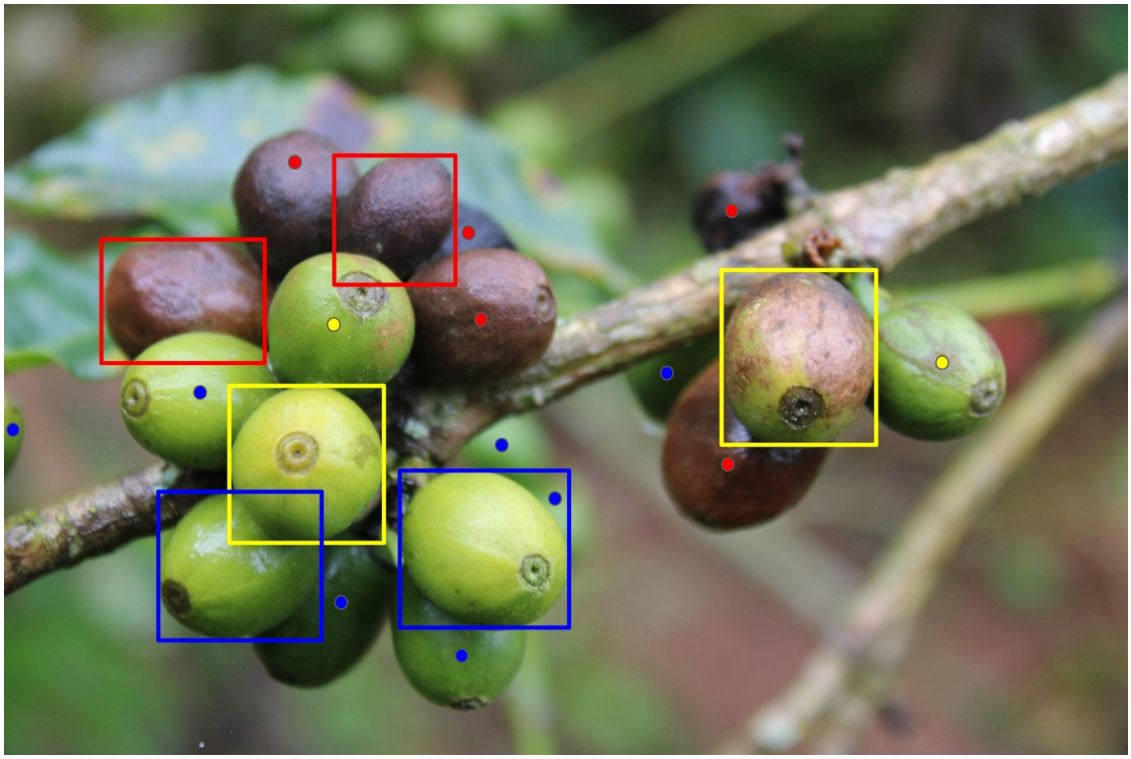


Weakly semi-supervised object detection for annotation efficiency

Leveraging a mix of strong bounding box labels and weak point labels for detecting coffee berry disease

Master's thesis in Complex Adaptive Systems

NILS EICKHOFF
AXEL EIMAN

DEPARTMENT OF PHYSICS
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Weakly semi-supervised object detection for annotation efficiency

Leveraging a mix of strong bounding box labels and weak point labels for detecting coffee berry disease

Nils Eickhoff
Axel Eiman

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Chalmers University of Technology
Gothenburg, Sweden 2023

Weakly semi-supervised object detection for annotation efficiency
Leveraging a mix of strong bounding box labels and weak point labels for detecting coffee berry disease

Nils Eickhoff
Axel Eiman

Supervisors: Olof Mogren and Aleksis Pirinen, RISE
Examiner: Mats Granath, Department of Physics

Master's Thesis 2023
Department of Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Image of coffee berries on the plant infected by coffee berry disease, annotated with points and boxes

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Weakly semi-supervised object detection for annotation efficiency
Leveraging a mix of strong bounding box labels and weak point labels for detecting coffee berry disease
Nils Eickhoff
Axel Eiman
Department of Physics
Chalmers University of Technology

Abstract

Annotating datasets is a common obstacle for many industries that may limit the potential for adopting machine learning methods. One example of such an industry is agriculture. Resources may be limited, especially in developing areas, but there is great potential for machine learning models to be used for applications such as tracking diseases. This work revolves around developing an efficient machine learning (ML) pipeline and using it to detect coffee berry disease (CBD) in a dataset with images of coffee plants. CBD is a fungal plant pathogen that is difficult to manage and often causes major problems for coffee production. Three common methods to alleviate the burden of manually annotating datasets are semi-supervised learning, weak supervision, and utilizing machine learning in the labelling process. Recently developed open-set detectors that boast impressive performance have a natural use case in this process. These models can predict bounding boxes for arbitrary objects without specific training and can therefore be used to generate proposals for ground truth bounding boxes in a dataset. Following this initial step, manual annotation using time-efficient point labels for the remaining objects in each image results in a mix of strong box labels and weak point labels in each image. This work explores this setting and proposes two models for the task; Point-guided loss suppression (PLS) and mixed Point-Teaching (MPT). The PLS model is a simple adaptation of YOLOv8, which when compared to the semi-supervised case gives a slight improvement in performance on the CBD dataset and a slight decrease in the MS COCO benchmark. The MPT framework consists of two models where one model is used to generate boxes that the other model uses as pseudo labels during training. The resulting performance for the MPT framework is generally worse, only performing above the baseline in a few cases. The exact efficiency of utilizing point labels is difficult to determine, but our results indicate that there are potential use cases where annotating points is more efficient than boxes, especially with further development of the models.

Keywords: object detection, annotation efficiency, coffee berry disease, weakly semi-supervised learning, computer vision, YOLOv8

Acknowledgements

We would like to thank RISE for the opportunity to conduct this thesis and specifically direct our gratitude towards Olof Mogren and Aleksis Pirinen. Thank you for your support and guidance throughout the project. We would also like to thank the rest of the deep learning team for the great discussions and fine company we have had during our time in this organization. Furthermore, we would like to extend our gratitude to Mpendakazi Agribusiness for their vital role in this project. Finally, a special thanks to our examiner Mats Granath, and our opponent Anton Sandberg.

