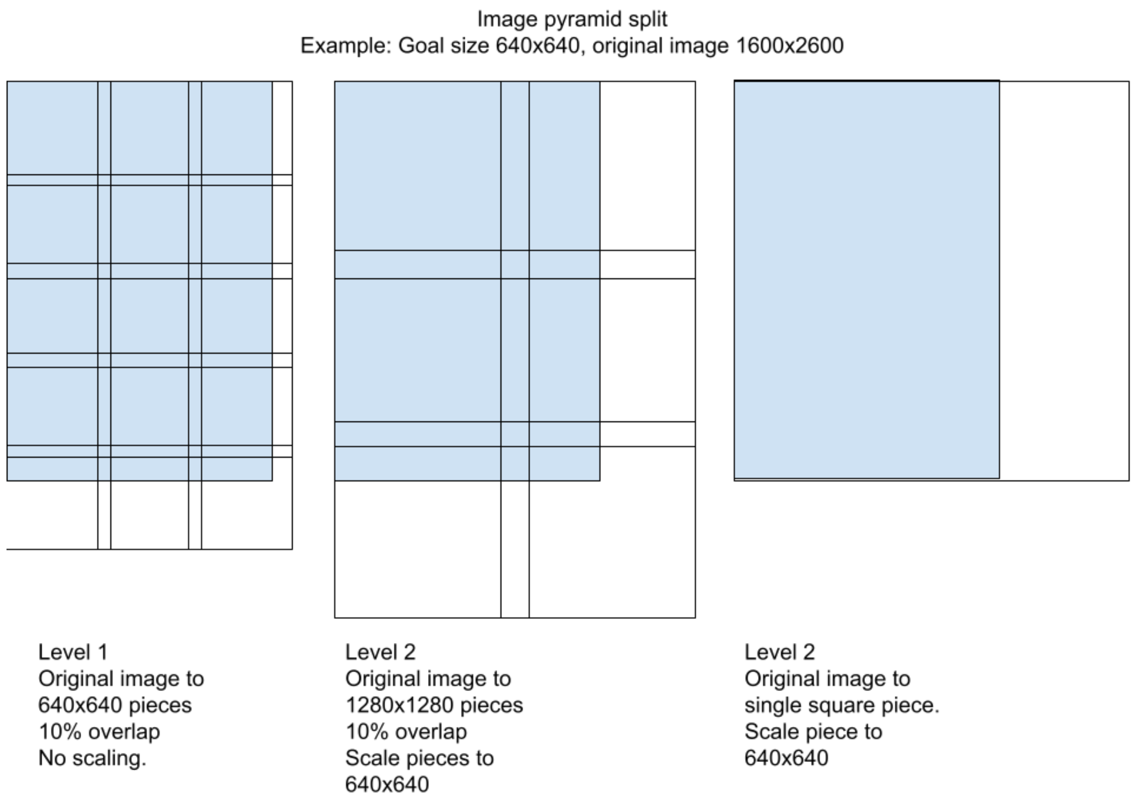


Figure 26. Pyramid split

Multiple ways to split an image into smaller parts exist. An equal split is the simplest way to split an image into smaller pieces. The image was split, defined size pieces with a grid. The first version of the object detection training dataset used a pyramid split with overlap (Figure 26). Assumed performance improvements were:

1) the Same annotation was in training data several times in differently scaled versions
2) With overlap, overlapping annotations were more likely included in the dataset
3) Generated machine learning model was capable of detecting close-up images of berries

Original image sizes were not divisible by split size, so the spare part of a split image is filled with black. If no annotation was inside a split image, this image was rejected from the final dataset. Only images with annotations were kept.

In the pyramid split, annotation coordinates were also scaled when scaling images. The annotation was discarded if the annotation size dropped below 13px due to the annotation coordinate scaling. The pyramid split process still produced a more significant amount of label annotations per single full image than the original label amount due to single labels being visible in all zoom levels. However, the annotation size threshold removed some amount. If used, the overlap between images added more labels than initially present in images.

4.3 Detection process and prediction dataset

A detection or inference pipeline was developed to support multiple use cases. The detection pipeline process's main requirement was to fulfil three inference cases:

1. Berry density estimation using deep learning prediction results required an inference process.
2. Benchmark dataset inference results were needed to compare prediction results with benchmark dataset annotations.
3. The prototype application required a machine learning model inference backend.

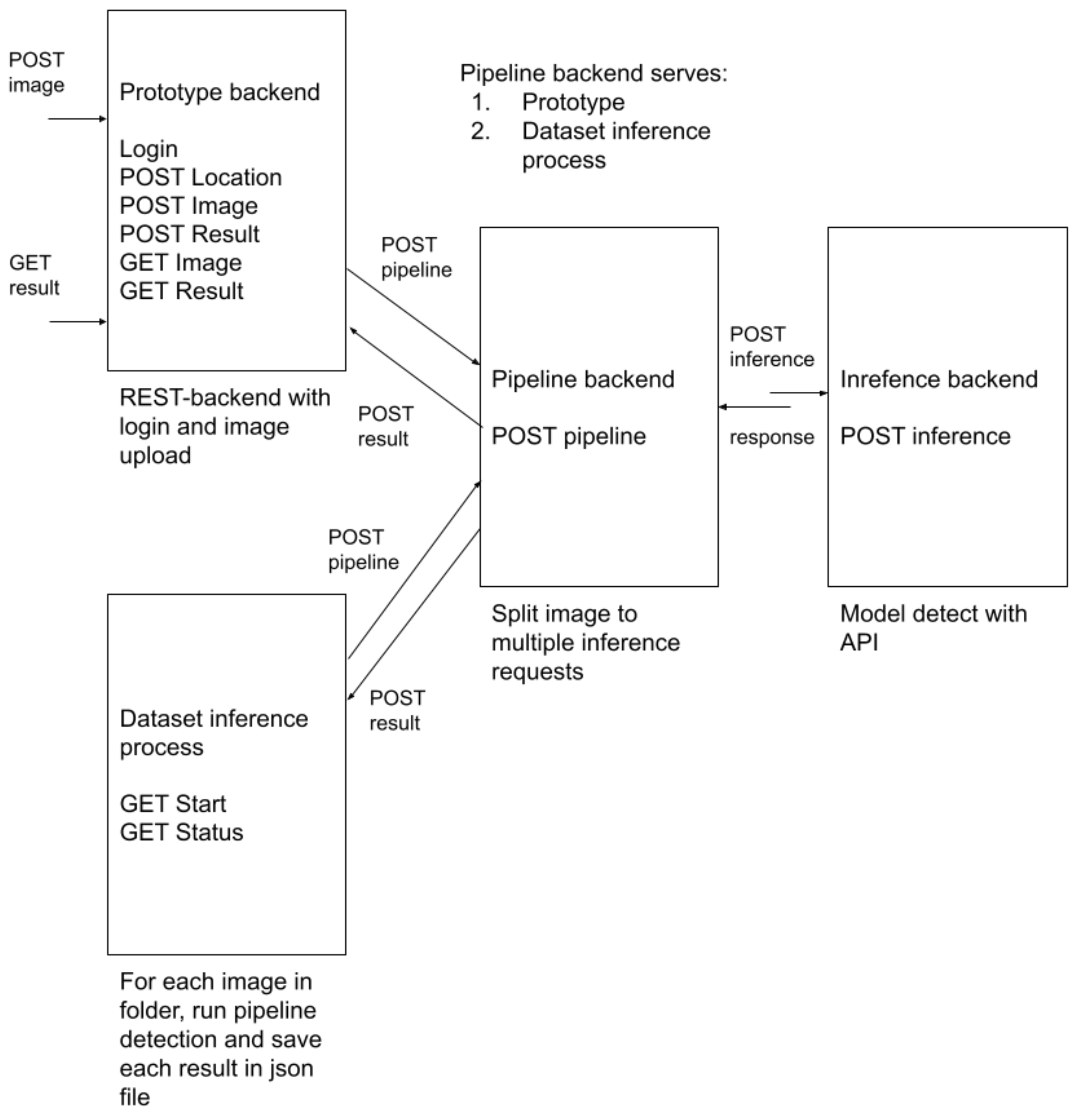


Figure 27. Prototype or dataset inference pipeline backend

For the detection process, Inference speed was not required but preferred as one single image is split into multiple images and amount of full images was in the thousands. With a CPU-only Kubernetes inference server with a limited CPU quota, the detection process for all image pieces can take minutes. With a GPU-supported dedicated server, detecting all image pieces can take only seconds. Therefore, one architecture requirement was the capability to deploy the inference process separately (Figure 27).

Main properties for the inference pipeline backend:

1. Receive image as URL and respond task as received.

2. Separate inference backend for modularity to allow deployment of the inference backend into a dedicated GPU-supported server.
3. Split image and host image pieces for inference backend. The split technique is kept the same as the training data split technique.
4. Orchestrate image piece inference process into a queue of requests to the inference backend.
5. Merge detection result labels into original image coordinates.
6. Delete image pieces after inference.
7. Optionally draw detection labels into the image for visualisation purposes.
8. Notify that the image process is ready at the given address. The processing time of a single image may reach minutes highly depending on the inference server.

The entire backend architecture contained three separate backends (Figure 27). The first part is the main backend with API (Application Programming Interface) to act as the main entry point to the user. This main backend differed depending on the user's task. For example, if the task was a prototype application, the main backend was REST (Representational state transfer) API. API can save the image, and on image receive, request the pipeline backend to start inference and receive results from the pipeline backend. The prototype backend stores results and the user can request results for each corresponding image. Dataset inference process was another main entry point backend. Dataset process looped all images in the folder and ran the pipeline backend process for each image. After receiving the result, it saved it to a separate JSON (JavaScript Object Notation) file.

The pipeline backend split images, hosted image pieces, requested inference backend, merged result labels in original image coordinates, and passed results by requesting a predefined address. The inference backend was a minimal API with an endpoint to request model inference. It responded with detected labels in the JSON array.

From the dataset of approximately 10000 images, 5000 images were selected for processing through the inference pipeline as a prediction dataset. This dataset contained all 3081 images selected for the annotation process. Processing annotated images enable comparing annotations with machine learning predictions.

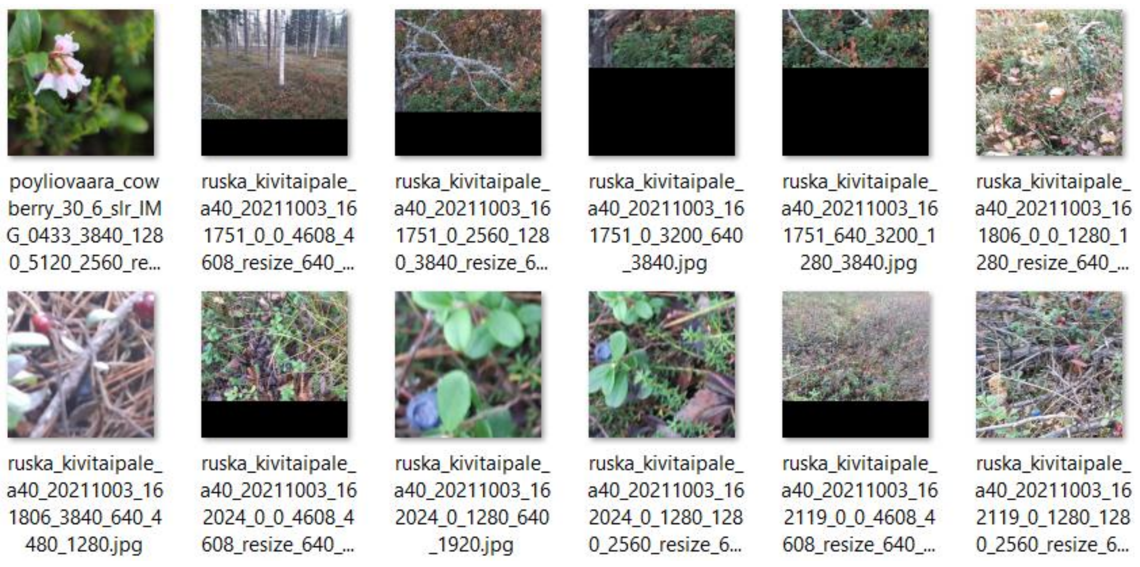
5 MODEL TRAINING PROCESS

After the data process pipeline, the dataset was ready to be used as a training dataset for machine learning. First, an initial model was trained from the dataset created by the steps described previously.

Machine learning model results were analysed to study achieved accuracy and find ways to improve learning results. Input data was adjusted when the initial model results indicated issues in the training dataset. Different data process pipeline configurations were used with the adjusted dataset to train four models. Finally, model results were compared with the help of the benchmarking dataset. The fourth model was selected as the final machine learning model for the density estimation chapter based on the comparison results.

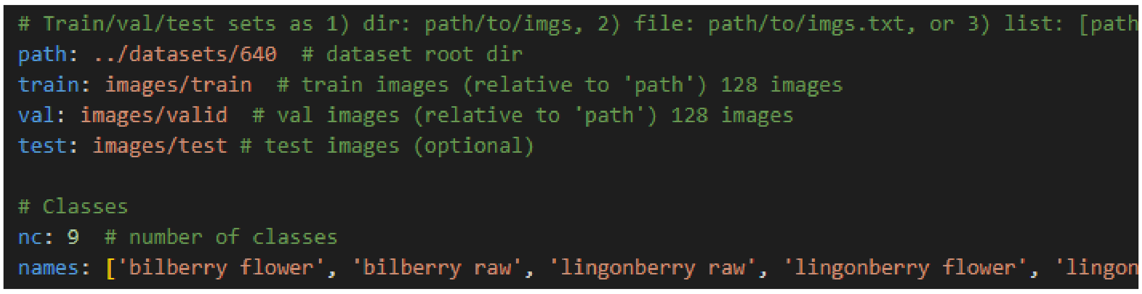
5.1 Training the initial model

Pyramid split produced 39 949 split images from 2855 primary annotated images—an average of 13 split images from one image. A pyramid split could produce a much larger number of image pieces. Still, image pieces with zero annotations were discarded, so the final number of split images was much smaller than the theoretical maximum. For the first training, the pyramid split overlap between the same zoom level images was set to zero percentage (0%).


poyliovaara_cowberry_30_6_slr_IMG_0433_3840_1280_5120_2560_re...
ruska_kivitaipale_a40_20211003_161751_0_0_4608_4608_resize_640_...
ruska_kivitaipale_a40_20211003_161751_0_2560_1280_3840_resize_6...
ruska_kivitaipale_a40_20211003_161751_0_3200_640_3840.jpg
ruska_kivitaipale_a40_20211003_161751_640_3200_1280_3840.jpg
ruska_kivitaipale_a40_20211003_161806_0_0_1280_1280_resize_640_...
ruska_kivitaipale_a40_20211003_161806_3840_640_4480_1280.jpg
ruska_kivitaipale_a40_20211003_162024_0_0_4608_4608_resize_640_...
ruska_kivitaipale_a40_20211003_162024_0_1280_640_1920.jpg
ruska_kivitaipale_a40_20211003_162024_0_1280_1280_2560_resize_6...
ruska_kivitaipale_a40_20211003_162119_0_0_4608_4608_resize_640_...
ruska_kivitaipale_a40_20211003_162119_0_1280_1280_2560_resize_6...
Figure 28. Examples of first training data.

39 949 split images contained 185 191 annotated labels. The original annotated dataset contained 61 269 annotations. Pyramid split with overlap generated more than double the number of labels into split images (explained previously). An example of pyramid split training data is shown in Figure 28. This split image dataset was divided into three groups:

1. Train dataset (27 727 images) 70%
2. Test dataset (3 993 images) 10%
3. Validation dataset (8 229 images) 20%

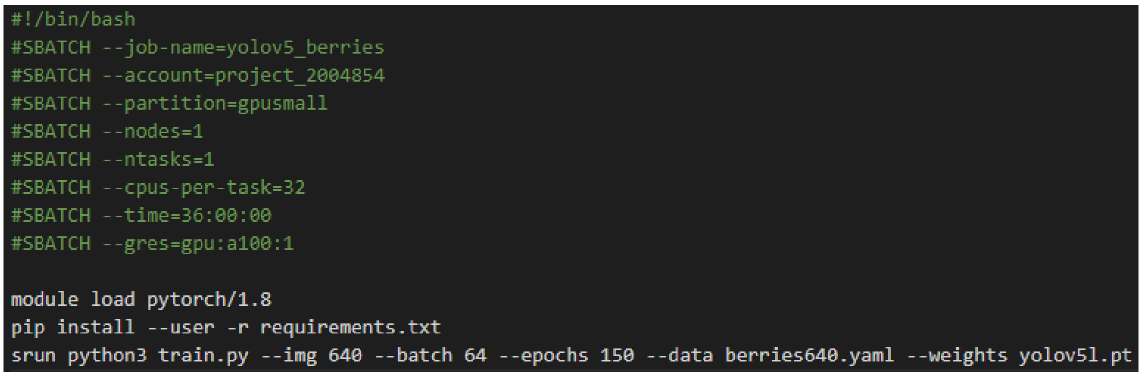


```
# Train/val/test sets as 1) dir: path/to/imgs, 2) file: path/to/imgs.txt, or 3) list: [path
path: ../datasets/640  # dataset root dir
train: images/train  # train images (relative to 'path') 128 images
val: images/valid  # val images (relative to 'path') 128 images
test: images/test # test images (optional)

# Classes
nc: 9  # number of classes
names: ['bilberry flower', 'bilberry raw', 'lingonberry raw', 'lingonberry flower', 'lingon
```

Figure 29. YOLOv5 dataset YAML config file content example.

Annotation labels were saved into YOLO format for the YOLOv5 object detector. The training was done with the 2021 autumn version of YOLOv5. Separate image and label folders were created with identical folder structures (train, test, valid) inside with image files and corresponding text label files in the labels folder. The folder structure with YOLOv5 is added to created dataset config file (Figure 29).



```
#!/bin/bash
#SBATCH --job-name=yolov5_berries
#SBATCH --account=project_2004854
#SBATCH --partition=gpusmall
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=32
#SBATCH --time=36:00:00
#SBATCH --gres=gpu:a100:1

module load pytorch/1.8
pip install --user -r requirements.txt
srun python3 train.py --img 640 --batch 64 --epochs 150 --data berries640.yaml --weights yolov5l.pt
```

Figure 30. Mahti supercomputer batch job script.

The training was done with CSC (IT Center for Science) Mahti supercomputer (Center for Science 2020). Training is done via a patch script (Figure 30) added to the work queue. The training was supported with Nvidia A100 GPU nodes available in the Mahti supercomputer (Center for Science 2022). Batch script important defined settings were:

- 32 CPU nodes
- 1 Nvidia A100 with 40GB memory
- 36 hours maximum amount of processing time
- Image size 640px
- 150 epochs
- Batch size 64
- YOLOv4 5l model with pre-trained weights
- Dataset config file

5.2 Analysing the initial model results

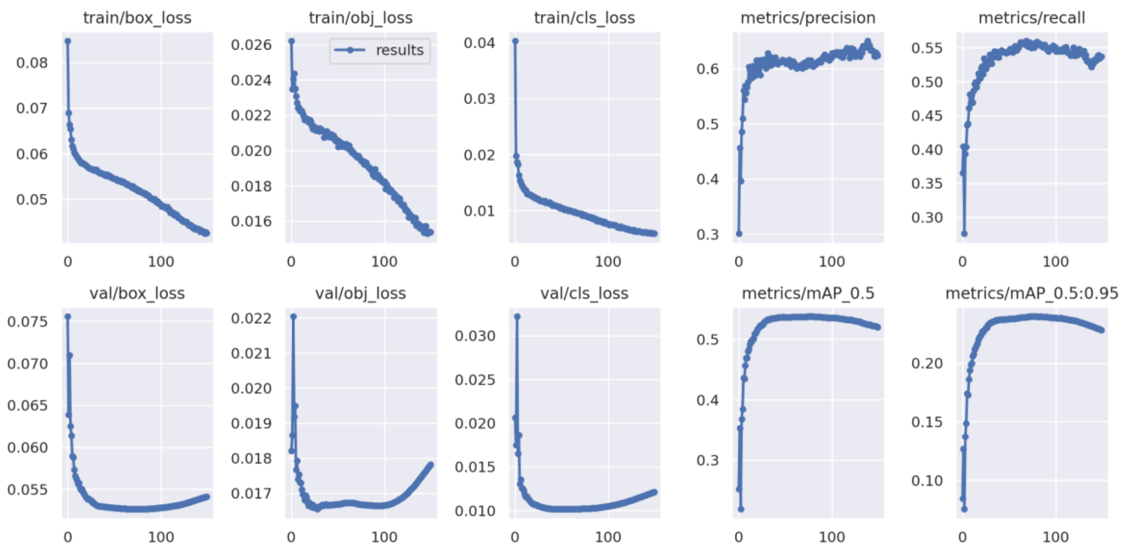


Figure 31. Initial model training results.

150 epochs were selected as the maximum epoch number after initial test training runs, indicating overfitting after 100 epochs (Figure 31) in validation results as metrics/mAP get worse after 100 epochs. YOLOv5 saves two model weight files. Best and last The last is model weights after 150 epochs. The best weight file is always used. Metrics define the best epoch. The batch size selected was 64. The batch size was selected based on initial test training runs as large as possible with 40GB GPU memory.

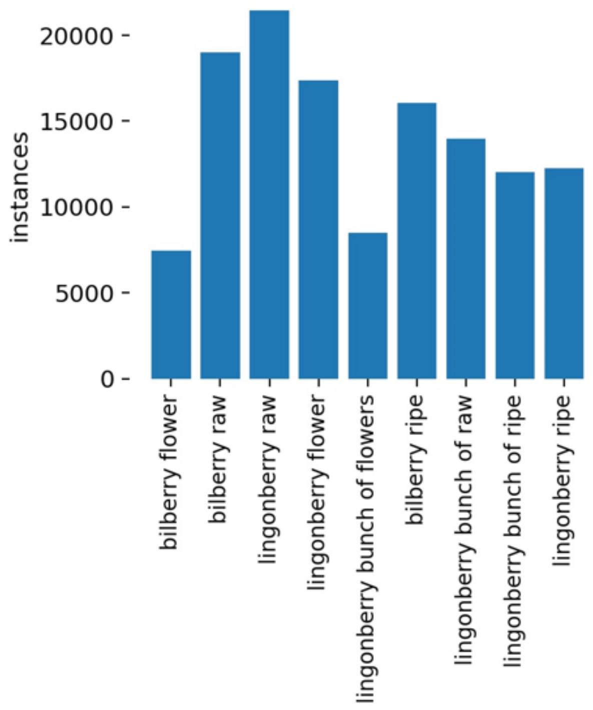


Figure 32. Initial model annotation instances per class.

Training data object instances were unbalanced. Mainly Bilberry flower and lingonberry flower bunch (Figure 32). Autumn 2021 raw lingonberry additions were visible in the dataset as the most numerous class.

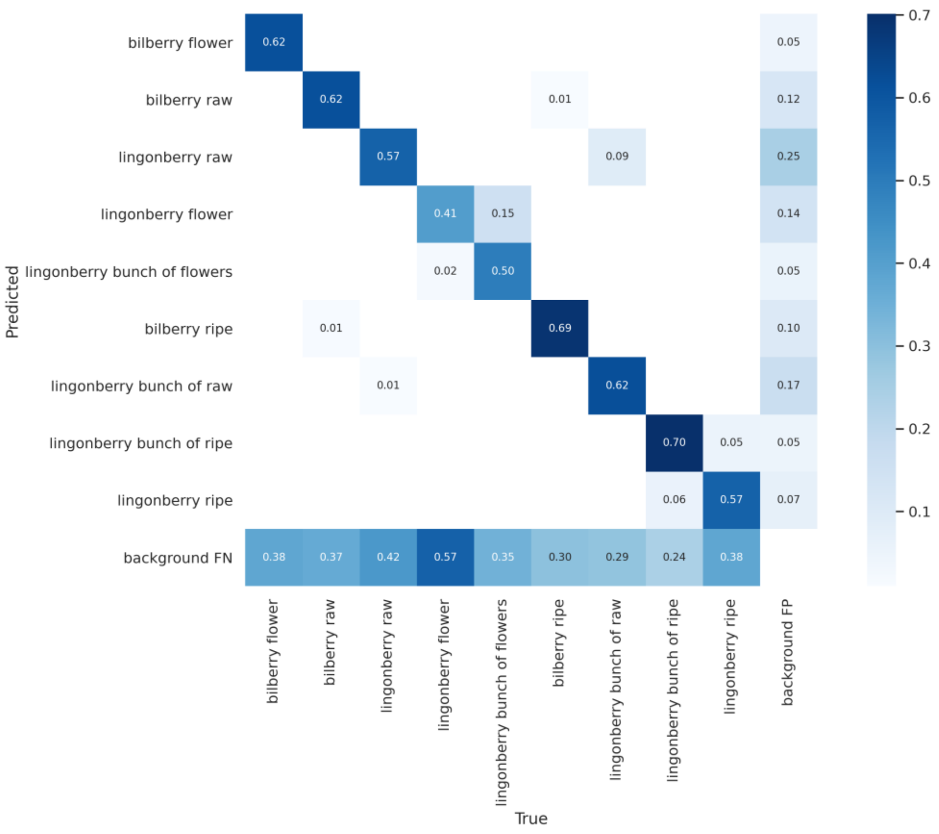


Figure 33. Nine class confusion matrix.

The confusion matrix (Figure 33) indicated a large number of issues. Overall results were weak in the confusion matrix. Single lingonberry flower and a bunch of lingonberry flowers results were especially weak. In Figure 31, metrics indicate 0,6 precision and 0,55 recall.

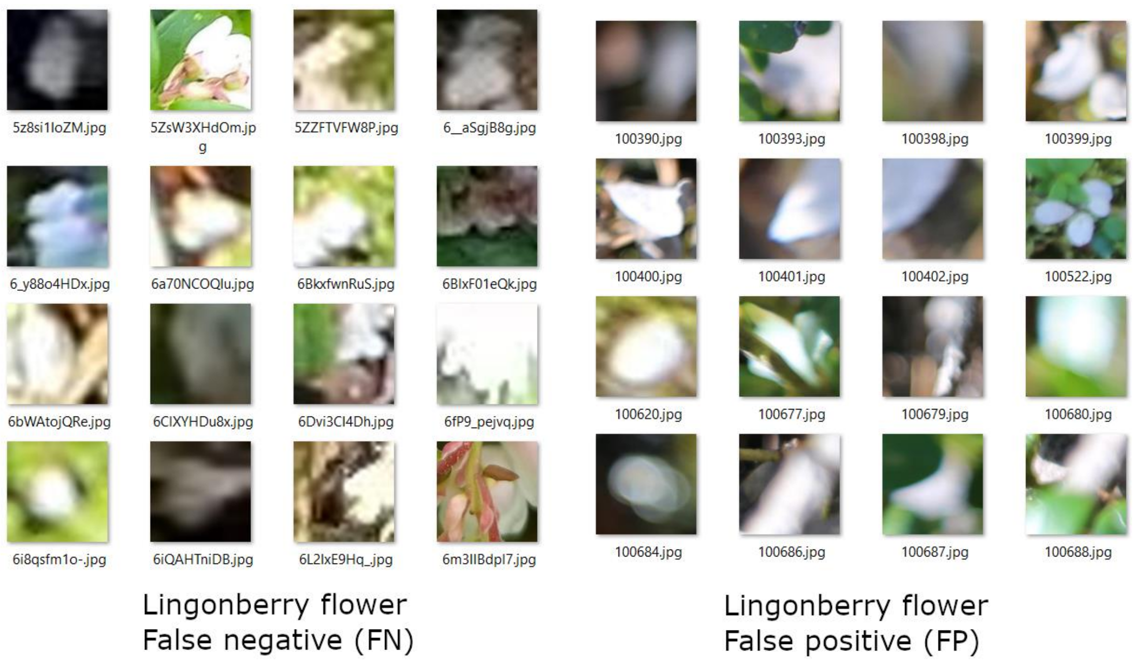


Figure 34. Single lingonberry flower false negative examples compared to false positive examples.

The main issues were in lingonberry classes by confusion matrix (Figure 33). For visual inspection, predictions were cut into separate images in four categories: False positive, False negative match and mismatch. Cutting labels into separate images was done for annotated images, and division was made with intersection over union (IoU) -comparison. IoU-Comparison was made between predictions and annotations. Visual inspection of single lingonberry flower predictions in categories highlighted issues with single lingonberry classes. For example Figure 34, false positive and false negative single lingonberry flower predictions were side by side compared. Single lingonberry class was difficult to detect by humans as a flower often in an image is only a white area.

Visual inspection and confusion matrix indicated that separate single lingonberry detection contains issues within the dataset. One way to solve this was to remove single labels and only detect lingonberries bunches, thus removing the most problematic classes. This conversion in the dataset has the potential also to improve bunch detection results.

5.3 Adjusting data

Results from initial training implied possible changes to the dataset to improve object detection results. As a result, Dataset annotation class changes were implemented, and the image split method was redesigned.

5.3.1 Removing and renaming labels

Having nine labels was not necessary. Original annotated data could be simplified by removing duplicate lingonberry labels but solving nine label issue had multiple challenges. Simply renaming bunch and single berry labels to have the same label left the dataset with overlapping labels of the same lingonberry phases and berry phases cropped differently. Typical annotation of a single lingonberry image contained annotated bunches with single berries inside, and bunches with one berry labelled as a single.

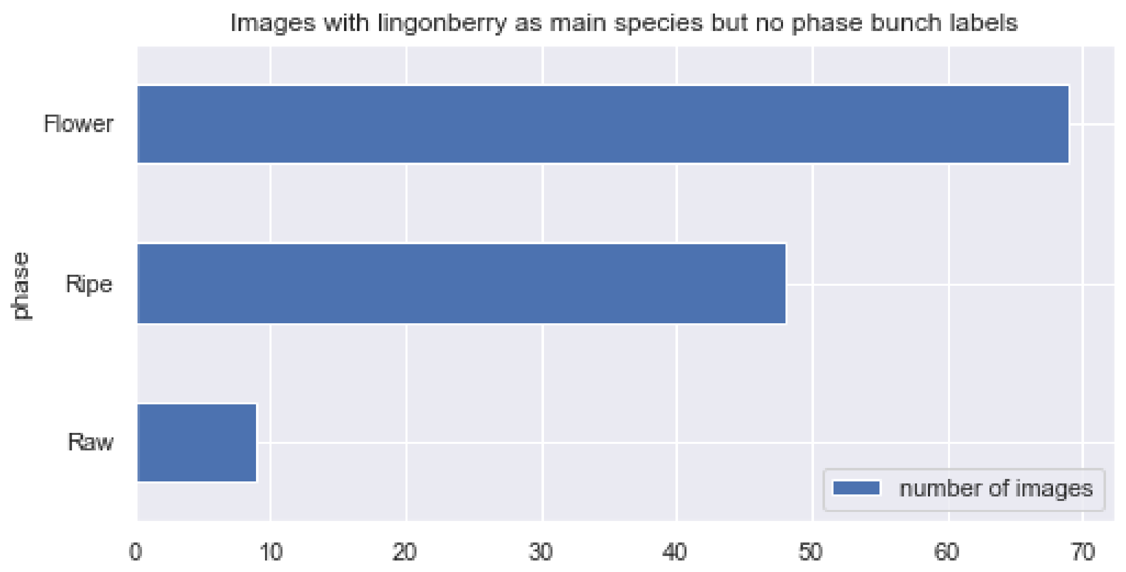


Figure 35. Images with lingonberry as the main species but no bunch labels in the image.

It was detected that the original annotation dataset contains images not properly annotated with bunches. Figure 35 shows the number of lingonberry images without any bunches annotated by phase. Missing bunches were due to some annotations being annotated before a bunch of labels were introduced in the annotation process. All lingonberry images needed relabelling, but approximately 100 images were not corrected. These images were removed from training. Removal was deemed to be acceptable, as it similarly made the final dataset more balanced. Lingonberry flower images were overrepresented in training data when no balancing was done.

Phase	**Species**	**label**
Flower	Bilberry	bilberry flower
Raw	Bilberry	bilberry raw
Ripe	Bilberry	bilberry ripe
Flower	Lingonberry	lingonberry bunch flower
Raw	Lingonberry	lingonberry bunch raw
Ripe	Lingonberry	lingonberry bunch ripe

Table 4. Dataset labels used in later trained object detection models

It was assumed that bunch as a separate label would be a more consistent way to count lingonberries. Single lingonberry phase labels were kept for comparison in the annotation phase. Single lingonberry labels were removed from training data to improve lingonberry detection and limit false positives occurring in the first model with nine classes (Figure 18). The final class list is shown in Table 4.

Bunch did cause challenges when counting a single berry amount in nature. One possible option was deciding the average number of berries/flowers in a bunch and multiplying with that number. Another option was to create an image classification neural network for counting single berry growth instances in a bunch. Still, in this method, the issue arose that not all berries were visible in the image, as an object may be hidden behind each other. Therefore, the loss of the information in the form of a single berry count to gain more accuracy in detection was assumed to be an improvement.

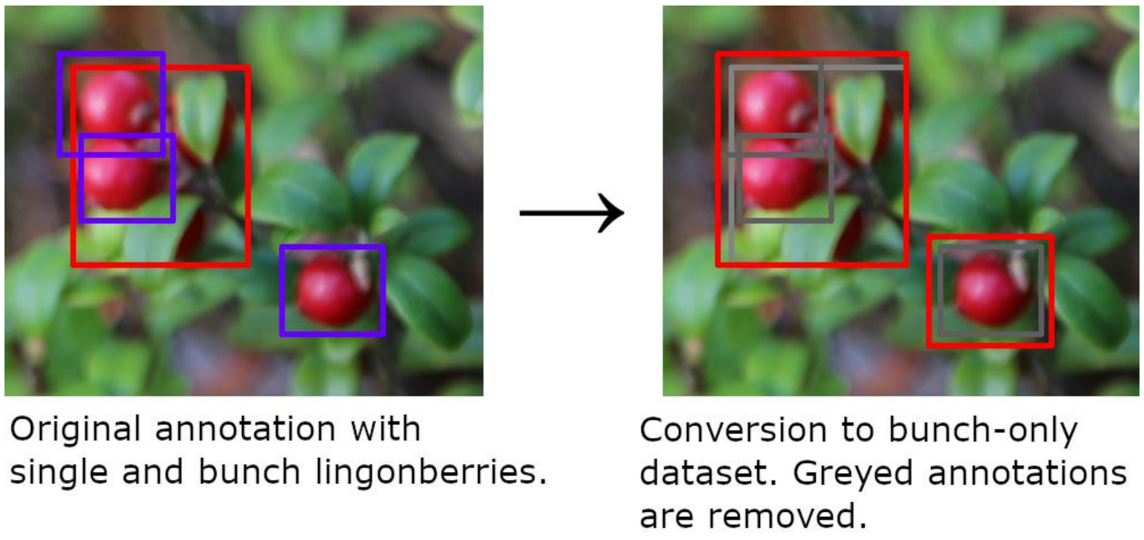

Original annotation with single and bunch lingonberries.
Conversion to bunch-only dataset. Greyed annotations are removed.