Nils Eickhoff & Axel Eiman, Gothenburg, November 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AI	Artificial intelligence
AP	Average precision
CBD	Coffee berry disease
CNN	Convolutional neural network
EMA	Exponential moving average
IoU	Intersection over union
ML	Machine learning
MIL	Multiple instance learning
NLP	Natural language processing
MPT	Mixed point-teaching
PLS	Point-guided loss suppression
WSSOD-P	Weakly semi-supervised object detection with points

Contents

List of Figures

List of Tables

1

Introduction

1.1 Background

In the agricultural sector, predicting and preventing yield loss is necessary. Mpendakazi Agribusiness in Tanzania has problems with coffee berry disease and is looking for alternative solutions for detecting infection. In recent years, large progress within artificial intelligence (AI) has enabled new applications in computer vision, especially through deep learning. Training these deep learning models requires not only raw data but also annotations for the data. The annotation process can be very costly in terms of time and resources. However, there are ways to reduce the effort and this thesis aims to use new combinations of such methods to lower the cost even further, without degrading the performance.

1.1.1 Coffee Berry Disease (CBD)

An infectious fungus called *Colletotrichum Kahawae* causes coffee berry disease (CBD) in Arabica coffee plants. The pathogen is mainly constrained to Africa but has recently been reported in other parts of the world as well [CABI, 2022]. The berries show two symptoms of the disease; scab lesions and active lesions. Active lesions are dark, sunken spots that grow to cover the whole berry, causing it to rot. Pink spore masses, that eventually turn white, can occur on the lesions given humid conditions. The infection can also occur in the young berry stalks, which can cause shedding before the berries show symptoms. Scab lesions appear pale, corky, and slightly sunken. These can appear on both young and mature berries and may heal completely or turn into active lesions once the berry ripens. The symptoms of the disease can be detected visually and two examples are shown in figure 1.1 Therefore, this presents a good use case for an automated vision-based solution to detect these symptoms.

The disease has had a significant impact on Arabica crops since it was first recorded in 1922. At high altitudes, crop losses can range from 50-80% [Prihastuti et al., 2009]. Even with extensive control efforts that can account for up to 45% of the annual production costs for a field, unfavorable weather conditions can facilitate losses of up to 50%. The main control measure used for the disease are fungicides. Copper-based solutions are the most common as they are cheaper than their organic counterparts.

(a) Scab lesions

(b) Active lesions

Figure 1.1: Subfigure 1.1a shows an example of berries infected by scab lesions. These are pale, corky marks that are sometimes slightly sunken. Subfigure 1.1b shows an example of berries with active lesions. Here we see the symptoms covering the berries, giving them darkened and sunken skin.

Despite being the main control measure, the process is both expensive, tedious and can be harmful to the environment.

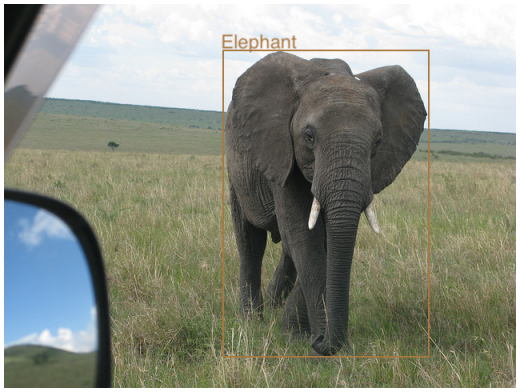
Once berries are infected they can shed from the branch, and if they don't they are discarded by the farmers. In some cases, there is a possibility to harvest infected berries and use them for local consumption if the symptoms are mild. Assessing the disease is difficult as berries may or may not drop to the ground, and scab lesions may or may not result in the loss of the berry. According to [CABI, 2022], it is difficult to accurately assess the disease and the loss it causes, but can be done by recording what happens to specific branches.

1.1.2 Object detection

Object detection is a computer vision task where the goal is to localize and classify objects in images. The most common case is closed-set object detection. Given a set $\mathcal{C}$ of classes (e.g. *Bird*, *Elephant*, *Frisbee*, *Person*) the goal is to localize and classify each object instance in an image from the given set of classes; see an example in figure 1.2. Creating tight bounding boxes containing the object achieves localization, and classification is done by assigning one of the classes to the box.

Algorithms for object detection commonly utilize deep learning and achieve remarkable performance in many cases. Historically most models have been based on convolutional neural networks (CNNs) and these can be categorized as one-stage approaches or two-stage approaches [Sultana et al., 2020]. Two-stage detectors have a separate part proposing the regions and another for classifying them, while one-stage detectors do this in a single stage. In recent years, transformer-based models have been gaining popularity for object detection, such as DETR [Carion et al., 2020] and its derivatives.

This is a rapidly developing field where new deep learning models frequently improve over each other, achieving better performance than their counterparts. Most of these



(a) All objects are correctly predicted.

(b) Some objects are correctly predicted.

Figure 1.2: In figure 1.2a, there is only one object, namely the elephant. There is only one bounding box and it is tight around the elephant with the desired label *Elephant*, a well-executed prediction. In the second figure 1.2b there are four objects; three people and one frisbee. The bounding box to the left is once again tight around the object with the correct label *Person* and therefore a successful prediction. The uppermost bounding box is also tight around the object but has the wrong label *Bird* instead of the correct label *Frisbee*. The last bounding box is not tight around the object, but has the correct label *Person*. Out of the four objects, one person has not been detected since it lacks a bounding box. The bounding boxes and class labels were drawn by the authors of this thesis, and the images were collected from the COCO dataset [Lin et al., 2014]

models are fully supervised, which means they rely on a training set consisting of images with associated bounding boxes and class labels for each object of interest in the image. Usually, not all objects in an image are of value for the user, meaning a limited number of objects are of interest and the rest are considered as background.