Figure 36. Visualisation of bunch-only lingonberry dataset conversion process

A dataset edit script was created to convert the dataset from three lingonberry labels into six labels (Table 4). The script does this conversion while simultaneously designed to solve issues in the dataset. Script steps:

1. Remove images from the dataset with 50% labels are lingonberry, but no bunch of labels present
2. Merge intersecting labels maintain the largest bounding box label name as the label. Set newly created merged label extends to maximum extends of labels in merge
3. Grow remaining single lingonberry bounding box sizes with 10% padding
4. Rename single lingonberry labels to bunch category retaining phase

Removing images with lingonberries but no bunch annotations was done to confront non-corrected annotations. The image was removed if an image had 50% of lingonberries as species but no bunches. A 50% limit was set to prevent removing many images with only a small number of lingonberries. For example, the target label in the image as bilberry ripe may have one or a few single ripe lingonberries visible. This image was acceptably annotated. A visual example of the data conversion process in Figure 36 illustrates merge conversion. Overlapping annotations were merged into a single label with extends from maximum extends from the overlap group. The label name was derived from the largest annotation in a group by size. Single lingonberry labels left in the dataset were resized larger. 10% more padding was added to single lingonberry width and height in all directions providing resize does not extend over image bounds. Resize action was done to uniform lingonberry labels. Bunch labels containing other features forming a bunch were annotated, like a branch connecting berries. Single lingonberry annotations were tightly annotated around a berry flower shape. Adding more padding to single lingonberries left was assumed to help uniform resulting bunch-only dataset.

5.3.2 From pyramid split to simple equal split

With such a large number of false positives, the first model was too sensitive. Pyramid split (Figure 26) was assumed as one cause. Pyramid split scaled images and labels within. Thus, the label with 12x12px bounds was scaled to 6x6px. Scaling caused by the pyramid split was decided to be too small as it was nearly impossible for a human to classify. Therefore, the pyramid split had a minimum label size limit. All annotations smaller than the limit were removed. Scaling created many annotations that were not annotated by humans and likely contained a large number of ambiguous annotations. Arbitrary limit likely removed annotations that were valid by human standards but were removed as too small. This added complexity to the dataset by pyramid split was problematic, and removing the pyramid split could diminish a large number of false positives.

The pyramid split was built with an inference pipeline in mind. As training data needed splitting, so also will the inference process. Detection in prototype and training data split would have the same split technique. Pyramid split theoretical advantage with detection was that predictions overlapping split images were likely detected in different zoom level split if the object is large enough. If the object were too small to be detected after scaling, the split overlap would enable the object to be detected. But with overlap between images at the same zoom level and other zoom levels, the same object could be detected multiple times from multiple split images. Therefore, pyramid split detection requires merging detections. Also, the detection label would affect merging as a slightly overlapping object may occur. Only the same class object would be merged in case of overlapping more than a given threshold. Merging detected berries in different zoom levels increased complexity, so it may uncontrollably affect detection results.

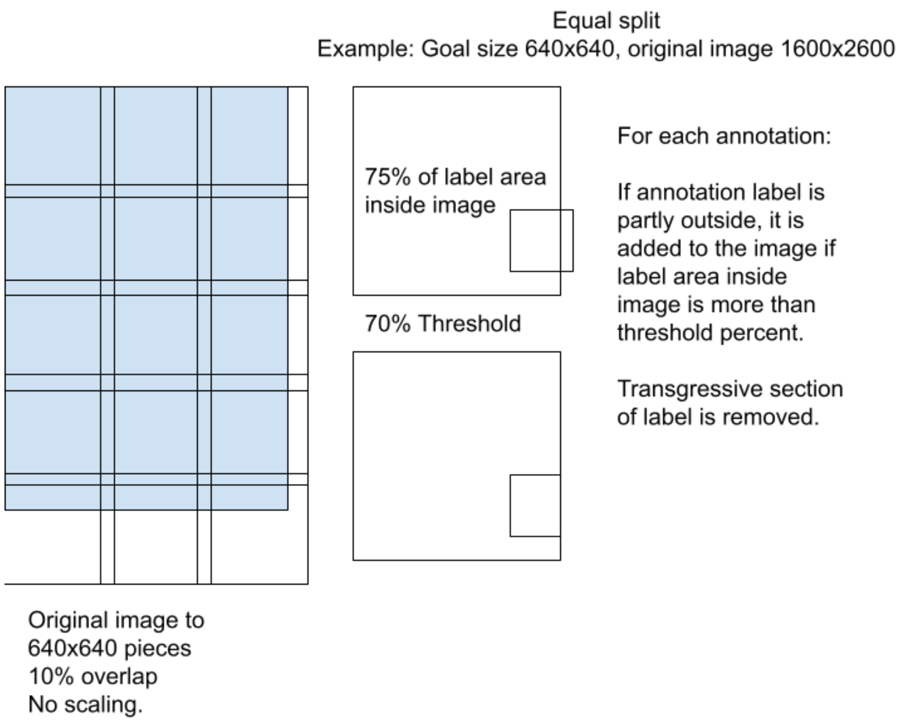

Equal split
Example: Goal size 640x640, original image 1600x2600
75% of label area inside image
70% Threshold
For each annotation:
If annotation label is partly outside, it is added to the image if label area inside image is more than threshold percent.
Transgressive section of label is removed.
Original image to 640x640 pieces
10% overlap
No scaling.

Figure 37. Equal split with optionally overlapping cut grid and threshold to include objects partially outside the piece bounds.

The pyramid split was replaced with an equal split at a 1:1 pixel zoom level. The split codebase is planned to use in inference also due to code reusing advantages. As a detector split technique, pyramid split could detect close-up images of berries and remove double detections of the same berry located between images. A simple equal split in detection was deemed adequate. Detecting berries in close-up images is not essential, and double detection due to the berry located between images is likely to be a minor issue.

Label split must overlap image -threshold percentage	Lost annotations percentage
100%	17%
70%	7%

Table 5. Lost annotations with different thresholds of required overlap with annotation and image.

The pyramid split was done with an overlap in the image grid for each zoom level (Figure 37). The second training dataset removed the overlap property, simplifying the image split process. The dataset image split process loses annotations when the berry was located on the edge of the image.

The overlap threshold lowers the number of labels removed, but some numbers of labels are not included when the threshold is more than 50%. Between two split images. If the threshold is 50% or less, the label was added in both or just one split image. With a 100% overlap requirement, the label must be entirely enclosed in a split image area.

17% of labels were lost in a simple equal split (Table 5), using 100% overlap. Losing 17% of labels is assumed to hurt model training. A large number of labels were only slightly over the image, and thus model received images with berries on the image edge but not annotated. A decision was made to keep the overlap threshold value between 60-90% to prevent images with annotations cut in half or more but to keep as many annotations as possible. With a 70% label threshold, only 7% of the labels were lost. Using an overlap threshold significantly improved from 17% missing with a 100% overlap threshold (Table 5).

5.4 Incremental test training for the adjusted model

New object detection training was initiated with the adjusted dataset and the dataset process pipeline. Training was done incrementally with different dataset pipeline process parameters to evaluate optimal settings. The final, fourth model, was looked into more depth as it created the most accurate results.

Data adjusting decisions were made based on multiple incremental model test training. Common properties applying to all trained models are:

1. Six (6) classes. Single lingonberry classes were removed from all models
2. Instead of the pyramid split, a simple equal split was used with different parameters
3. Model type YOLOv5 Large
4. Modified primary annotated dataset as training data

model date	image overlap	label threshold	Description
8.2.2022	0	1	A balanced dataset with single lingonberries simply removed
28.2.2022	0	1	An unbalanced dataset with single lingonberries simply removed

9.3.2022	0	1	Unbalanced dataset with single to bunch conversion
2.5.2022	0.1	0.8	Unbalanced dataset with single to bunch conversion. Image split with

Table 6. Four six-class model descriptions

The first model (8.2.2022) trained used a balanced dataset with approximately 4000 instances in each class. This number was the number of bilberry flower instances—the lowest class amount in the dataset (Figure 15. Annotated labels count). The first model dataset modification removed single lingonberry labels but did not merge them with bunches. Instead, a simple equal split method was used with no overlap between split images. The label threshold was 1. Therefore, all labels not entirely inside the split image were discarded.

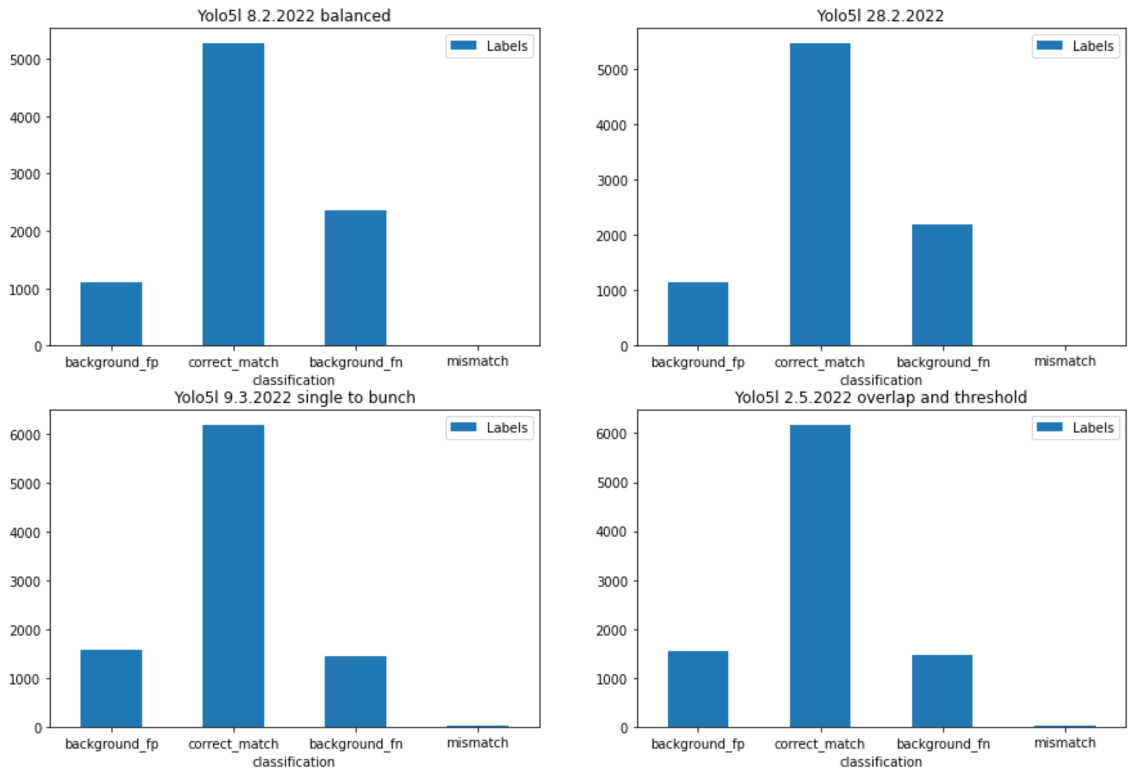


Figure 38. Benchmark dataset results comparison between four models.

The second model (28.2.2022) was like the first model but without dataset balancing. The second model contained more class instances. This training was to test balancing dataset impact on benchmark results. The second model was 62% correct, while the first was 60% correct. The larger unbalanced dataset had better benchmark dataset results (Figure 38). Mismatch represents a label with 50% IoU but the wrong label compared to the annotated benchmark label.

The third model (9.3.2022) implemented fully the steps described adjusted data -chapter. Dataset used was a bunch-only lingonberry dataset done with the conversion script process. Less false negative results were the most likely consequence of changing single lingonberries to bunches. Therefore, the third model was more capable of detecting bunches with only single lingonberry.

5.5 Final and fourth model results

The fourth model (2.5.2022) implemented the fully steps described adjusting data -chapter. In addition, compared to the third model, image overlap was set to 10% (Table 6) and label overlap to 80%. Benchmark dataset results did not improve noticeably compared to the third model (Figure 38).

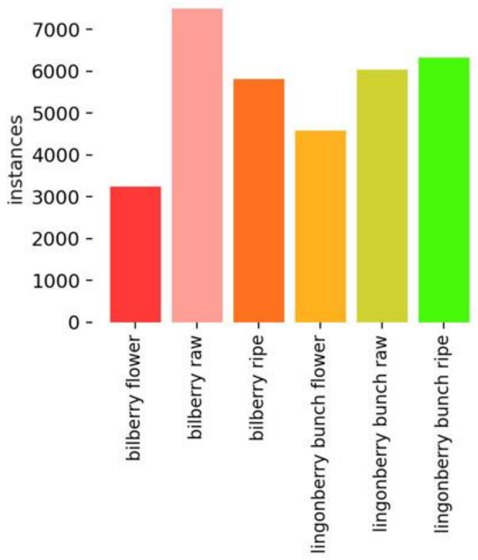


Figure 39. Fourth model annotation instances per class.

Training dataset unbalance remained in six class data (Figure 39) as in initial model training. Merging single lingonberries with a bunch of lingonberries solved most lingonberry unbalance issues. The number of bilberry instances related to each other remained similar to the initial training. Total amounts differed from the initial due to split method differences. For example, a pyramid split created more instances than a simple equal split. Overall, merging lingonberry classes solved some imbalance issues in the dataset.

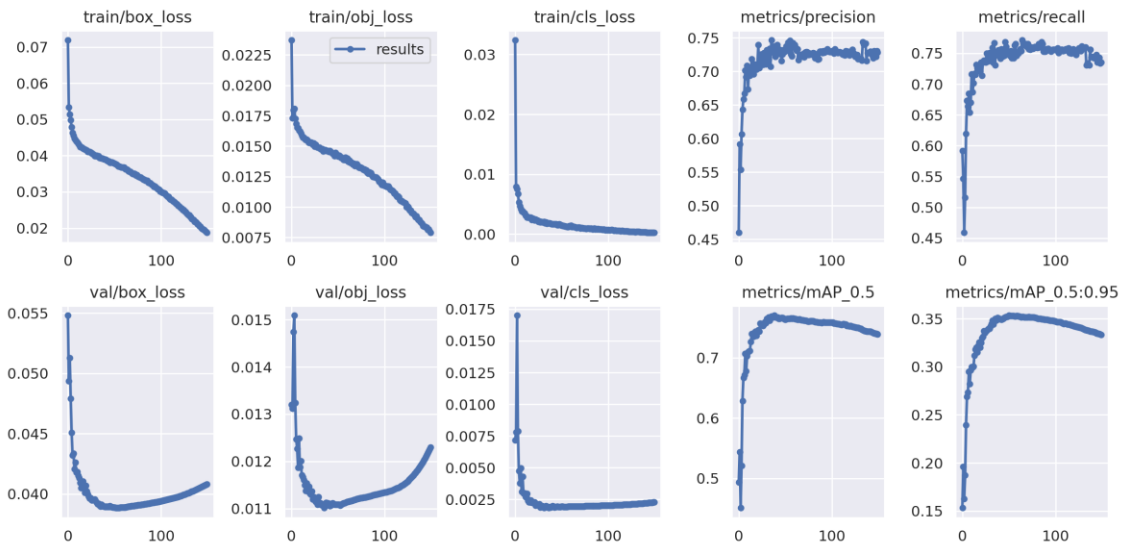


Figure 40. Fourth model training results.

Train settings were like the initial training. YOLOv5 training was done with 150 epochs (Figure 40). The batch size selected was 64. The best model was selected for production, not the last model after 150 epochs. Training result metrics improved notably compared to the initial model (Figure 31).

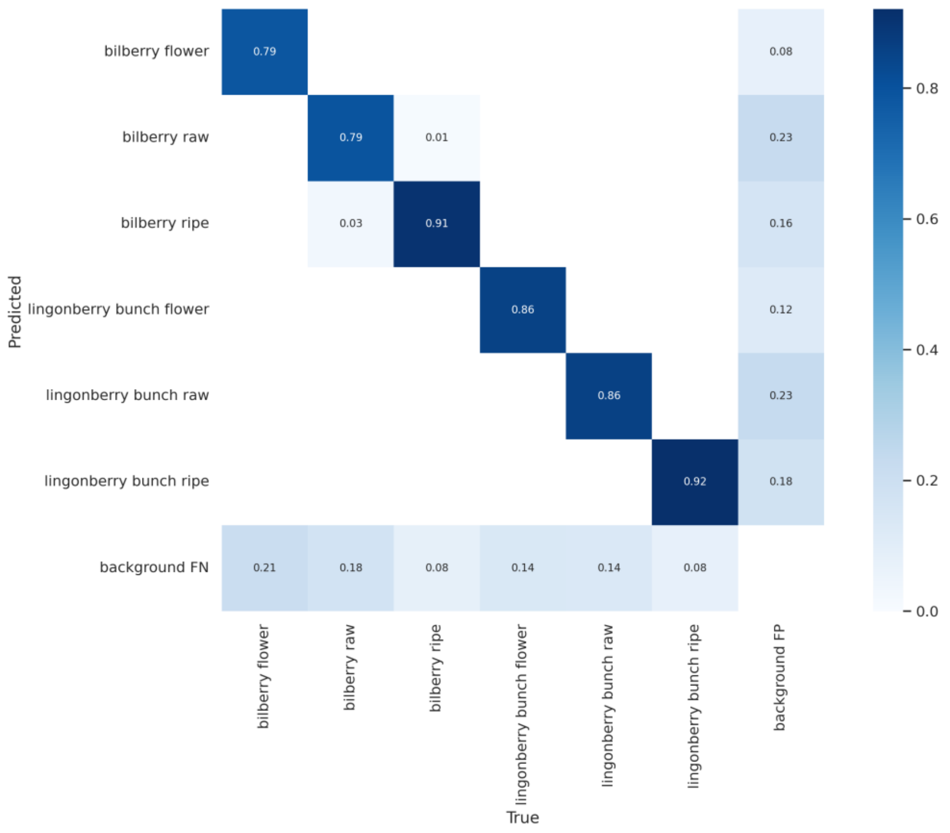


Figure 41. Six class confusion matrix.

The confusion matrix indicated notable improvements in training results compared to the initial model, as most problematic classes were removed from the dataset. In addition, training validation data results (Figure 40) with different benchmarking results (Figure 38) indicate the training dataset improved compared to the initial trained model.

The main issues remaining in the confusion matrix training results indicated that they were partly caused by the bilberry unbalance in the training dataset (Figure 41). Weak bilberry results can be caused by the significantly lower number of instances in training data. However, this does not explain raw bilberry results, as the number of raw bilberry instances was the highest in the dataset (Figure 39). Two other options for why bilberry flower and raw had weaker results than other classes were dataset diversity and class difficulty.

The Dataset diversity issue stems from the primary dataset selected for annotation. In the primary dataset, bilberry images had only one extraction location (Figure 13. The number of phase area images from different locations.). For all phases, images were collected only at a single time and in a single location. This bilberry dataset homology likely had an impact on results. For example, in the primary dataset, all raw bilberry images were homogenous in colour and feature because the data is a single point of the day and location data had single weather condition lighting.

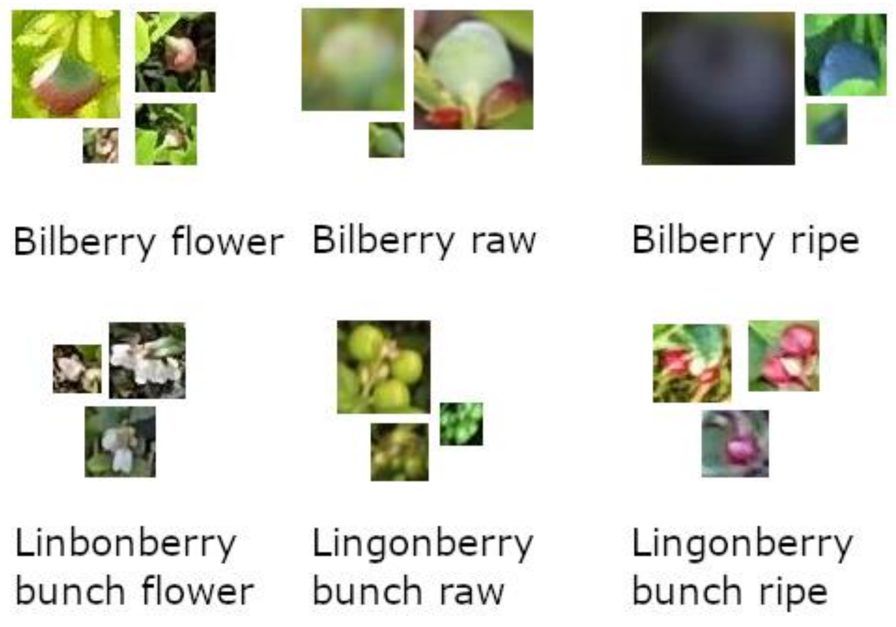


Figure 42. Example of annotated classes in primary dataset.

Class difficulty refers to visual differences between classes; some classes are more difficult to recognise than others. Initial dataset classes had difficulty annotating due to class visual ambiguity. For example, single lingonberry classes were difficult to define in annotation due to the low amount of detail. As single lingonberries were removed and merged with bunches, after likely most difficult class visually became especially bilberry raw and bilberry flower (Figure 42). Raw berry was difficult to detect for annotating personnel, but lingonberry raw as a bunch had enough detail to make it easier to detect than raw bilberry (Figure 42). The Bilberry flower was also quite difficult to detect when annotating. Class difficulty estimated based on visual inspection of the dataset indicated that bilberry flower, especially bilberry raw, was expected to have the lowest results. Ripe berries were easiest to detect when annotating. Class difficulty effects model results in two ways: Annotating had more errors due to detection difficulty, and data contained fewer features for the model to use for classification.

Regarding class difficulty, bilberry classes in the original dataset lagged in balance and variety. Adding more variety to the dataset and annotations can improve the fourth model results.

6 DENSITY ESTIMATION

After berry detection with deep learning was achieved with the accuracy described in the fourth model section, berry density estimation methods were tested. Four main density estimation methods were tested, and methods frame mean counterparts. Finally, all eight results were listed and compared in the results section.

6.1 About density estimation goals

Bilberry and lingonberry field measurements aimed to find berry yield in kg/ha. In the field, this was done by counting berries inside multiple sample plots of a 1m² area. The mean was calculated from multiple sample plots, and the berries/m² number was converted to kg/ha using the mean fresh weight of a single berry—0,35 g for bilberry and 0,23 for lingonberry. (Kilpeläinen 2016). In this thesis, conversion to yield was not done, and the target unit was kept as berries/m².

The number of berries in the photographed area was estimated by calculating the berries in an image. In the image, not all berries are visible as an unknown number of berries is lost behind leaves or other visual obstacles present in an image. Therefore, the presumption to study was that the number of berries visible in a certain area would correlate with the berry density estimation. Berries were counted from an image, the image area was estimated, and results were compared to the field measures to test this presumption.

Previous scientific literature confronts challenges of density estimation from a single image in estimating people density (Silveira Jacques 2010). With pixel level density estimation, the background was subtracted from detected objects and thus reaches the estimation value but skips the actual object count (Silveira Jacques 2010). The texture level approach estimates the number of people and density value. As an example of the texture level approach, Wu et al. propose to create image cells and a support vector machine (SVM) to solve the regression problem of estimating density (X. G. Wu 2006). A significant drawback of the described method is that density cells depend on camera parameters (Silveira Jacques 2010).

The thesis density estimation method underlining techniques can be divided into two groups. 1) Object-level density method based on detected berry objects. Object level density estimation tends to produce more accurate results than pixel or texture-based (Silveira Jacques 2010). 2) Using reference scale visible in the image. This reference scale was a counting frame with measured side lengths of 1 meter each.

In field measurements, density was estimated by counting berries inside a 1m² area marked with physical boundaries and frames. The deep learning model trained estimates the number of berries by searching for berries from images taken from the frame. Image area is estimated with various methods, which are compared below. Four methods are:

1. The label count in frame images was done via selecting images from the same general distance. The distance was estimated based on the physical frame being visible in the image. Assumed images were from the same distance and thus comparable to each other.
2. Label count inside a frame. Berries were selected by cropping labels inside the frame visible in the image and counting only berries selected inside the 1m² frame.
3. Label count after homography transform image. Homography transforms an image to represent the image as an orthographic projection image straight up. The transformation was done with four frame corner points. Homography transformation enabled estimating more labels than counting only inside the frame.
4. Estimate image area from detected berries bounding box size means. If detected berries were smaller, the image area was assumed to be higher.

Correlation value and linear regression were estimated between field measurement result berry density and from various area estimation methods. Linear regression corrected berries in surface/m² to the assumed berries/m². Methods were compared by evaluating correlations with field measurements.

The best result was a high correlation because detecting the actual berry amount was highly unlikely with one image. Using correlation to estimate density means that the actual berry count was unnecessary with lingonberry bunches. Instead, bunch count was assumed to correlate. The bunch count was directly compared with field measurements for correlation. In this thesis, lingonberry count meant bunch count.

The primary correlation improvement method was instead of one image from the frame, one could use the mean value from multiple images. Eight methods in the following chapters were based on the four methods mentioned above and their mean grouped counterparts. Single method correlation results were grouped by count frame, which was assumed to be the same frame across multiple images when field measurement value and location were the same.

6.2 Frame versus count number of berries

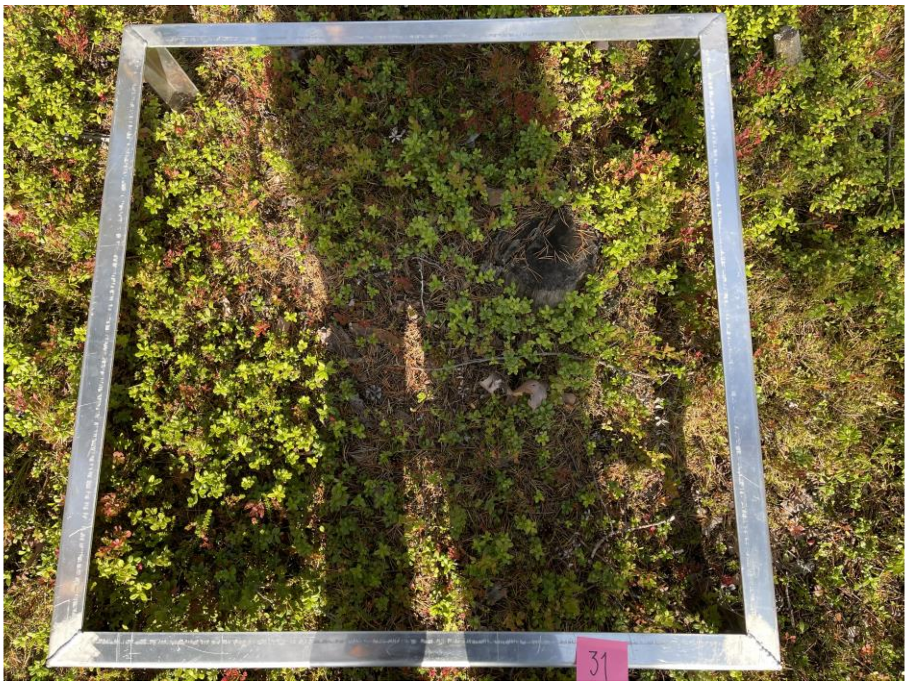


Figure 43. Frame image was defined as an image with a frame fully enclosed inside view.

First, simple image area estimation was done by comparing images' berry numbers with frames in near or full view (Figure 43). These images were assumed to cover an approximately similar area. This assumption most likely caused noise in correlation, but it was done due to simplicity and a set baseline. Data was also easily available as a table connecting frame image name to field measurement value (number of counted berries inside the frame). For simplicity, the frame image area was defined as one (1).

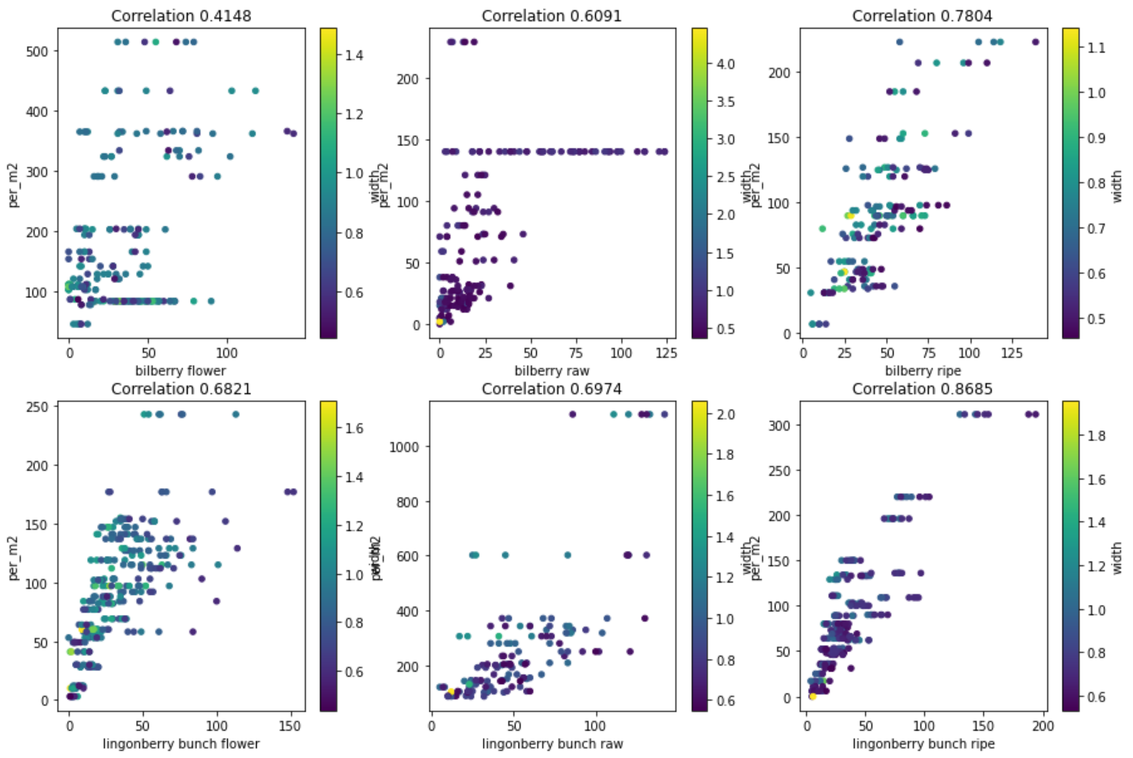


Figure 44. Scatter plot of frame values compared to same label berries found by deep learning in full frame images. Colouring represents image mean label percentage width from image width.

The predicted labels table was created by running the detection process on the primary dataset. Next, the amount of prediction in the image table was created from the labelled dataset, grouping all labels with identical image paths and unstacking detected label names into separate columns.

The count frame table dataset was merged with predicted labels using the image path. Field measurement value (per_m2) was compared to the number of detected berries in the image target label. The target label was the berry phase counted in the field measurement process, and the count value was written down. Image target label or count label was the column in the count frame dataset.

Pearson correlation was calculated for each class separately between columns count label value and field measurement value (per_m2) values in Figure 44. Each dot in Figure 44 represented a single image. Dot colouring was the mean detected label width in the percentage of the image width. Colouring was an additional visual image area estimation inspection. As images from assumed similar distances should have approximately similar mean label width values.

Several interpretations was drawn from the resulting scatter plots (Figure 44) and corresponding Pearson correlations per class.

- Horizontal lines forming from dots were caused by images taken from the same physical frame with the same count values from field measurement. The deep learning model detected different amounts for each image.
- The colour of the mean label width indicated that images with larger labels have fewer berries found by the model. Again, the result was expected, as close-up images, in theory, should have a smaller area and thus fewer berries.
- Colours of mean label width were like each other, suggesting that label size could be used to estimate the image area.
- A large vertical line in bilberry raw with near 150 counted berries by field measurement was due to dataset unbalance. All Ounasvaara images with the frame visible were from the same single-count frame.
- Correlation existed. Better area estimation and better deep learning model results could improve the correlation. However, model deficiencies were likely to cause weaker correlations in bilberry flower, and raw results as the fourth model results were weakest with the classes in question.

An assumption was made from the first scatter plots (Figure 44) that using multiple images' mean berry number values could improve the correlation. Two columns grouped images—location and field measurement value (per_m2). Location grouping was used to minimise images grouped with identical field measurement values but not taken from the same frame. Image group mean row label values were used.

6.3 Mean frame versus count number of berries

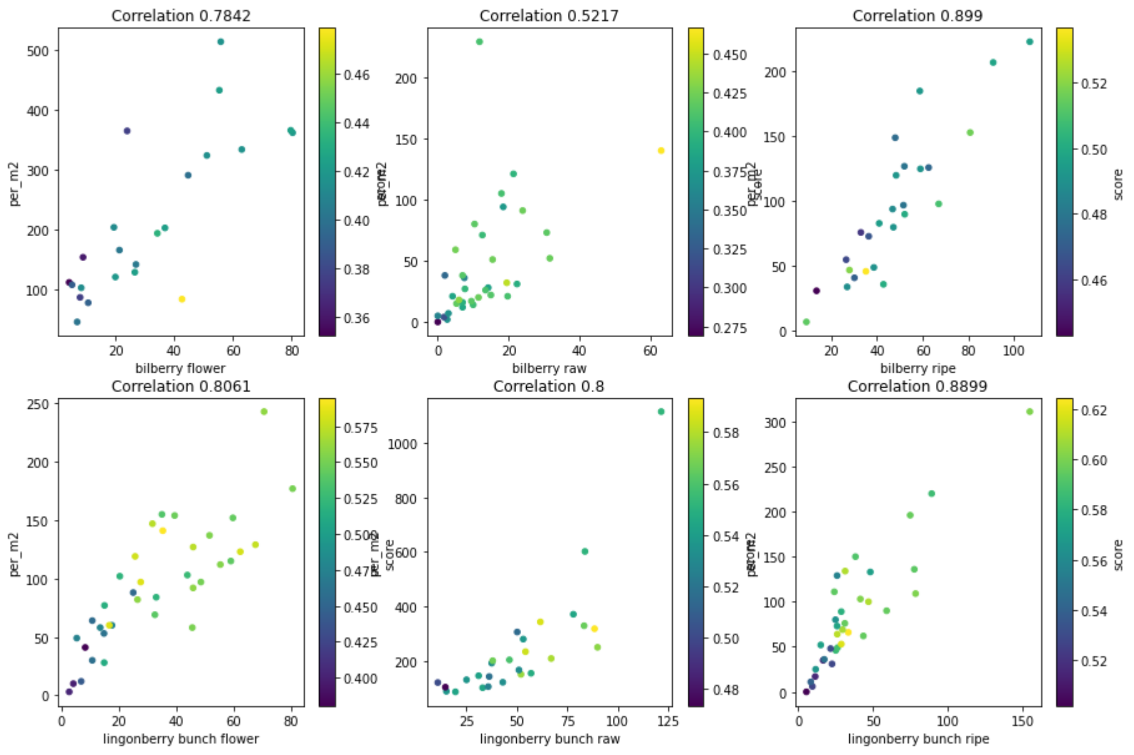


Figure 45. Label scatter plots grouped images row compared taken from same count frame. Colouring represents the mean model score in the image group.