Object detection models can be trained on large datasets that cover a wide domain with many object categories. This can enable object detection models to generalize well [Michaelis et al., 2020]. Recent developments in natural language processing (NLP) have paved the way for new applications within this area. By combining NLP with object detection models trained on large datasets, models such as Grounding DINO [Liu et al., 2023] can detect arbitrary objects not present during training. Normally, object detection models are trained on a finite set of classes $\mathcal{C}$, hence the term closed-set. Grounding DINO performs open-set detection by introducing language to a closed-set detector. This is achieved by allowing human inputs in the form of text prompts. For instance, the user can enter category names such as *berry* or *leaf*, and the model will try to detect objects in these categories. By essentially removing the need for case-specific training, this becomes a quick-start option in many use cases. Depending on the application the performance may vary since it is easier for a model to generalize to a domain more similar to the one it has been trained on.

1.1.3 Labelling datasets for object detection

Most state-of-the-art object detection methods are based on having ground truth labels in the form of bounding boxes, with specific classes labeled for each box. They act as the target for what the model learns. Datasets are often labelled manually, requiring annotators to go through the dataset and add the required labels. In the case of object detection, annotators have to draw bounding boxes tightly around all objects of interest in the image and assign classes. This process is labour-intensive and often a substantial bottleneck for the implementation of many machine learning (ML) systems, not limited to object detection. The field has seen controversies where cheap labor in crisis-ridden areas was exploited to supply annotations for ML datasets [Miceli et al., 2020].

Based on the above-mentioned problems with providing full supervision, it is increasingly relevant to train models with minimal annotation effort. The main categories mentioned by [Ge et al., 2023] are weak supervision and semi-supervised learning. Here, weak supervision only corresponds to labels containing less information than normal boundary boxes, e.g. points, squiggles, or image-level categories. These weaker labels are easier to produce and therefore reduce the annotation cost. This is in contrast to semi-supervised learning, which in this thesis refers to the combination of supervised and unsupervised learning, meaning only a fraction of the training data is labelled. Weakly- and semi-supervised approaches reduce the required annotation effort, but the resulting performance is often lacking compared to the fully supervised option. In weakly semi-supervised learning, such as [Ge et al., 2023], one fully labelled set and one large weakly labelled set of images are used in a mix between weak and semi-supervision.

Integrating ML into the labelling process is another option to reduce the required effort. A central approach considered in this thesis is using pre-trained models that can generate preliminary labels. These can be reviewed and changed by annotators, which is less laborious than creating them from scratch. Another method that is not considered in this work is active learning, which uses an iterative solution in which algorithms can query specific data to be labelled as it is trained to maximize the impact of the annotation effort.

To minimize annotation effort and develop a seamless pipeline it is useful to take advantage of several of these approaches. For object detection, open-set detectors can be used to generate preliminary labels to start the labelling process. If we want to combine this process with cheap labels, the weakly semi-supervised setting with points (WSSOD-P) is an interesting option. The open-set detector generates useful preliminary labels for each image, and an annotator can then adjust these as needed and complement them with much cheaper point labels. This process results in a new interesting case with a mix of weak and strong labels within each image, which is an important difference from the normal WSSOD-P case, like [Ge et al., 2023] proposed, where images are either fully or weakly labelled. This setting with mixed label types within images is central to this thesis and will be denoted as *mixed* WSSOD-P.

1.1.4 Point-Teaching

The current state of the art for WSSOD-P is called Point-Teaching [Ge et al., 2023]. A student-teacher pair, as first proposed by [Tarvainen and Valpola, 2017], builds up the main training framework along with a few key components. The student and teacher are two models with the same architecture, where the teacher produces pseudo box labels which the student then trains on. The teacher is then updated with the exponential moving average (EMA) from the student. To generate pseudo labels for the point annotated images, a point-matching method based on the Hungarian algorithm [Kuhn, 1955] and a data augmentation process termed point-guided copy-paste are proposed by [Ge et al., 2023]. To train the student during training, two multiple instance learning (MIL) losses are used, one point-wise as suggested by [Ge et al., 2023] and one image-wise as proposed in [Bilen and Vedaldi, 2016]. MIL loss is used for weak supervision by treating multiple predictions as bags rather than individual instances. The framework was implemented with the Faster R-CNN model [Ren et al., 2015] and trained on the MS COCO dataset [Lin et al., 2014]. It achieved the current best performance for WSSOD-P with 33.15 in AP on the MS COCO dataset, getting remarkably close to the performance of fully supervised models.

1.2 Aim

The aim of this project is to annotate a dataset and then develop a framework that can utilize a mix of bounding box and point annotations in the same image for object detection tasks. This is to minimize costs associated with labelling through preliminary labelling and cheap point labels, while still achieving good model performance. The developed framework will be tested on the task of detecting coffee berry disease in a relatively small dataset of 203 images, but where the total number of object instances exceeds 30000. The results will give insight into whether this framework is of use for agricultural companies like Mpendakazi Agrobusiness in Tanzania, and if so this weakly semi-supervised framework will ease their annotation effort.

1.3 Limitations

- Active learning solutions will not be implemented in this project.
- Off-the-shelf models and backbones will be used to enable fair and useful comparisons to other use cases.

1.4 Research questions

The work in this thesis revolves around the following research questions:

- How can a framework for WSSOD with point and box annotations in the same image be designed?

- What are the implications of such a pipeline for the annotation effort required and the resulting model performance?
- How do the size and characteristics of the dataset affect the performance of such an object detection pipeline? Factors that will be investigated or discussed are the number of images, the number of instances per image, the total number of instances of each class, as well as the fraction of instances labelled with point labels versus box labels.
- What applications may this framework be useful for?

2

Theory

2.1 Metrics

In the realm of object detection there are many ways to evaluate the quality of single prediction as well as overall model performance. The choice and interpretation of the metrics chosen for evaluation plays an important role in various design choices in one's solution, as well as in understanding the results. Metrics relevant to this project are described below.