Mean berry count Pearson correlation with field measurements mainly improved, except with bilberry raw (Figure 45). Improvements indicated that yield estimation improves using the mean value from multiple images.

Label	Frame images	Grouped frame images
lingonberry bunch raw	137	27
bilberry ripe	157	26
bilberry raw	180	35
bilberry flower	216	23
lingonberry bunch ripe	285	34
lingonberry bunch flower	299	37

Table 7. Amount of frame data in each class as a single image or grouped by location and berry frame count value.

Table 7 illustrated the number of frame images. Grouping caused significantly reduced data for correlation comparison. The limited number of grouped images was because the primary dataset was not designed to be grouped. The dataset was mainly training data for object detection. Field crews were not instructed to take many images from a single frame.

6.4 Frame versus frame cropped berries

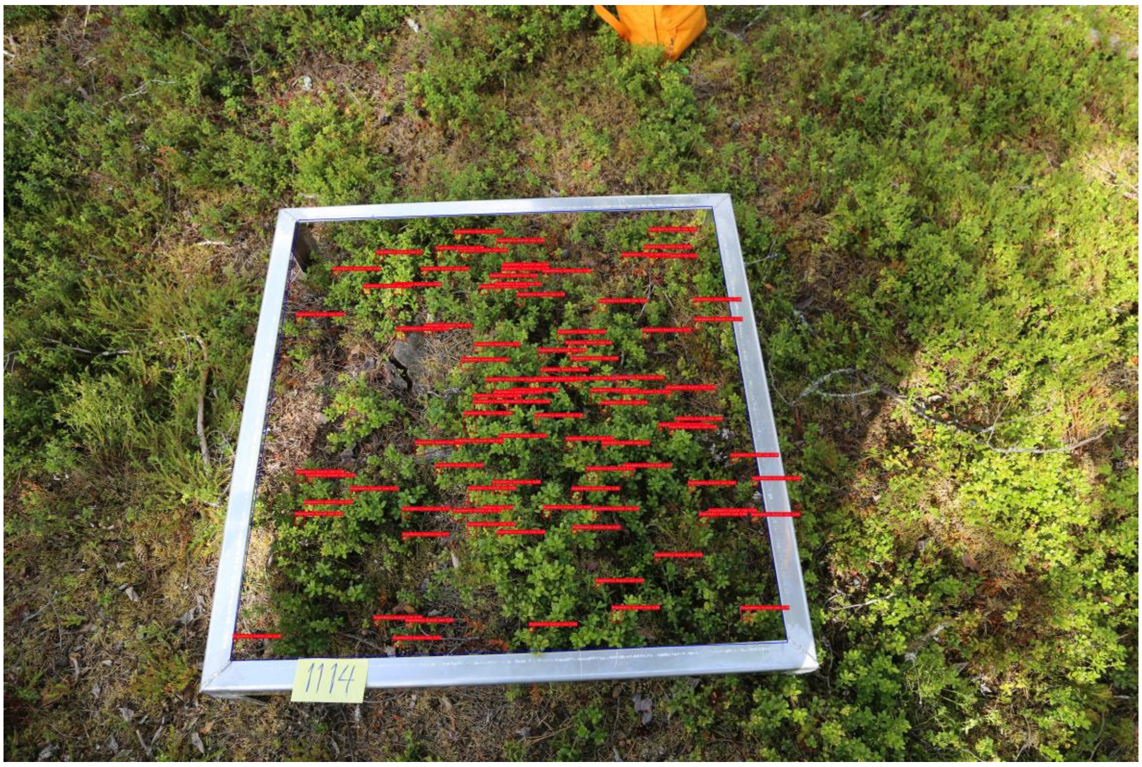


Figure 46. Example of frame area cropped labels.

The polygon area of the frame dataset was used to crop detected labels inside each image, selecting only labels inside the frame (Figure 46). The cropped image has only inside frame labels; thus, all labels were from a $1m^2$ area. Cropping produced accurate area-to-berry number relation, but two issues remained. First, the example figure demonstrated how a large image area was not utilised when cropping. Secondly, cropping requires the frame to be entirely in the image. Using only images with the entire frame lowered available images for scatter plot analysis. In addition, using entire frame images raised a new issue. The distance required for a full-frame image is so high that visually inspecting is likely too far for the computer to detect all surface berries that would be detected in more detailed images. Finding the best distance to take an image requires a deeper study. With a dataset of different distance images with different size frames, one could determine the optimal distance with current consumer-level mobile cameras.

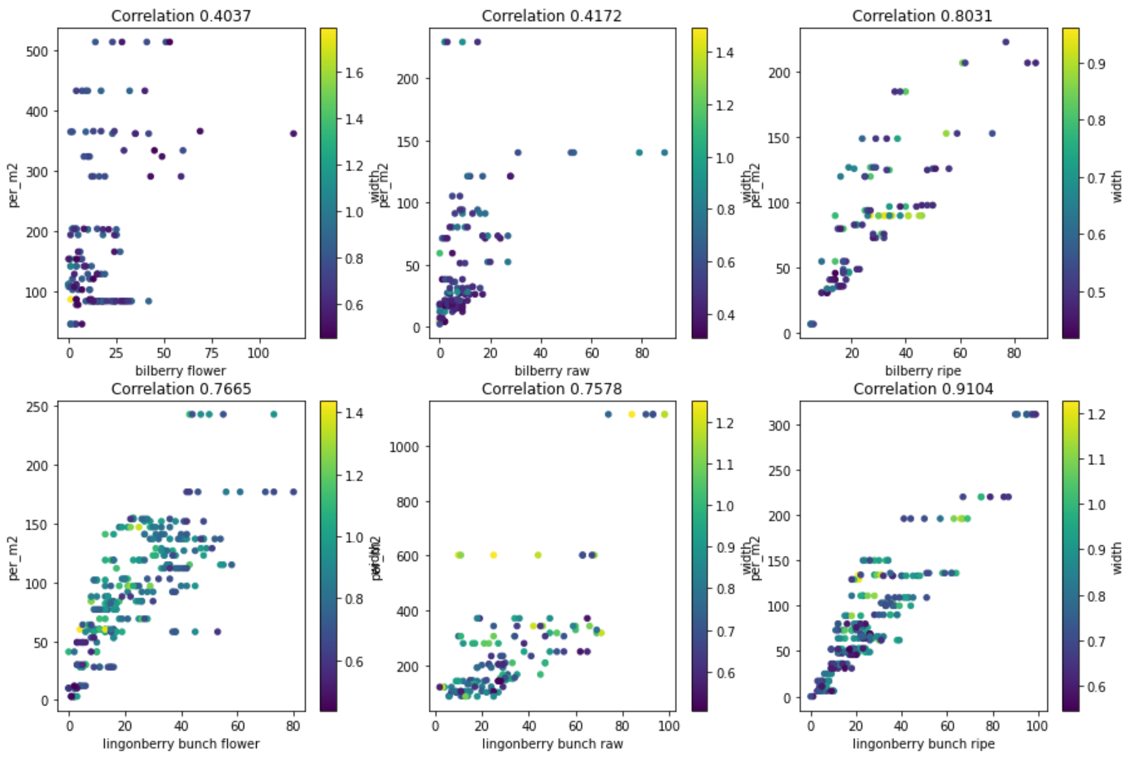


Figure 47. Scatter plot of frame values compared to frame cropped label berries found by deep learning. Colouring represents image mean label percentage width from image width.

Pearson correlation with single image data pointed improved with crop compared to label count in frame images. On the other hand, in two of the worst correlations, bilberry flower and bilberry raw, correlation results decreased.

6.5 Mean frame versus frame cropped berries

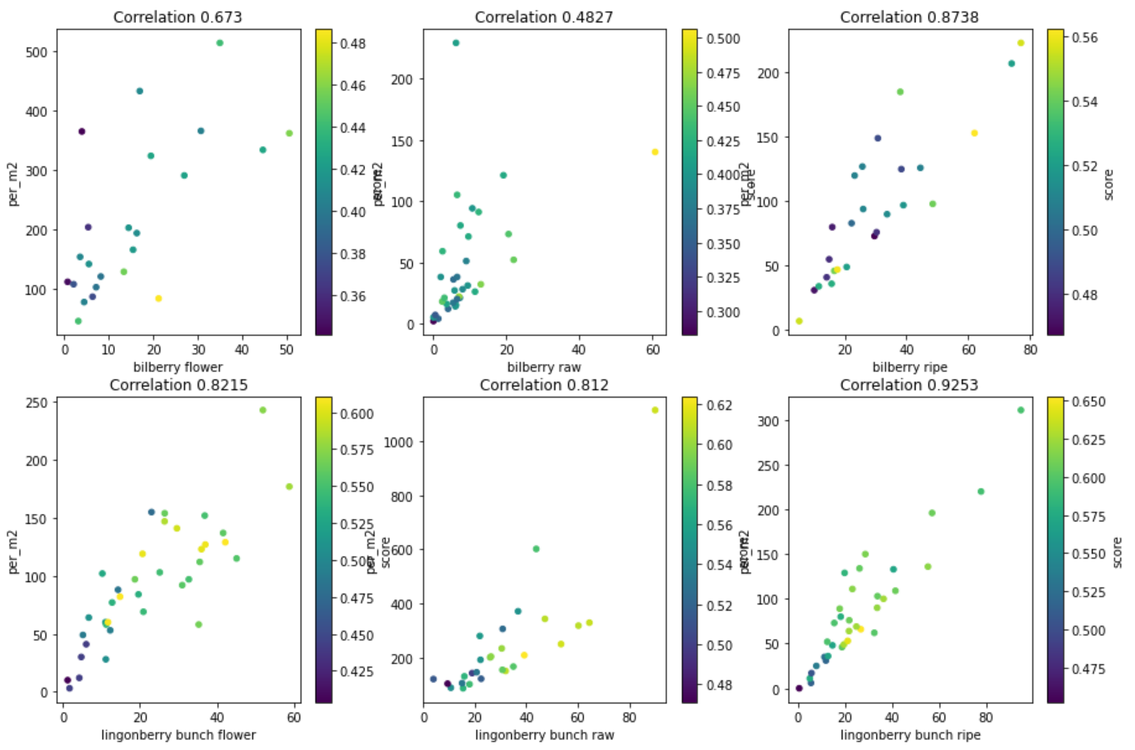


Figure 48. Label scatter plots grouped images row compared taken from same count frame. Colouring represents the mean model score in the image group.

Images grouped by location and field measurement count value Pearson correlation results changed from single images, similar to the label count in frame images (Figure 48). Overall the correlation results were higher than the grouped label count in frame images.

Label	Frame images	Grouped frame images
bilberry ripe	105	26
bilberry raw	119	34
lingonberry bunch raw	134	27
bilberry flower	142	23
lingonberry bunch ripe	248	34
lingonberry bunch flower	261	37

Table 8. Amount of frame data in each class as a single image or grouped by location and berry frame count value.

The number of frame images for analysis dropped slightly (Table 8) as the annotation process of drawing required the whole frame to be visible in the image. Frame images in the simple label count method included images with frames partially visibly. Annotated frames with polygon required the whole polygon to be visible.

6.6 Frame versus homography two m² berry count

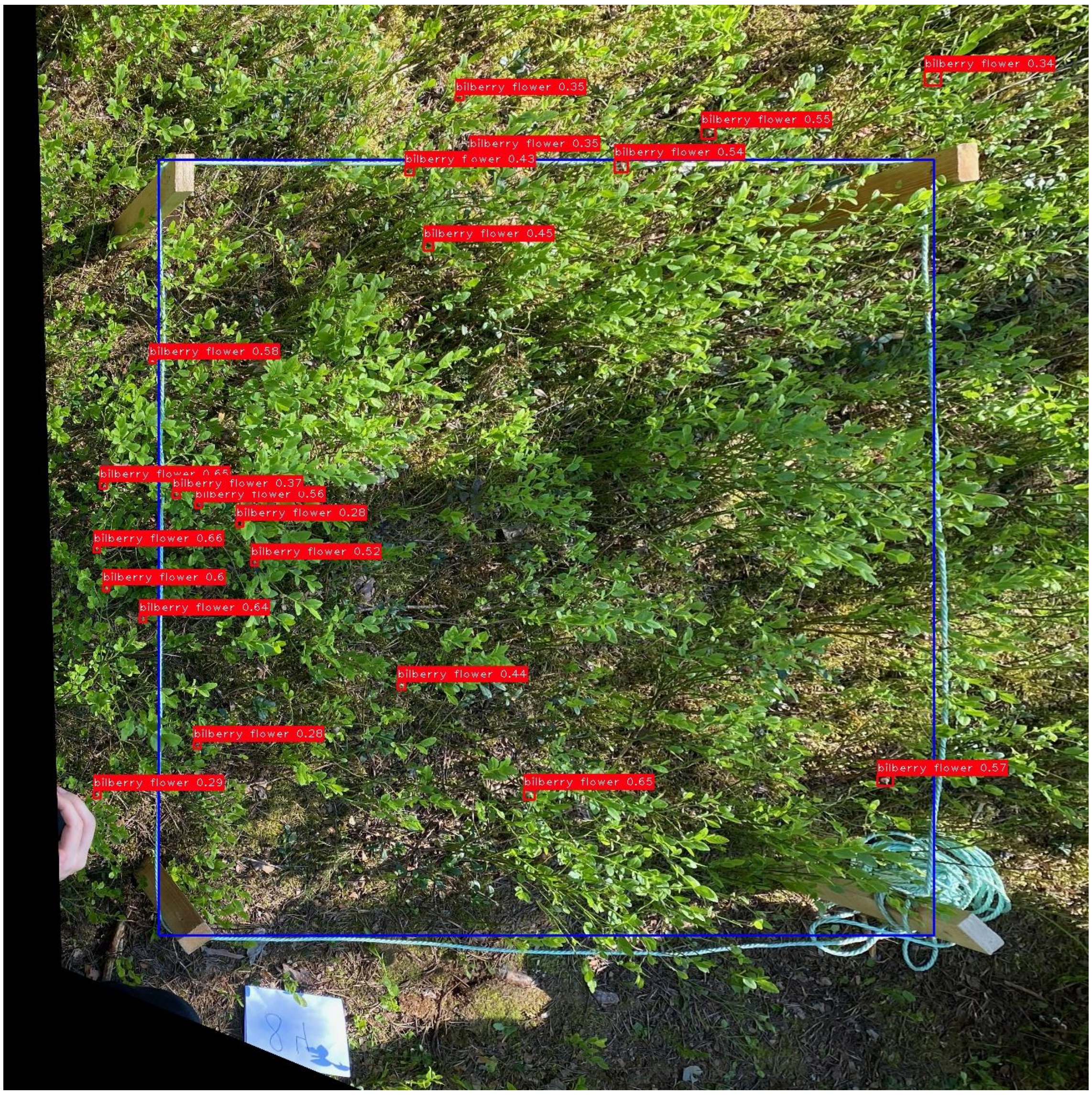


Figure 49. Homography converted images with labels in homography image coordinates.

Homography perspective correction was done to frame images with a frame annotated as a polygon with four corner points. Homography conversion requires a minimum of four points (Malis 2007). Homography was done to enable a larger area of the original image to be used for density analysis than simply cropping labels with a frame polygon. The source image was a frame image with polygon annotation. Anchor points were the four corners of the frame polygon. The target image was created from an empty image with dimensions 1400x1400 pixels. Frame anchor points in the target image form a square polygon in the centre of the image with dimensions 1000x1000 pixels. The target image was a frame image corrected to show the frame as a square in the middle of the image with 200 pixels padding in each direction (Figure 49). Frame side length is one (1) meter, and the image was translated to 1000 pixels in the target. With padding of 200 pixels in all directions, the target image represented an area of 1,96 m² with 1,4m (meter) sides.

A homography correction matrix transformed image labels into target image coordinates. The original image likely extended over the 1,96 m². Thus, some number of original image cover area was lost. Labels left outside the image were discarded. Sometimes, the original image's visible area did not cover the entire target image area. This area can see as black (Figure 49). The Black area left in the target image area was calculated and removed from the total area available for density analysis. The benefit of homography as to have nearly double the area for density analysis compared to simple label cropping with a frame.