2.1.1 Intersection over Union (IoU)

There are many different types of object detection frameworks with different layouts, but one thing they have in common is a step of bounding box regression. This is the step that makes the prediction of a rectangle to specify the location of an object. The most popular metric to evaluate predicted boxes is the Intersection over Union (IoU) of the predicted box B_p and the ground truth box B_{gt}:

$$IoU = \frac{|B_p \cap B_{gt}|}{|B_p \cup B_{gt}|} \tag{2.1}$$

This metric gives a score of one for a perfectly predicted box that is equal to the ground truth. If the boxes don't overlap perfectly, the intersection will be smaller than the union of the boxes, resulting in a lower score. In cases where the boxes don't overlap at all the score is zero. See figure 2.1.2 for an example of IoU.

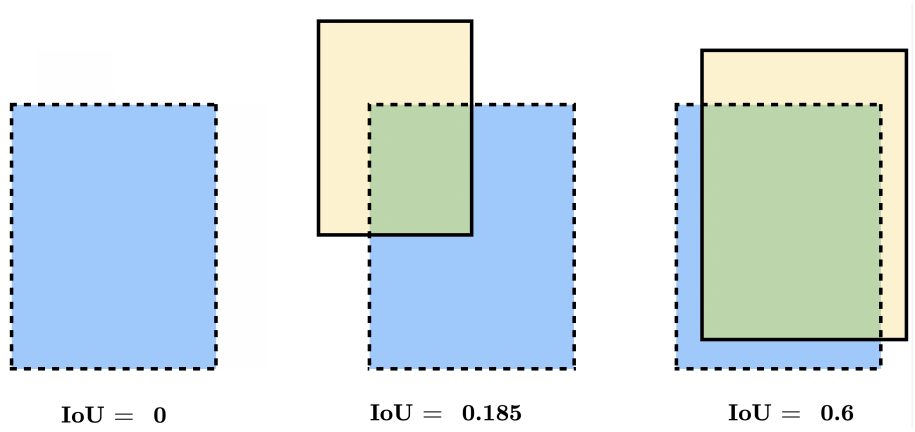


Figure 2.1: A figure illustrating three common IoU cases with an IoU threshold of 0.5, see section 2.1.2 for an explanation about where this threshold is used. Here, the dashed blue box is a ground truth and the solid yellow box corresponds to a prediction. Left: No prediction, therefore no overlap, and IoU=0. A missed object results in a false negative. Middle: A prediction that intersects the ground truth. The IoU here is less than the threshold so a false positive. Right: A prediction that intersects more and has an IoU=0.6, which is higher than the threshold and results in a true positive.

2.1.2 Average Precision (AP)

In object detection the interpolation of the average precision (AP), proposed by [Harman, 1986], is used as a benchmark and the AP is calculated as the area under the precision-recall curve. The different instances of precision-recall pairs are obtained by changing the threshold for selecting a prediction from the model, also known as the confidence level. Models generally provide a confidence score with their predictions, and by decreasing the threshold for what the model should predict as positive, one gets more predictions.

To determine whether a predicted box is correct, a threshold for IoU between predicted and ground truth boxes is set. Predicted boxes with IoU to a ground truth box greater than this threshold are considered correct predictions. Those with lower IoU than the threshold are considered incorrect predictions; see an example of this in figure 2.1. The choice of threshold essentially determines how accurate one requires the box to be, a higher threshold requires more accurate predictions. In most cases, object detection problems have multiple classes, so AP is calculated for each class, and the mean average precision (mAP) is used.

A fixed threshold of IoU ≥ 0.5 would be represented as mAP@50. Sometimes the average along an interval of such IoU thresholds, with a certain step size, is used. Moving from 50% to 95%, with the normal step size of 5%, would be denoted as mAP@50:95.

2.1.3 F1 score

F1 score is the harmonic mean of precision and recall, so a good F1 score indicates that we find many objects without making many wrong predictions:

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2.2}$$

3

Methods

3.1 Dataset and labelling

A dataset of 203 images was collected from farms in the Mbeya, Songwe, and Kilimanjaro regions by Mpendakazi Agribusiness in Tanzania. The images are photographs of coffee plants infected with CBD. There are both healthy and infected berries in the images and the images are quite varied, see figure 3.1. Some images are taken closer to the branch, only showing a few berries, while some are wider shots containing more than a thousand berries. The depth of field is quite narrow in some of the images and many berries are quite blurry. In some cases there are similar images, several photos of the same branch with slightly different angles, focus, and zoom. An example with two similar images is presented in figure 3.2. Even though the dataset is quite small in terms of the number of images, there is a large number of berries in many of the images. This means that an algorithm trained on this dataset has many instances of each class to learn from despite the small number of images.

(a) Few berries in even light

(b) Many berries in even light

(c) Backlight and dark berries

Figure 3.1: The figures show the wide variation between the images in the dataset, displaying a deviation in the number of berries, the level of contrast, and the light condition.

To compare with fully supervised detectors and different levels of semi-supervision, the dataset was fully labelled with bounding boxes by the authors. Labels were first generated for a subset of the images in collaboration with a state-of-the-art pre-trained open-set detector, Grounding DINO [Liu et al., 2023]. The model was given the word *berry* as a text prompt, and returned associated objects as

(a) Taken a bit more to the left

(b) Taken a bit more to the right

Figure 3.2: Subfigure 3.2a is photographed more to the left and subfigure 3.2a more to the right. The differences in the images are very small and the same berries are present in both images, hence the images are considered similar.

labeled bounding boxes. Using the open-source labelling software Label Studio [Tkachenko et al., 2022], these predictions were reviewed and additional boxes added for any instances not detected by Grounding DINO. With a subset of the dataset labelled, a YOLOv8 model was trained and used to create more accurate predictions on the rest of the dataset. These predictions were then similarly reviewed and supplemented with boxes where needed to provide each object in the dataset with box labels.

To train and evaluate the models developed in this project, the dataset was split into training, validation, and test sets with 143, 30, and 30 images, respectively. For the sake of independence between training, test, and validation, the test- and training set were sampled without similar images.

The validation and test sets remained fully annotated with boxes, and the fraction of box labels in each image in the training set can be varied for experiments. To vary the fraction of box labels, random boxes are sampled to be kept as box labels while the rest are either removed to create a semi-supervised setting, or changed into a point by sampling a point within the box. As the objects of interest in this dataset are berries with a round shape, point labels were sampled from a multivariate normal distribution with the means being the center coordinates of the box, and variance being dependent on the width and height respectively. The variance was chosen so that the edges of the box are three standard deviations away from the mean to ensure most points naturally end up within the box. An example of this is presented in figure 3.3 below. Any points outside the box are clipped to the edges of the box.