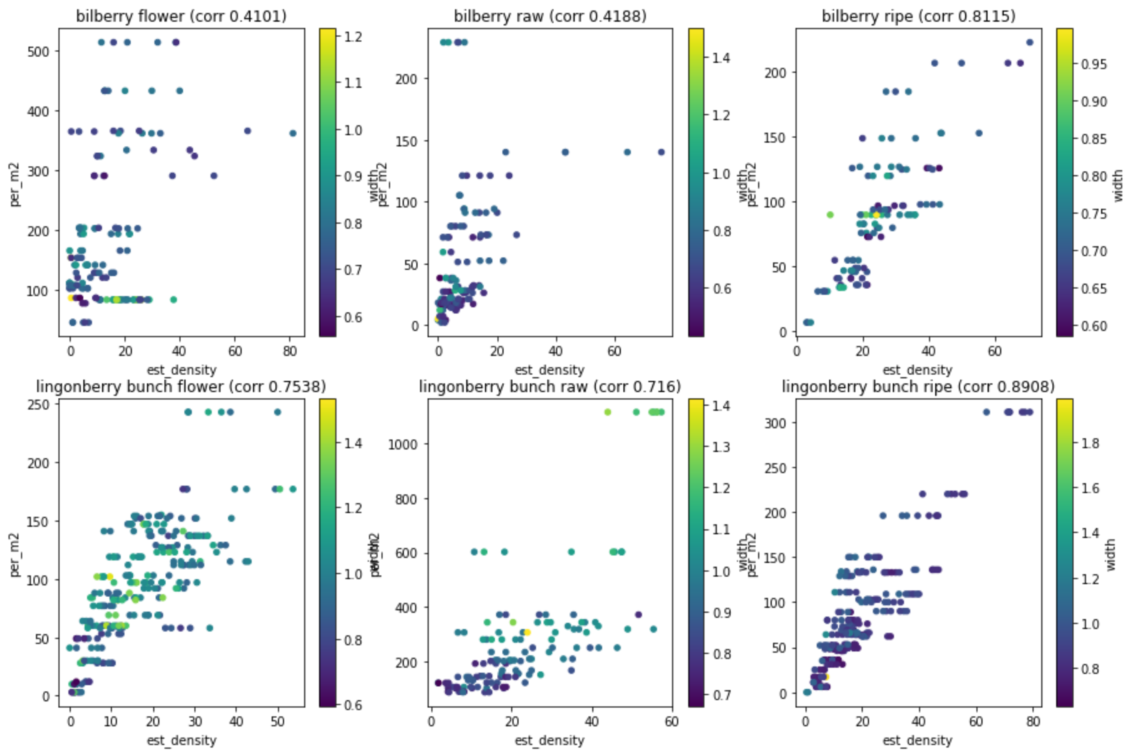


Figure 50. Scatter plot of frame values compared to labels in homography converted frame images by deep learning. Colouring represents image mean label percentage width from image width.

Pearson correlation results per class wer like the frame cropped dataset (Figure 50). Correlation values were not significantly improved.

6.7 Mean frame versus homography two m² berry count

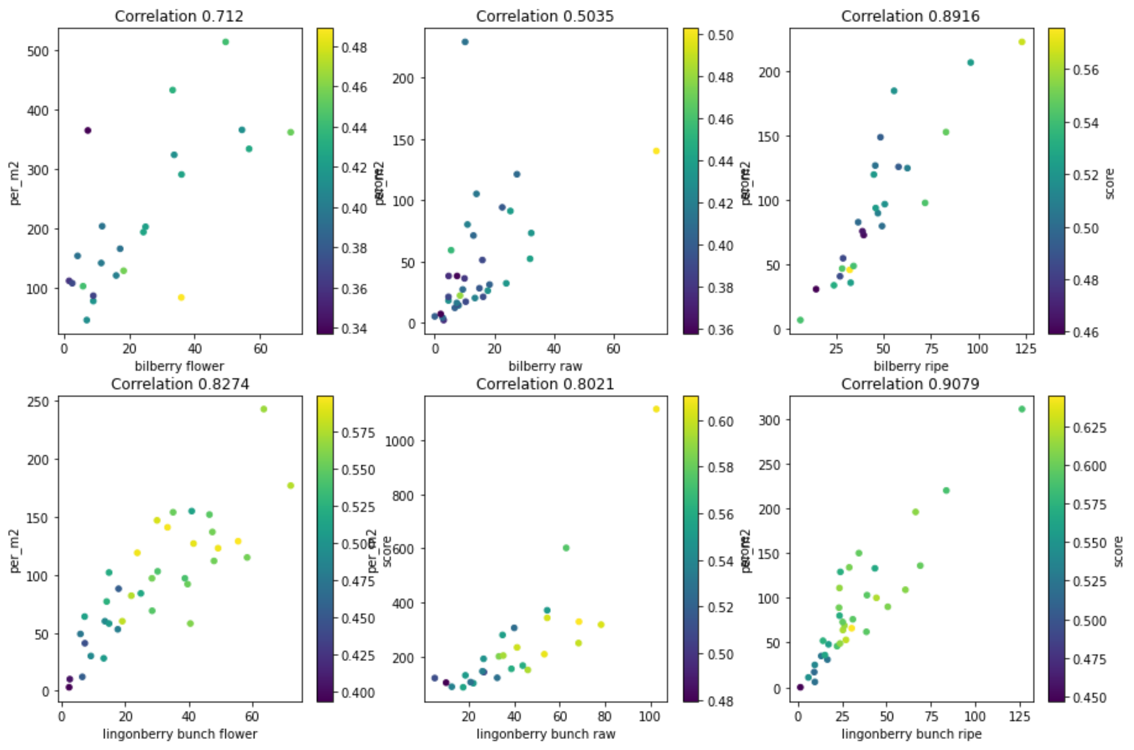


Figure 51. Label scatter plots grouped images row compared from the same homography corrected images. Colouring represents the mean model score in the image group.

Mean results from grouped images by frame with homography correlation (Figure 51) improved correlation compared to frame images separately like in all previous area estimation methods. However, compared to cropping, homography did not significantly improve with current data.

6.8 Frame versus label size estimated berry count

Estimating image area with image mean label size (detected berry size) emerged as an option when frame images were filtered from the dataset. Frame images mean label side width or height affected to be 1% of parent image width or height. Thus mean label size was assumed to be proportionate to the parent image area. The opportunity of estimating an image with label size was to estimate berry density without a frame in the image. Without a visible frame, estimating berry density could be done anywhere.

The base dataset should include known berry densities from images taken from varying distances to create a density estimation method that does not depend on frame reference. The target result from the analysis for a base for the prototype was linear regression formula to estimate the area based on detected labels. This linear regression was devised from a scatter plot, comparing the estimated area based on label sizes with field measurements. Therefore, frame value was needed for comparison. Frame annotation dataset contained frame images and images without visible frames. Images without a visible frame were assessed to be taken from the inside known frame. In theory, these images pose the same berry density per m² as the corresponding frame. Frame dataset contained 1274 images with visible frames and 949 images with known berry density values but no frame in the image.

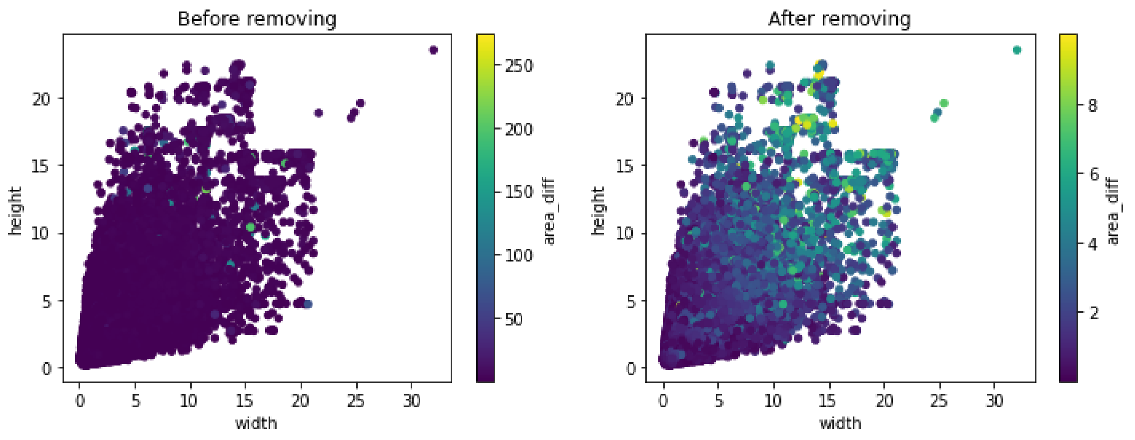


Figure 52. Main dataset prediction label height and width scatter plot—values as a percentage of corresponding image width or image height. The first plot is before removing labels with a 10 times larger area than other labels in the same image. Second plot after removing.

Predicted labels in the dataset varied in size due to multiple reasons. Image angle causes closer labels to be larger than labels further away, and different berry species or phases may differ in size. With image area estimation with mean label size, a single mislabel with a significantly larger label size can negatively affect image area estimation results. Large mislabels were observed in the dataset. Single mislabel has a minor effect on the total of berries counted, but label size estimation may be significant. Data cleaning from significantly different labels with significantly larger label sizes than the rest of the labels in the same image:

1. Calculate the label mean width and height of each image.
2. Calculate the area on each label.

3. Calculate the mean label area in the image. Mean prediction width multiplied with mean prediction height.
4. Calculate the difference in the area (area_diff) as label area divided by the mean label area in the current image
5. Remove labels with an area difference value of more than ten (10). Therefore, labels with a 10 times larger area than the mean area in the current image are removed (Figure 52).

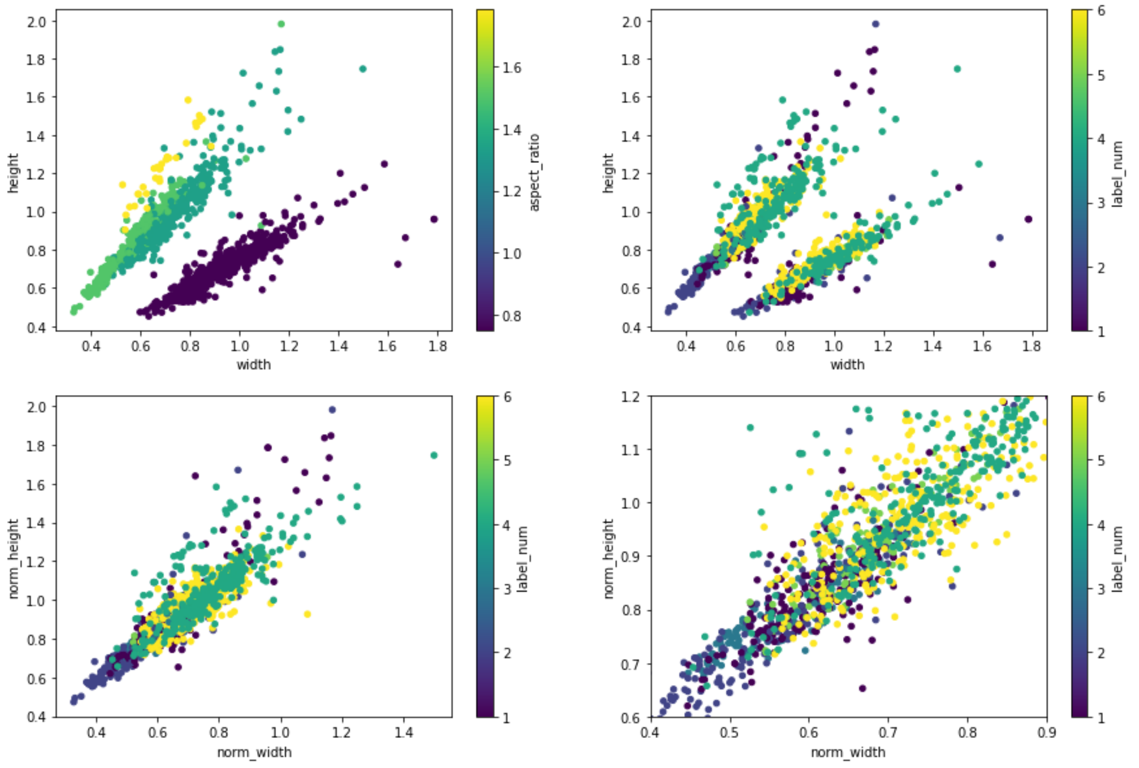


Figure 53. Label width and height percentage from frame image width and height using a dataset of only images with frame in view. Above label sizes before aspect ratio normalisation and below with normalisation.

For initial viability, the dataset of images with the frame visible was assumed to be from a similar distance. Plotting image mean label width and height in scatter plot revealed two distinct clusters (Figure 53, top row). The scatterplot colour (Figure 53, top right) was the image aspect ratio calculated by dividing image width by image height. Figure 53 top left colouring was the device number. Colouring was to check if the device was the reason for the two clusters, as device camera differences can affect the label size in the corresponding image.

Image width and height were normalised by swapping width and height values in images with portrait aspect ratio to mitigate noise caused by aspect ratio. Removing aspect ratio derived difference results in a single cluster in the scatter plot (Figure 53, bottom row). In figure 51, the bottom right was a close-up of the bottom left scatter plot with different colours for each berry label. Colouring likely indicates that label type affected label size and thus affected the estimated image area from the label size. Field measurement environment differences between berry phase and species impacted cluster formation.

Images with visible frames were normally distributed around one (1) m². So practically image area included a larger area than just the frame. This gap is corrected with the final linear regression. As a consequence, the error in area value was systematic. Thus linear regression from surface detected berries to total area berry density will also account for this error.

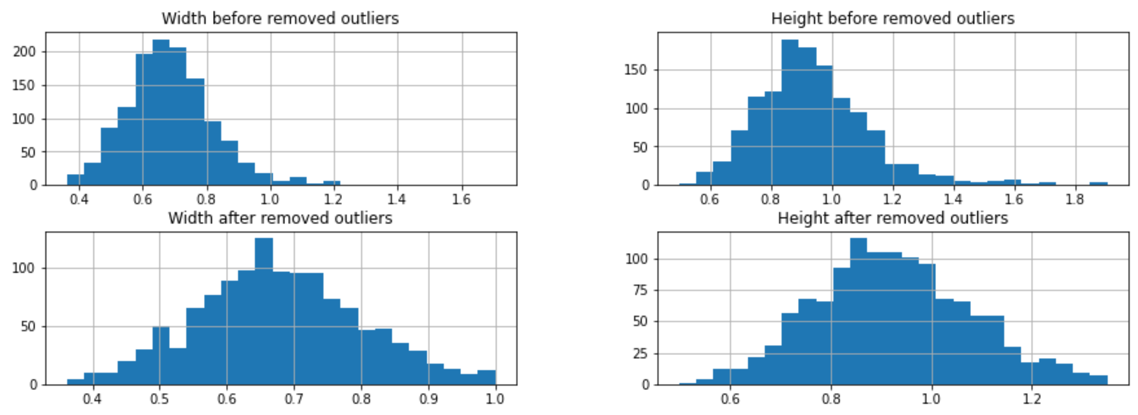


Figure 54. Outliers of label width and height were removed from the frame dataset.

The label means width and height per frame image were normally distributed with outliers (Figure 54, top row). Outlier images were removed from the dataset such that normally distributed dataset peaks remain in the centre (Figure 54, bottom row). One finding was that in frame annotation, 0.68% width and 0.9% height from image size meant that it was most likely an average frame image. Therefore image with a mean of 0.68% width and 0.9% height label size indicated that the image is the mean frame image. We assume that the frame image had a 1m² surface area.

	Bilberry	Lingonberry
Width	0,60	0,74
Weight	0,82	1,01

Table 9. Berry species coefficient values for width and height.

As label type likely affected image area estimation due to species class size differences, different coefficient values were made for each species: lingonberry and bilberry. The coefficient value was the total mean estimated width or height for each species from the frame dataset with removed outliers (Table 9).

$$estimated\ width = \frac{species\ width\ coefficient}{image\ mean\ label\ width\ \%}$$

Equation 2. Calculating estimated width with species width coefficient and image mean label width

$$estimated\ height = \frac{species\ height\ coefficient}{image\ mean\ label\ height\ \ \%}$$

Equation 3. Calculating estimated height with species height coefficient and image mean label height

$$estimated\ image\ area = estimated\ width * estimated\ height$$

Equation 4. Calculating estimated image area

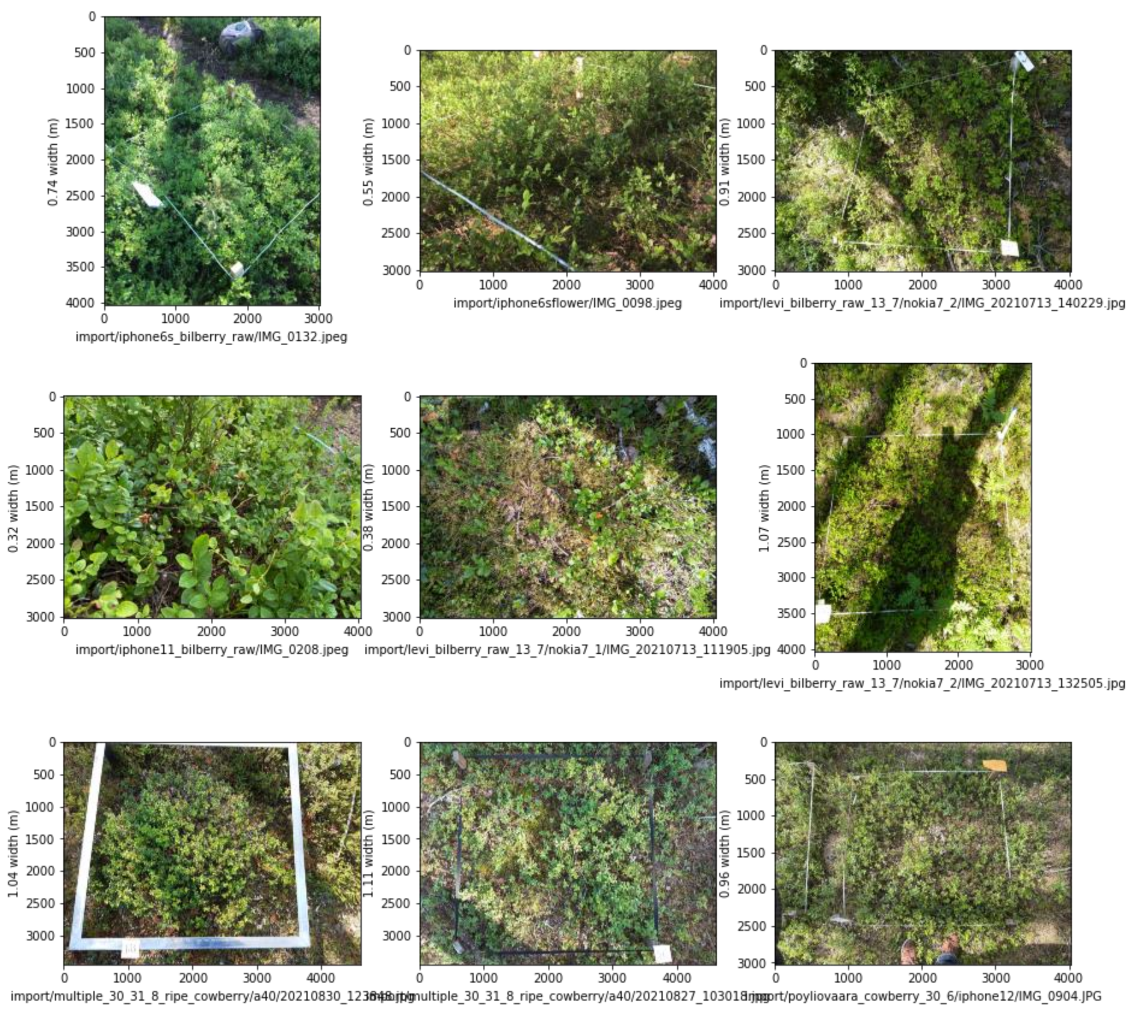

0.74 width (m)
import/iphone6s_bilberry_raw/IMG_0132.jpeg
0.55 width (m)
import/iphone6sflower/IMG_0098.jpeg
0.91 width (m)
import/levi_bilberry_raw_13_7/nokia7_2/IMG_20210713_140229.jpg
0.32 width (m)
import/iphone11_bilberry_raw/IMG_0208.jpeg
0.38 width (m)
import/levi_bilberry_raw_13_7/nokia7_1/IMG_20210713_111905.jpg
1.07 width (m)
import/levi_bilberry_raw_13_7/nokia7_2/IMG_20210713_132505.jpg
1.04 width (m)
import/multiple_30_31_8_ripe_cowberry/a40/20210830_123648.jpg
1.11 width (m)
import/multiple_30_31_8_ripe_cowberry/a40/20210827_103018.jpg
0.96 width (m)
import/poyliovaara_cowberry_30_6/iphone12/IMG_0904.JPG

Figure 55. Example image widths are estimated from the mean label size.