3.2 Point-guided loss suppression (PLS)

To utilize points and boxes in the same image, the first solution proposed is a customized YOLOv8 model [Jocher et al., 2023] that can train in the mixed label setting. We call this a Point-guided loss suppression (PLS) framework, and the code for this framework will be made available on GitHub [Eiman and Eickhoff, 2023].

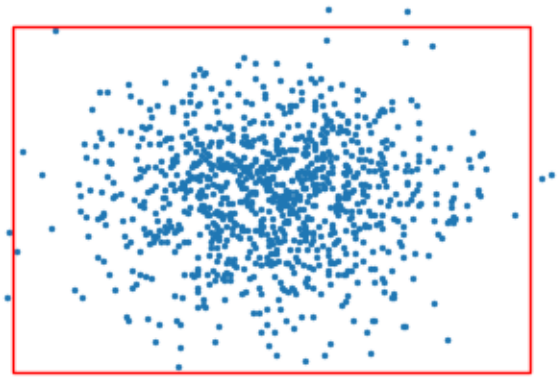
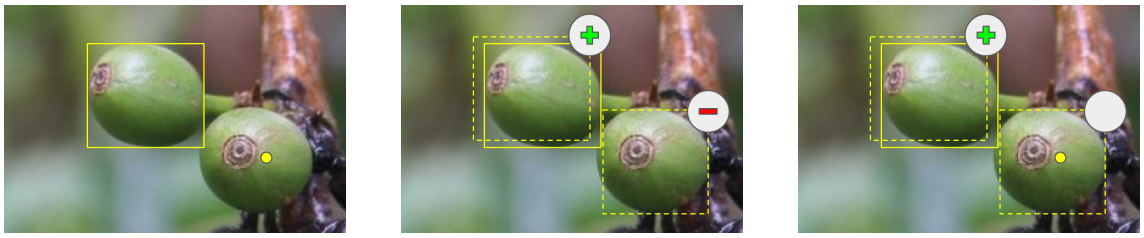


Figure 3.3: Constructed example of 1000 points sampled from a single bounding box. The points come from a normal distribution with the means of the x and y coordinates in the center of the box, and variance dependent on the width and height.

YOLOv8 is the latest version of the well-known YOLO models, and was chosen as it has been shown to outperform previous versions [Jocher et al., 2023]. The model was kept as similar to the original as possible. The changes made let it take advantage of point labels in quite a simple manner. As the name implies, point labels are used to suppress the loss of the model. When the model makes a prediction that encloses a point, the loss is suppressed for that prediction. Apart from how it can use point labels, it is equivalent to the original model in how it uses box labels.

The idea is that the point labels is used to avoid a high loss for predictions on objects present in the image that lack a box label. In the semi-supervised case discussed in this project, each image contains some objects that do not have an associated label. Without any adaptations from a normal object detection model, this results in a high loss for predictions that would be considered good given proper ground truth boxes. Utilizing the point labels to disregard any prediction made on such an object is expected to mitigate this issue. This concept is visualized with constructed examples in figure 3.4. A few changes had to be made to the model for the point labels to work at all, as YOLOv8 uses labels for more than just computing the value of the loss functions. The YOLOv8 architecture uses a grid of many so-called anchor points, where each anchor point predicts a bounding box. When calculating loss, candidate predictions for each ground truth object are selected as those anchor points the label encloses. Annotated points have no area and can not enclose anything, which poses a problem where a different solution is required for point labels. The chosen solution, for replacing the functionality of ground truth boxes enclosing anchor points, was to select candidate anchor points from within a radius of the given ground truth point. The radius was chosen to provide each ground truth point with around 20 candidate predictions, a comparable number to what box labels in the CBD dataset get.



(a) Ground truth labels **(b)** Semi-supervised case **(c)** PLS

Figure 3.4: The figure compares semi-supervised learning in this mixed setting with PLS. Solid rectangles represent ground truth boxes, the points are ground truth points, and the dashed rectangles correspond to predictions. Subfigure 3.4a shows the ground truth labels in this example, one box- and one point label. Subfigure 3.4b illustrates the semi-supervised learning case. Here, only the ground truth box is available. The prediction on the top left object gets a low loss since it overlaps the ground truth box well, while the prediction on the object lower to the right of the image results in a high loss due to the absence of a ground truth box. Subfigure 3.4c shows the point-guided loss suppression, where the ground truth point is present. Again, the prediction on the top left object gets a low loss, but the lowermost prediction encloses the ground truth point label and the loss is suppressed.

From the group of candidate predictions for each ground truth box label, the prediction with the highest IoU is selected. Point labels are incompatible with IoU and difficult to score which poses a problem. Therefore, a prediction is only matched with a point if it does not overlap with any ground truth box. Where there are several predictions for a ground truth point, they are prioritized by distance from the center of the predicted box to the ground truth point instead of the IoU.

3.3 Mixed Point-Teaching (MPT)

The framework described here is based on the Point-Teaching framework proposed by [Ge et al., 2023]. The code developed for this framework will be made available on GitHub [Eiman and Eickhoff, 2023]. With the use of the mixed setting with both box and point labels within images there are some key differences between the MPT framework and the original. The framework and its differences from the original are described below.

The framework consists of two initially identical models that are trained jointly in a mutually beneficial manner. One is called a teacher model, whose purpose is to produce pseudo-labels that the student model can train on. The student model is the main model, responsible for making predictions once trained.

To describe the layout of the training process, it starts with a teacher model producing predictions of bounding boxes on an image labelled with a mix of ground truth bounding boxes and points. By matching these predictions with ground truth point labels, they can be used as pseudo-labels to train the student model. Only the predictions that enclose a ground truth point of the same class are kept, and

the rest are discarded. Using a Hungarian-based point matching method, as proposed by [Ge et al., 2023], each point label is then matched with a single box. For any remaining point not matched with any of the predictions, a pseudo box label is placed there using an augmentation method we call point-guided label copy-paste. In this method, the closest label of the same class within the image is copied to the location of the point. If there are no available box labels of that class in the image, a random one is copied from the dataset instead. After this step, all points have an associated box which will be used to train the student model with image- and point-wise multiple instance learning (MIL) loss, represented as $\mathcal{L}^I_{\text{mil}}$ and $\mathcal{L}^P_{\text{mil}}$, respectively. MIL essentially means that multiple instances are treated as a *bag*, rather then the usual case where the instances are treated individually. These losses are balanced with two gain parameters, λ_1 for the image-wise MIL loss and λ_2 for the point-wise MIL loss. The overall loss $\mathcal{L}$ is calculated as:

$$\mathcal{L} = \mathcal{L}_{\text{det}} + \lambda_1 \mathcal{L}^I_{\text{mil}} + \lambda_2 \mathcal{L}^P_{\text{mil}} \tag{3.1}$$

where $\mathcal{L}_{\text{det}}$ consists of the sum of losses in the original YOLOv8 model.

The teacher is then updated by an exponential moving average (EMA) from the student; see Ultralytics [Jocher et al., 2023] for details regarding implementation and parameter values. See an illustration of the MPT framework in figure 3.5.

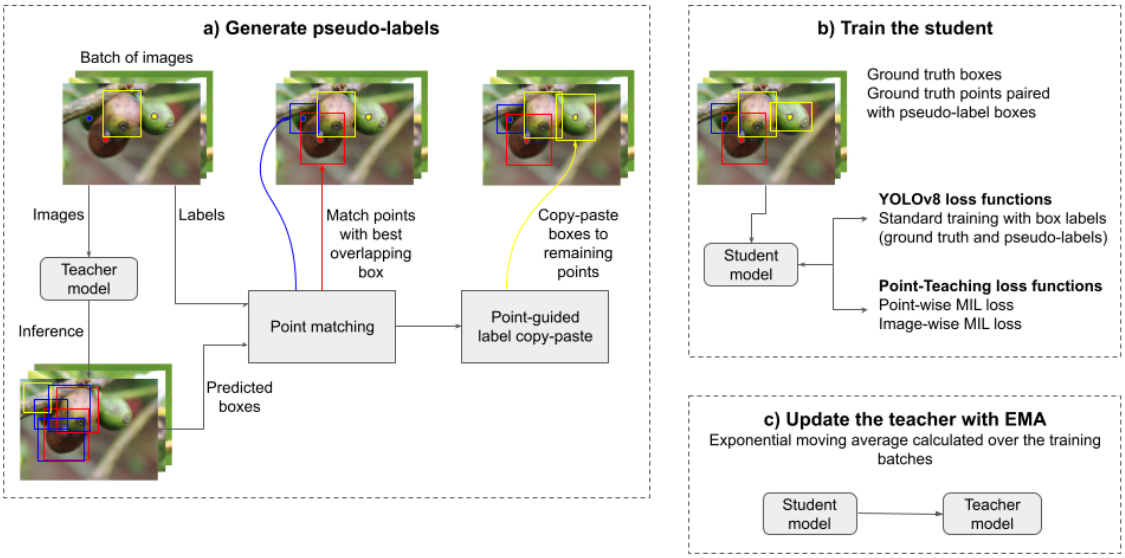


Figure 3.5: An illustration of the MPT framework. For each training iteration, the teacher model makes a prediction on the image. These predictions are then matched to point labels with the point-matching algorithm, and the point-guided label copy-paste will make sure that any unmatched point also gets a box. These now fully labeled images are then used as input for the student model, which is trained with the YOLOv8 loss functions, as well as the point- and image-wise MIL loss functions. Lastly, the teacher is updated with EMA of the student.

In order to improve the quality of the pseudo boxes containing point labels and balancing the frequency of different classes, one so called Objectness-P branch and one Objectness-I branch are added to the detection head of the YOLOv8 model, respectively, as proposed by [Ge et al., 2023]. The Objectness-I branch has exactly the same architecture as the classification branch of the YOLOv8, while the Objectness-P branch differs in the last output layer, having two outputs per predicted box.

3.3.1 Point matching

Each ground truth point label should be matched with the most suitable predicted box from the teacher's predictions. By using the Hungarian-based point matching proposed by [Ge et al., 2023], the best-predicted box is calculated according to a spatial and a classification cost. The spatial cost is designed to be low for predicted boxes with equal class label as the annotated point and also for boxes that enclose the point. The classification cost is low for predicted boxes with a high confidence score in the Classification branch and the Objectness-P branch. Let $\mathcal{L}_{\text{match}} \in \mathbb{R}^{N_p \times N_b}$ be a cost matrix, where N_p denotes the number of annotated points and N_b the number of predicted boxes. Let the index i represent an annotation point and the index j corresponds to a predicted box. Then the cost for a predicted box b_j and an annotated point $\hat{p}_i$ is calculated as:

$$\mathcal{L}_{\text{match}}(i,j) = \underbrace{(1 - \mathbb{1}[\hat{p}_i \text{ in } b_j] \cdot \mathbb{1}[\hat{p}_i^l = b_j^l])}_{\text{spatial cost}} + \underbrace{(1 - \sigma(s_{j,\hat{p}_i^l}) \cdot \sigma^P(s_{j,1}^P))}_{\text{classification cost}} \tag{3.2}$$

In equation (3.2), $\hat{p}_i^l$ corresponds to the class label of the annotated point $\hat{p}_i$ and likewise b_j^l is the class label of the predicted box b_j. Let C be the number of classes apart from the background, then $s \in \mathbb{R}^{N_b \times (C+1)}$ represent the Classification branch score and $s^P \in \mathbb{R}^{N_b \times 2}$ corresponds to the score of the Objectness-P branch. σ and σ^P is the *softmax* operation along the second dimension of the Classification and Objectness-P outputs, respectively.