Equations 2, 3 and 4 were used to calculate the estimated width, height and area of the dataset containing all images with known berry density values from field measurement. Visual inspection of results indicated close-up images having smaller estimated widths than images with the larger assumed area (Figure 55).

Label	Frame images	Grouped frame images
lingonberry bunch raw	149	27
bilberry ripe	169	26
bilberry raw	217	35
bilberry flower	219	23
lingonberry bunch ripe	311	34
lingonberry bunch flower	336	37

Table 10. Amount of data with known berry density in each class as a single image or grouped by location and berry frame count value.

The dataset for Pearson correlation was larger than in other methods as it can now include images without a visible frame but with a known berry density value (Table 10). However, the dataset size was only slightly increased.

$$estimated\ density = \frac{number\ of\ most\ common\ labels\ detected}{estimated\ image\ area}$$

Equation 5. Calculate estimated berry density.

With several labels detected and estimated image area, berry density with visible berries was counted with Equation 5. Calculate estimated berry density. The estimated density was calculated for the dataset.

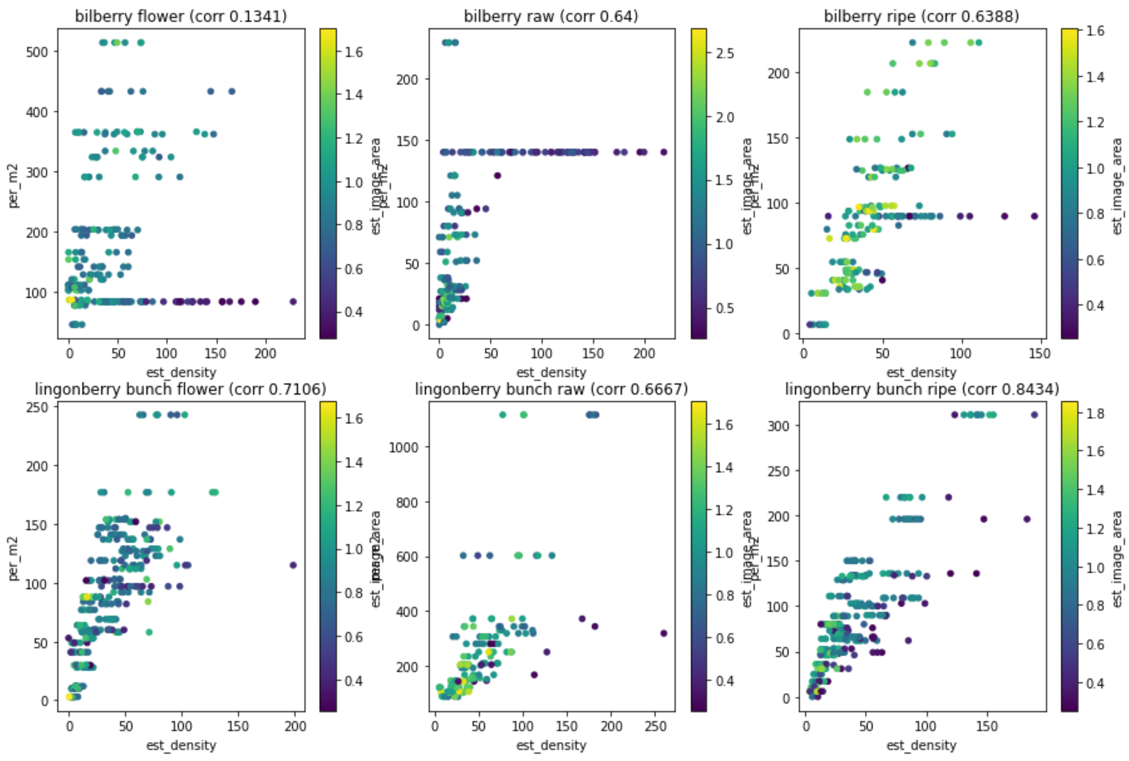


Figure 56. Scatter plot of images compared to estimated density with label size. Colouring represents the estimated image area.

Scatter plot of estimated image area based on label size compared to field measurements (Figure 56). The dataset contains close-up images without visible frame and frame images with known berry density values. Noticeable vertical lines in bilberry scatter plots were Ounasvaara single frame images. With a single location having a single frame only, theoretically, all images taken around the frame, close or far, in the same phase have the same berry density. This difference in the field measurement process produced more data from Ounasvaara than any other place.

The scatter plot's colour was the estimated image area in the image (Figure 56). The Scatter plot did not show significant clustering with the estimated image area.

6.9 Mean frame versus label size estimated berry count

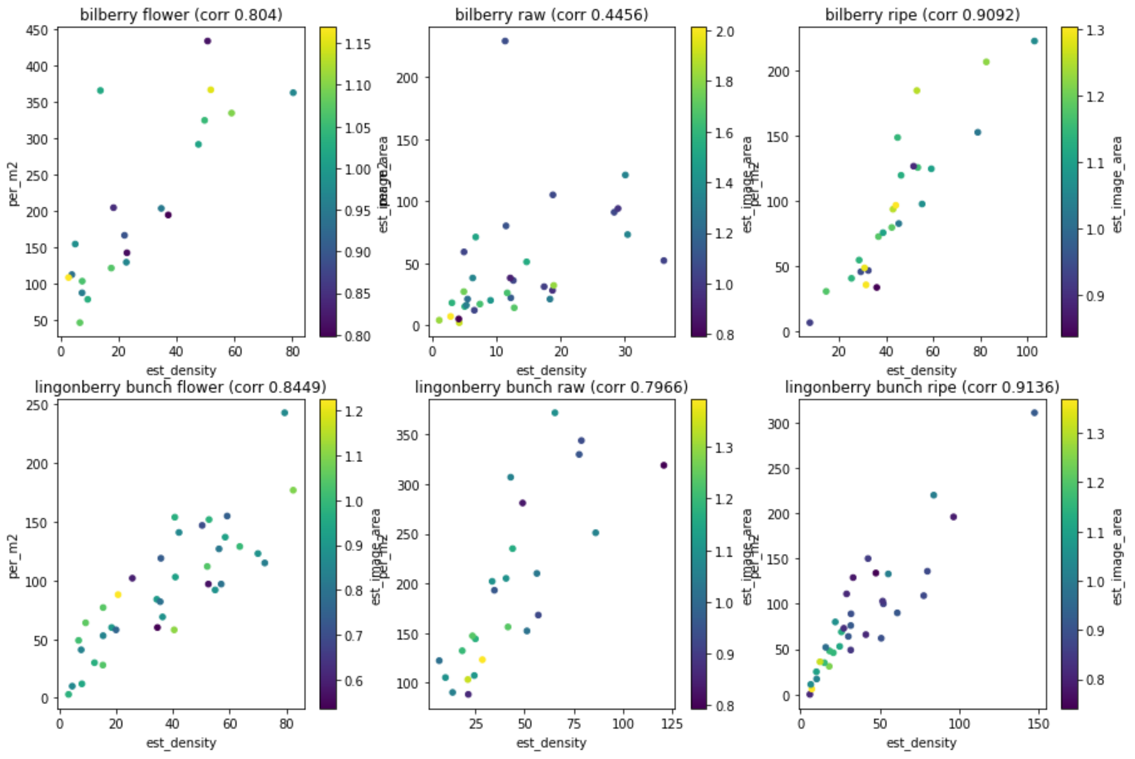


Figure 57. Scatter plots grouped images row compared to estimated density with label size. Colouring represents the estimated mean image area in the group.

The correlation of images was grouped with field measurement density value, and the location was shown in Figure 57. Pearson correlation values were similar to the previous area methods, but unlike previous methods, no frame data was needed to estimate density after species coefficient values were estimated. This freedom from visible frame makes the method of estimating density with berry size usable, for example, in prototype density estimation applications.

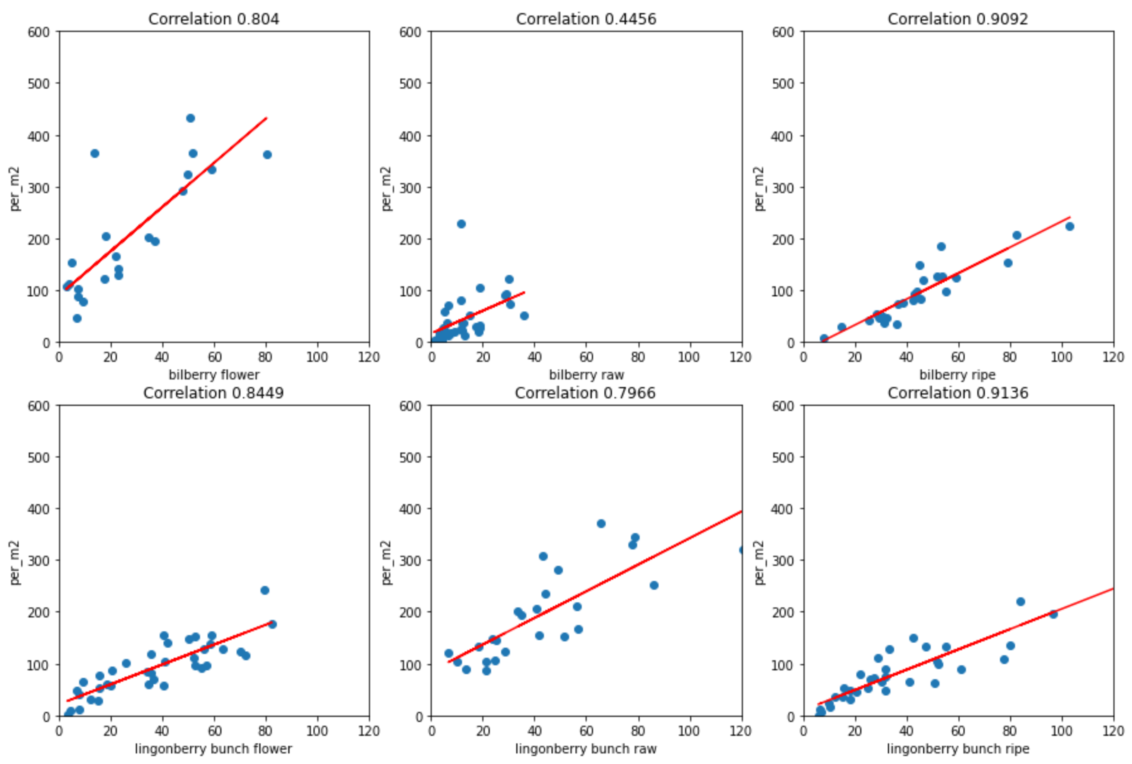


Figure 58. Linear regression for each class from grouped images.

A scatter plot created a linear regression model for each class separately (Figure 58). Intercept and slope -values was extracted from the regression model.

Class	Intercept	Slope
bilberry flower	89,5788	3,461025
bilberry raw	27,86355	1,510091
bilberry ripe	-12,7022	2,445188
lingonberry bunch flower	23,78982	1,81838
lingonberry bunch raw	-10,6856	4,946505
lingonberry bunch ripe	9,616994	1,927011

Table 11. Linear regression slope and intercept values per class.

$$regression\ adjusted\ estimated\ density \\ = estimated\ density * slope + intercept$$

Equation 6. Regression adjusted estimated density calculation.

With slope and intercept values (Table 11), the regression model adjusted estimated density was calculated with Equation 6. Regression adjusted estimated density calculation. The first estimated image area was calculated with the mean label size from the most common berry class. After estimating density, regression adjustment was done with slope and intercept.

image	**per_ m2**	**label_ sum**	**est_image _area**	**est_de nsity**	**est_density_ to_reg**
import/poyliovaara_cowberry_30_6/ slr/IMG_0427.JPG	88	26	1,6151	16,098	53,0622

Table 12. Example image with calculated estimation values.

In the example image, the label calculated was lingonberry bunch flower. The field measurement density value is 88 berries per m². The estimated image area from lingonberry bunch label sizes was 1,6 m². The estimated density was the label sum divided by the estimated image area. The resulting estimated density with only detected berries was 16 berries per m². After linear regression adjustment with lingonberry bunch flower slope and intersect values, regression adjusted berry density was estimated as 53 berries per m² (Table 12).

7 RESULTS

Different Pearson correlation results were compared for each density estimation method. Density estimation was done separately for a single image and multiple images from the same location. Eight different results were compared. Four single image correlations and four grouped by location mean correlations. The eight methods were (See chapter 6.1):

1. The label count in frame images. Single image density correlation.
2. The label count in frame images. Mean density in frame location correlation.
3. Label count inside a frame. Single image density correlation.
4. Label count inside a frame. Mean density in frame location correlation.
5. Label count after homography transform image. Single image density correlation.
6. Label count after homography transform image. Mean density in frame location correlation.
7. Estimate image area from detected berries bounding box size means. Single image density correlation.
8. Estimate image area from detected berries bounding box size means. Mean density in frame location correlation.

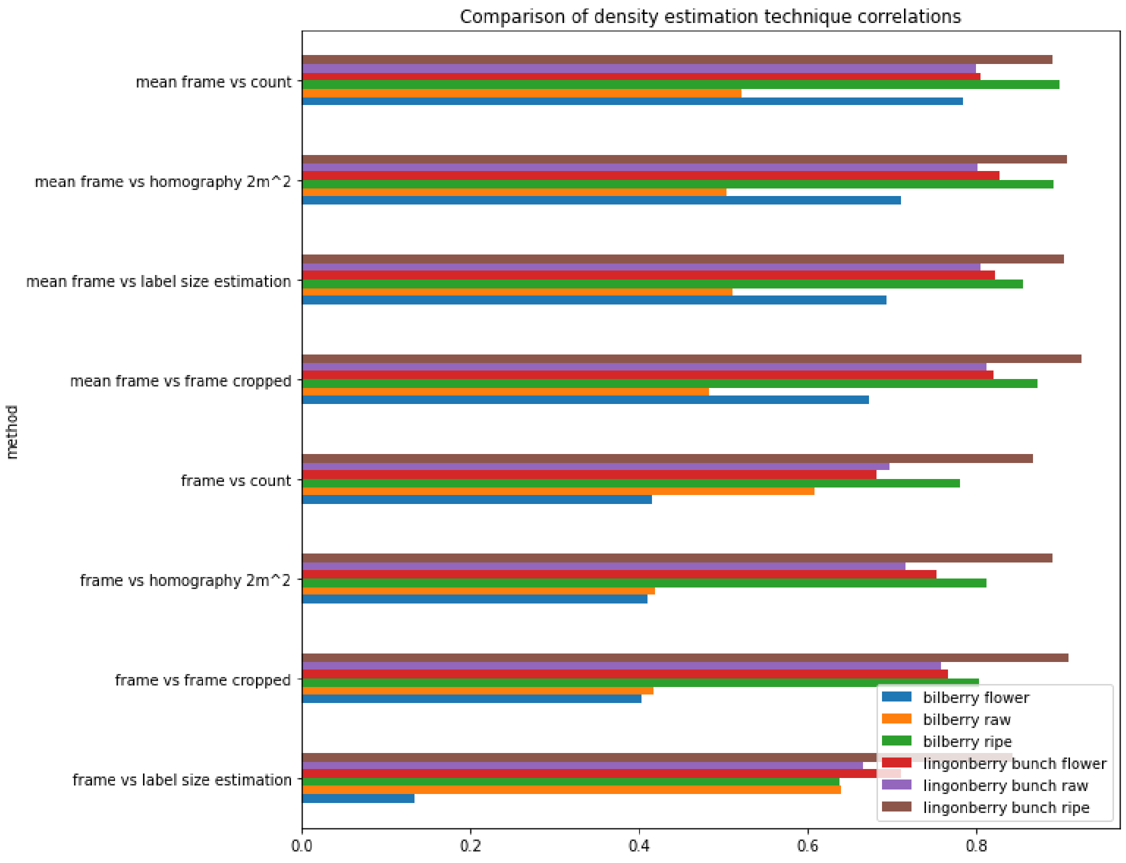


Figure 59. Comparison of density estimation technique correlations by class.