With the cost matrix described, the matching task can be defined as a bipartite matching problem:

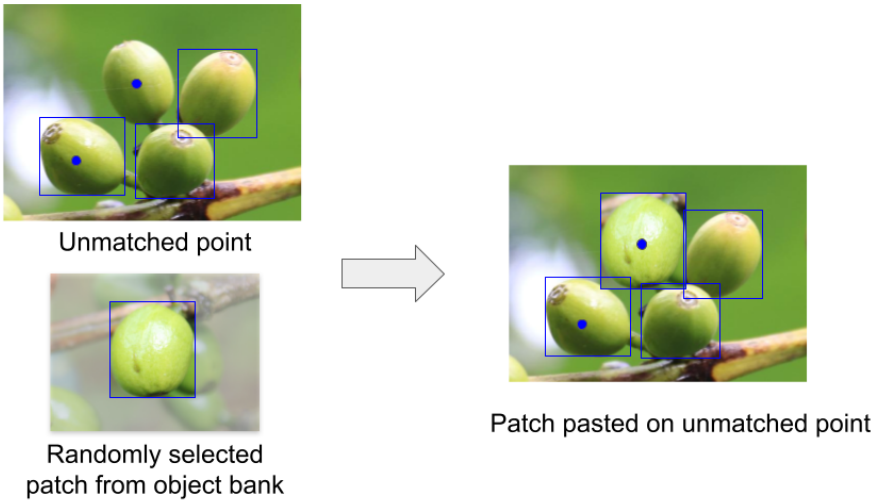
$$\hat{\pi} = \underset{\pi \in \mathfrak{S}_{N_b}}{\arg\min} \sum_{i}^{N_p} \mathcal{L}_{\text{match}}(i, \pi(i)) \tag{3.3}$$

Here, $\pi \in \mathfrak{S}_{N_b}$ is a permutation of N_b elements. Equation (3.3) can be solved with the Hungarian algorithm [Kuhn, 1955]. In the end, some points will not have an enclosing box and therefore be *unmatched*.

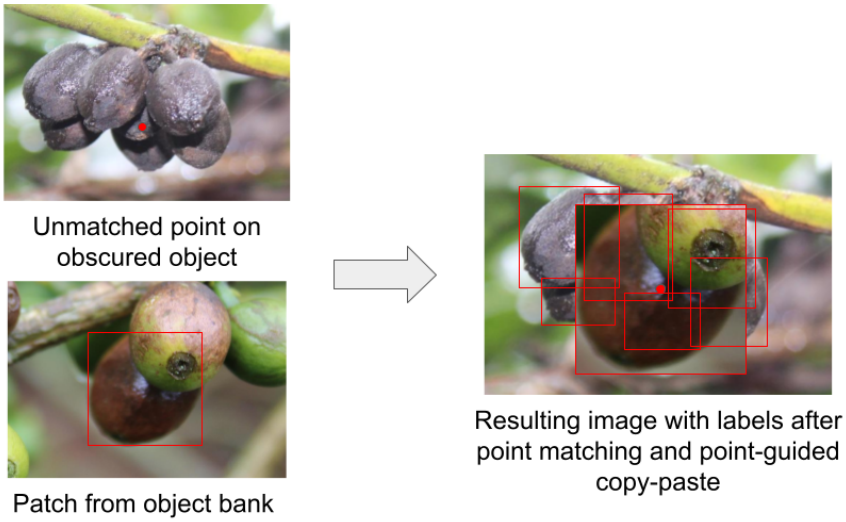
3.3.2 Point-guided label copy-paste

Some ground truth points are likely missed in the teacher's prediction and will therefore be left unmatched after the point matching procedure. To combat this, point-guided copy-paste introduced by [Ge et al., 2023] suggests having a dynamic object bank with patches cropped from objects with ground truth labels as well as

pseudo-labelled objects from the current batch. These patches are then pasted on unmatched points to replace them in the image, see figure 3.6. Pasting parts of an image cropped by box labels seems beneficial in most cases, but with obscured or overlapping objects a clear problem arises. Pasting a patch may cover a significant part of one or more ground truth objects and their associated labels as shown in figure 3.6b.



(a) A constructed example showing how the original point-guide copy-paste works.



(b) A constructed example showing how pasting random object patches can create issues in cases with obscured or overlapping objects. The resulting image has several objects completely covered by the pasted patch as seen by the red box labels.

Figure 3.6: How the original point-guided copy-paste from [Ge et al., 2023] works and why it may be detrimental in overlapping cases.

As the CBD dataset contains many overlapping objects, we propose an alternative solution. Instead of the dynamic object bank containing patches of images cropped by labels, a dynamic label bank is kept with a similar process. It contains all ground truth box labels in the dataset and the generated pseudo-labels of the current batch. For each unmatched point, the closest label of the same class in the same image is selected and pasted if available. Otherwise, a random label is selected from the bank. This is based on the assumption that the shapes and sizes of objects in the same image are more closely distributed than they are in the full dataset.