Results for each class in different methods can be seen in Figure 59. Figure 59 shows Pearson correlation values for each method and label. Weak results were presented in the classes bilberry raw and bilberry flower. Especially bilberry flower results were significantly weak with less than 0,2 correlation with label size density estimation without image mean. Bilberry raw results were erratic as in other classes mean from multiple images improved correlation, but not with bilberry raw. Lingonberry bunch ripe class correlations were high with or without mean. Mean density from multiple images impacted a positive correlation, especially when the dataset had a weak correlation with a single image.

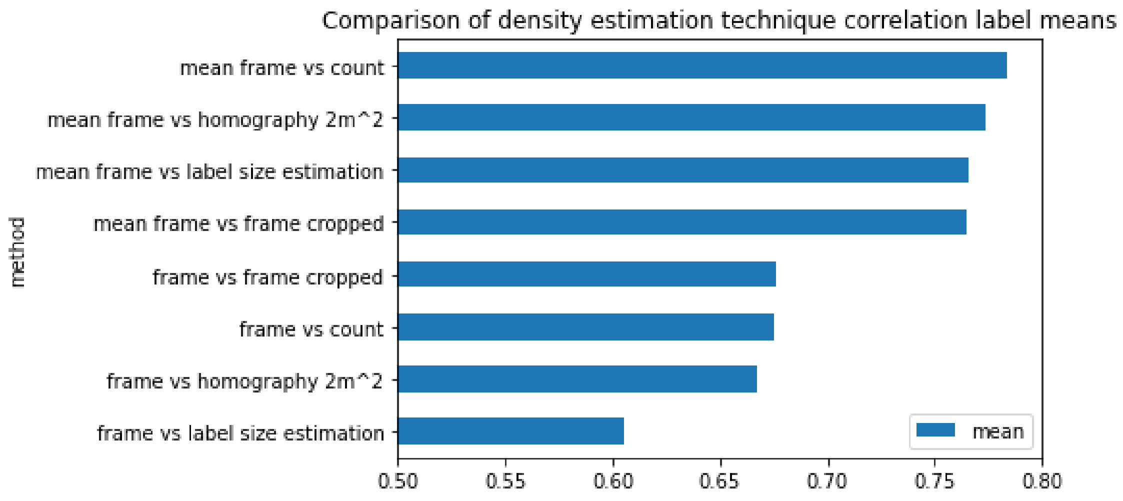


Figure 60. Comparison of density estimation technique correlation by mean of all classes.

All classes mean correlation graph per method in Figure 60. The main observation was that mean density from multiple images positively correlated with all classes. The simple mean label count in frame images yielded the best results, but not a significant amount. Correlation implied that no single method was distinctly better than any other method. Density estimation with label size as an unconventional method was relatively comparable with mean image correlation requiring mean density from multiple images as single image results were significantly weaker than any other method. This weak result was possibly due to close-up image results being more varied than frame images.

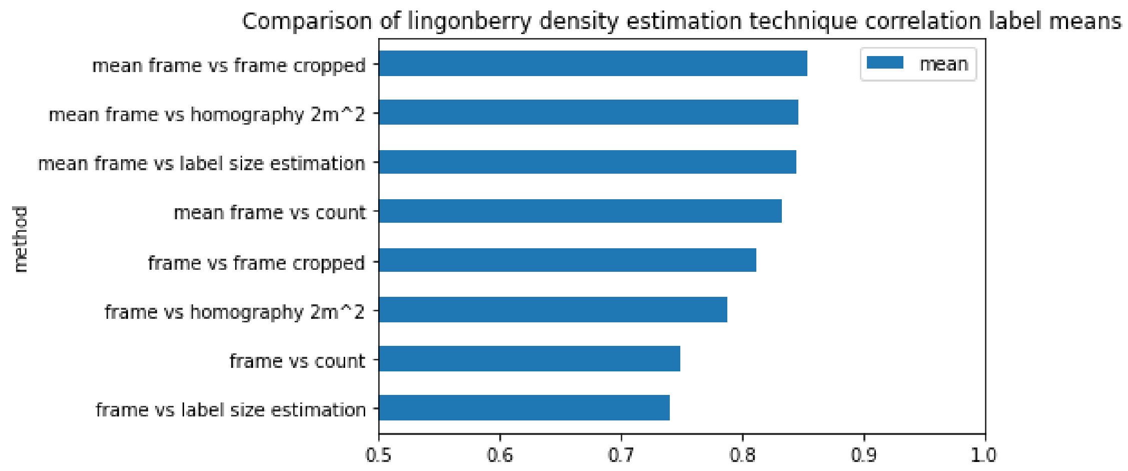


Figure 61. Correlation of lingonberry density estimation techniques by mean of all classes.

A separate comparison with only lingonberry classes was done, as bilberry correlations are more erratic than lingonberry (Figure 61). The bilberry correlation results likely stemmed from smaller frame datasets and homogenous deep learning training datasets. Correlation method results with only lingonberry implied that improvement of a single image and mean from multiple images may not be that significant with a more balanced dataset.

With all results considered, cropping labels with frames was the most promising method of density estimation (Figure 61). Homography results indicated that this more complex process did not yield significant improvement and thus did not justify its utilisation. The simple label count in the frame image method was viable if no alternative density estimation method was available. Density estimation with the label size method was significant in its capability to function without a frame in the image, and correlation results were not significantly weaker with the multiple images mean method. Although in future, a method to create a cropping polygon without a physical frame in an image with a known size would be more promising with correlation and yield better and more consistent results based on were estimation methods results.

According to the results, using the reference scale visible in the image yielded more accurate results than object-based density estimation. The simple label count in the frame image method yielded a higher correlation than density estimation with the label size method. Using pixel or texture-based methods are likely unnecessary, as an object-based method should produce better results (Silveira Jacques 2010), and the thesis results indicated using a reference scale was even more accurate.

8 CONCLUSIONS

The thesis described collecting image datasets and annotating lingonberry (Vaccinium Vitis-idea) and bilberry (Vaccinium Myrtillus) berries in three phases: The raw, flower and ripe. The resulting annotated dataset was used to train the YOLOv5 Object detection model. The process of improving the object detection model and the dataset were described. After adjusting and training the improved model, berry density was estimated with different techniques. The estimated correlation with field measurement density with multiple methods was compared.

The thesis suggests computer vision as a viable and highly potential method of detecting berries in different phases with deep learning computer vision. A computer is fully capable of detecting berries in their natural background. YOLOv5 object detection method can learn to detect berries from images without scaling the original image if an image is sliced into the object detector's native resolution. Berry density estimation with images yields promising results.

Splitting full-resolution consumer-level mobile phone camera images with a grid split into multiple smaller images is required to detect small objects in meaningful amounts. Pyramid split has potential, but more complex issues arise with a more complex method. Splitting the original image into smaller chunks is an excellent method to keep as much data as possible, but one should Keep It Simple, Stupid (KISS-principle). For training split, labels between split lines should be partially visible if 80% of the label area is contained in an image. This improved berry detection results.

Dataset balance is important. Saying "garbage in, garbage out" is not without merit. From this saying, "Mediocre in, mediocre out" could be deducted. The primary dataset initially lacked raw and ripe lingonberries. After this unbalance was corrected, bilberry annotations were the least robust classes. Results suggest deficiencies in bilberry classes with only a single main location (Ounasvaara) and the least number of annotations negatively affect results. In future, more bilberry annotations from other locations are required to balance the dataset.

The annotating process is labour intensive (Z.-Z. X. Wu 2022). Therefore, one should plan to annotate classes rigorously. Dataset quality was lowered with the decision to annotate lingonberries as single berry and bunch. More classes were also more complicated and reserved valuable time available for annotation. The main experience of berry annotation was periodically evaluating current annotations to keep the generated dataset as balanced and varied as possible.

A more balanced dataset with synthetic data would likely improve the object detection model (Nowruzi 2019). The additional synthetic dataset has the potential to generate more data in less time. A synthetic dataset would also help with balancing, as the number of generated synthetic images could vary between classes. The synthetic dataset also has the potential to limit misclassification with the dataset resembling the dataset used in this thesis. As all images had a natural background, plain white and other backgrounds in the dataset would likely minimise errors like object detection assuming blue shoes to be ripe bilberry. As a large label, this strongly and negatively affects the method of estimating the image area with label size.

Consumer-level mobile phone camera used for berry density estimation with berries in different phases detected from an image with deep learning object detection and image area estimation using various methods yields promising results. More reference data is needed for more comprehensive results as the primary dataset frame image number is low and not varied enough as images were mainly from a single distance from the ground. The main question is the correlation between field measurement control data with only seeing the top layer of berries visible in the image. The thesis suggests that estimating only with images taken on top of the growth shows promising results in estimating berry density in different phases. Combining results from multiple images from the study location yields more accurate results. The most preferred area estimation is cropping detected objects with a known area in an image. In future, this could be achieved, for example, with augmented reality applications creating a virtual frame in an image for cropping.

Estimation density with a mobile phone has potential. Berries can be detected from their natural background with computer vision. Density estimation from only visible berries correlates with field measurement density. Suggested improvements are: Combining object detection improvements, implementing an Augmented Reality virtual frame to estimate the image area, and instructing the user to take multiple images could lead to improved results and give potential to practical applications.

9 REFERENCES

Bayraktar, Ertugrul. Basarkan, Muhammed Enes. Celebi, Numan 2020. A low-cost UAV framework towards ornamental plant detection and counting in the wild. ISPRS Journal of Photogrammetry and Remote Sensing. ISSN 0924-2716. 1-11.

Benjumea, Aduen, Izzedin Teeti, Fabio Cuzzolin, and Andrew Bradley 2021. YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles. arXiv preprint arXiv:2112.11798.

Bohlin, Inka. Maltamo, Matti. Hedenås, Henrik. Lämås, Tomas. Dahlgren, Jonas. Mehtätalo, Lauri 2021. Predicting bilberry and cowberry yields using airborne laser scanning and other auxiliary data combined with National Forest Inventory field plot data. Forest Ecology and Management, Volume 502. ISSN 0378-1127.

Borel, Christoph. Young, S 2019. Image quality and super resolution effects on object recognition using deep neural networks.

Center for Science 2020. Supercomputer Mahti is now available to researchers and students – Finland's next generation computing and data management environment is complete. Search date 9. 7 2022. https://www.csc.fi/en/-/supercomputer-mahti-is-now-available-to-researchers-and-students.

Center for Science 2022. Docs CSC, Available systems. Search date 9. 7 2022. https://docs.csc.fi/computing/available-systems/.

DeTone, Daniel, Tomasz Malisiewicz, and Andrew Rabinovich 2016. Deep image homography estimation. arXiv preprint arXiv:1606.03798.

Dongjian, Wang. Dandan, He 2021. Channel pruned YOLO V5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning. Biosystems Engineering, Volume 210. ISSN 1537-5110. 271-281.

Galanty, Agnieszka . Tomasz Danel, Michał We˛grzyn, Irma Podolak 2021. Deep convolutional neural network for preliminary in-field classification of lichen species.

Gast, Nicolas. Ioannidis, Stratis. Loiseau, Patrick. Roussillon, Benjamin 2019. Linear Regression from Strategic Data Sources. Max-Planck Institute for Software Systems (MPI-SWS).

Gogul, I. and Kumar, Sathiesh 2017. Flower species recognition system using convolution neural networks and transfer learning.

Hackeling, Gavin 2017. Mastering Machine Learning with scikit-learn. Packt Publishing Ltd.

Hatab, Muhieddine, Hossein Malekmohamadi, and Abbes Amira 2020. Surface defect detection using yolo network. In Proceedings of SAI Intelligent Systems Conference. Springer, Cham. 505-515.

Ihalainen, Marjut. Salo, Kauko. Pukkala, Timo 2002. Empirical Prediction Models for Vaccinium myrtillus and V. vitis-idaea Berry Yields in North Karelia, Finland. Silva Fennica 37(1). 95–108.

Jocher, Glenn et al. 2022. YOLOv5 github.com repository README.md. Search date 25. 4 2022. https://github.com/ultralytics/yolov5/blob/master/README.md.

Kilpeläinen, Harri. Miina, Jari. Store, Ron. Salo, Kauko Salo. Kurttila, Mikko 2016. Evaluation of bilberry and cowberry yield models by comparing model predictions with field measurements from North Karelia, Finland. Forest Ecology and Management, Volume 363. ISSN 0378-1127. 120-129.

Krizhevsky, Alex and Sutskever, Ilya and Hinton, Geoffrey 2012. Neural Information Processing Systems.

Label Studio 2021. Announcing Label Studio v1.0. Search date 25. 4 2022. https://labelstud.io/blog/release-100.html.

Label Studio 2022. Import pre-annotated data into Label Studio. Search date 25. 4 2022. https://labelstud.io/guide/predictions.html.

Lecun, Y. and Bottou, L. and Bengio, Y. and Haffner, P 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 2278-2324.

Liu, Li & Ouyang, Wanli. Wang, Xiaogang. Fieguth, Paul. Chen, Jie. Liu, Xinwang. Pietikäinen, Matti 2018. Deep Learning for Generic Object Detection: A Survey. International Journal of Computer Vision.

Malis, Ezio. Manuel Vargas 2007. Deeper understanding of the homography decomposition for vision-based control. INRIA.

marjahavainnot.fi 2022. Luonnonmarjojen satohavainnot. Search date 25. 4 2022. https://marjahavainnot.fi/assets/info/Havaintometsan_perustaminen_v2.pdf.

Marmanis, Dimitris and Wegner, Jermaine and Galliani, Silvano and Schindler, Konrad and Datcu, Mihai and Stilla, Uwe 2016. SEMANTIC SEGMENTATION OF AERIAL IMAGES WITH AN ENSEMBLE OF CNNS. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. 473-480.

Ni, Xueping. Li, Changying. Jiang, Huanyu. Takeda, Fumiomi 2021. Three-dimensional photogrammetry with deep learning instance segmentation to extract berry fruit harvestability traits. ISPRS Journal of Photogrammetry and Remote Sensing, Volume 171. ISSN 0924-2716. 297-309.

Nowruzi, Farzan Erlik and Kapoor, Prince and Kolhatkar, Dhanvin and Hassanat, Fahed Al and Laganiere, Robert and Rebut, Julien 2019. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. 10.48550/ARXIV.1907.07061.

OpenCV 2022. Basic concepts of the homography explained with code. Search date 01. 07 2022. https://docs.opencv.org/4.x/d9/dab/tutorial_homography.html.

Puertas, Enrique, De-Las-Heras, Gonzalo. Fernández-Andrés, Javier. Sánchez-Soriano, Javier 2022. Dataset: Roundabout Aerial Images for Vehicle Detection. Data 7, no. 4. 47.

Redmon, Joseph. Divvala, Santosh Kumar. Girshick, Ross B. Farhadi, Ali 2015. You Only Look Once: Unified, Real-Time Object Detection. CoRR abs/1506.02640.

Rezatofighi, Hamid. Tsoi, Nathan. Gwak, JunYoung. Sadeghian, Amir. Reid, Ian. Savarese, Silvio 2019. Generalized intersection over union: A metric and a loss for bounding box regression. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019: 658-666.

Russakovsky, Olga. Deng, Jia, Su, Hao. Krause, Jonathan. Satheesh, Sanjeev. Ma, Sean. Huang, Zhiheng. Karpathy, Andrej. Khosla, Aditya. Bernstein, Michael. Berg, Alexander C. Fei-Fei, Li 2015. ImageNet Large Scale Visual Recognition Challenge.

Russell, Bryan C. Torralba, Antonio. Murphy, Kevin P. Freeman, William T 2008. LabelMe: a database and web-based tool for image annotation. International journal of computer vision 77, no. 1. 157-173.

Silveira Jacques, Julio Cezar Junior. Raupp Musse, Soraia. Jung, Cláudio Rosito 2010. Crowd Analysis Using Computer Vision Techniques. IEEE SIGNAL PROCESSING MAGAZINE.

Solawetz, Jacob 2020. YOLOv5 New Version - Improvements And Evaluation. Search date 25. 4 2022. https://blog.roboflow.com/yolov5-improvements-and-evaluation/.

Tzutalin 2015. LabelImg GitHub repository. Search date 29. 4 2022. https://github.com/tzutalin/labelImg.

Wu, Xinyu. Guoyuan Liang, Ka Keung Lee, Yangsheng Xu 2006. Crowd Density Estimation Using Texture Analysis and Learning. IEEE Int. Conf. Robotics and Biomimetics. 214–219.

Wu, Zhi-Ze. Xu, Jian. Wang, Yan. Sun, Fei. Tan, Ming. Weise, Thomas 2022. Hierarchical fusion and divergent activation based weakly supervised learning for object detection from remote sensing images. Information Fusion, Volume 80. 23-43.

Zitnick C Lawrence. Dollár, Piotr 2014. Edge boxes: Locating object proposals from edges. European conference on computer vision: 391-405.