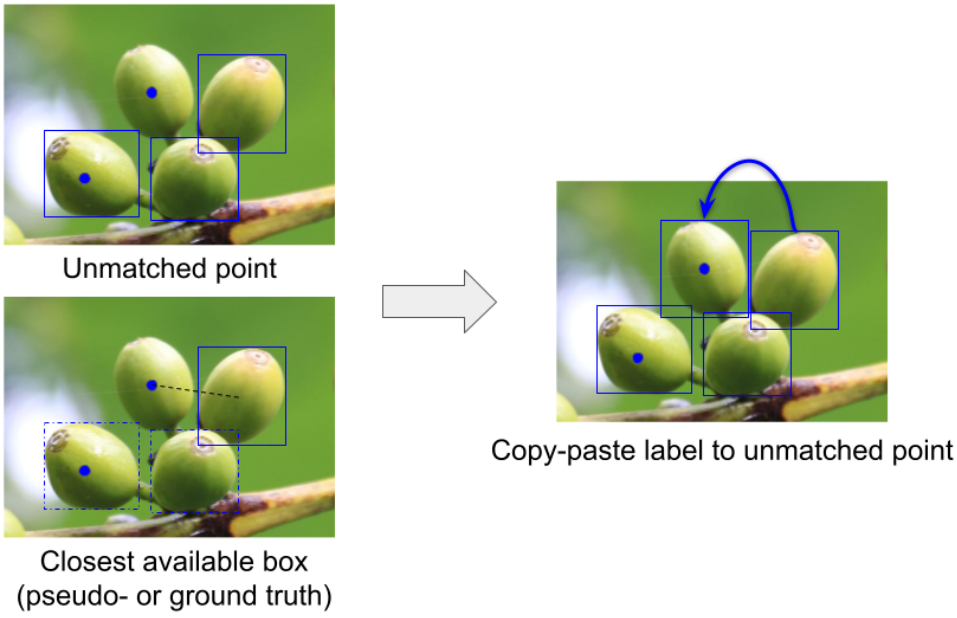


Figure 3.7: A constructed example showing how point-guided label copy-paste works. For each unmatched point, the closest available pseudo or ground truth box with the same class label is copied. In the event of no suitable boxes in the image, a box with the same class label is randomly taken from the dynamic label bank. The center of the box is positioned so that it matches the point. Note that only the label is copied and not the patch.

3.3.3 Image-wise MIL loss

The Objectness-I branch optimized with image-wise MIL loss is intended to suppress inconsistent classification predictions with image-level annotations [Ge et al., 2023]. The branch is added in parallel with the two existing branches in the head of YOLOv8 and has the same layout as the classification branch.

From the ground truth labels in each image, a one-hot image-level label vector denoted $\hat{\phi} = [\hat{\phi}_1, ..., \hat{\phi}_C]$ is created for which a class is present in the image or not. The outputs of the classification branch and the Objectness-I branch, denoted $\mathbf{s}, \mathbf{s}^I \in \mathbb{R}^{N_b \times C}$ respectively, are then used to calculate the image-wise MIL loss. With σ^I, σ denoting the *softmax* operation on the first and second dimension respectively, we can create two score matrices $\sigma(\mathbf{s})$ and $\sigma^I(\mathbf{s}^I)$. By taking the element-wise

product of these score matrices denoted $X^{\mathbf{s}}$, and summing over the predicted boxes, image-level classification scores are obtained:

$$\phi_c = \sum_{i=1}^{N_b} X_{ic}^s = \sum_{i=1}^{N_b} \left[\sigma(s)_{i,c} \odot \sigma^I(s^I)_{i,c} \right] \tag{3.4}$$

The image-wise MIL loss is then calculated as the sum of the binary cross entropies across all categories of the image-level labels and prediction scores:

$$\mathcal{L}_{\text{mil}}^I = -\sum_{c=1}^{C} \left(\hat{\phi}_c \log(\phi_c) + (1 - \hat{\phi}_c) \log(1 - \phi_c) \right) \tag{3.5}$$

In equation (3.5) $\hat{\phi}_c \in \{0, 1\}^C$ is the image-level one-hot labels.

One thing to note is that the Objectness-I branch and image-wise MIL loss likely have minimal effect when applied to the CBD dataset. Most images contain instances of all three classes, so the image-level label vector for most images is $\hat{\phi} = [1, 1, 1]$ which adds very little information. For applications with a larger number of classes, such as the COCO dataset it should intuitively be more effective.

3.3.4 Point-wise MIL loss

The Objectness-P branch and point-wise MIL loss introduced by [Ge et al., 2023] aim to encourage that each ground truth point label gets exactly one prediction. The Objectness-P branch performs binary classification on whether a given box is the best prediction inside a bag. It has the same architecture as the Objectness-I branch, except for the output layer, which gives two values per prediction.

For each point label $\hat{p}_i$, a bag Ψ_i is created which consists of boxes with the same class label as the point but also encloses the point. Let the class label of the predicted box b_j be denoted b_j^l and the class label of an annotated point $\hat{p}_j$ be denoted $\hat{p}_j^l$. Then the bag $\Psi_i = \{b_j | \mathbb{1}[\hat{p}_i \text{ in } b_j] \cdot \mathbb{1}[\hat{p}_i^l = b_j^l]\}$, where only the best predicted box b_j for a corresponding annotated point $\hat{p}_j$ will be positive and the rest of the boxes negative.

The Objectness-P branch performs binary classification to predict whether a box is the best prediction in a bag, with the output denoted $s^P \in \mathbb{R}^{N \times 2}$. A confidence score φ_i for bag Ψ_i is computed as:

$$\varphi_i = \sum_{k=1}^{|\Psi_i|} \left[\sigma(\mathbf{s})_{k,\hat{p}_i^l} l \odot \sigma^P(s^P)_{k,1} \odot \prod_{m \neq k} \sigma^P(\mathbf{s}^P)_{m,0} \right] \tag{3.6}$$

In equation (3.6), $|\Psi_i|$ corresponds to the number of predicted boxes in the bag Ψ_i. The score exponentially decreases with the number of predicted boxes in a bag, as there are more factors $\sigma^P(\mathbf{s})_{m,0} < 1$. This way the score can encourage only predicting a single box per object.

Let the number of annotated points be denoted N_p, then the Point-wise MIL loss can be calculated by summing over the binary cross-entropy loss for each point $\hat{p}_i$:

$$\mathcal{L}_{\text{mil}}^{P} = -\sum_{i=1}^{N_p} \log\left(\varphi_i\right) \tag{3.7}$$

3.4 Experiments

The methods used to test the frameworks and why these were used are explained in this section. The models were trained and evaluated for different datasets and scenarios. One experiment evaluated the annotation effort required for manually annotating points compared to boxes. Finally, a set of models were optimized and used to generate predictions on the CBD dataset. These experiments were designed and conducted to give a holistic perspective on the implications of using this framework for different object detection tasks.

3.4.1 Annotation effort

To assess the impact on the annotation effort, the time it takes to annotate points and bounding boxes, respectively, will be compared. The average time to insert points and bounding boxes for a few images in the CBD dataset was measured. This gives an idea of how the two label types compare in terms of the effort they require. 11 images were chosen for this experiment; one highly populated image with 373 instances, and 10 more sparsely populated images with a total of 327 objects. This way the experiment can show if the annotation speed differs when spending more time on a single image or switching between several.

Creating a point annotation in the labelling software requires pressing two buttons, one for selecting the class and one for placing the point in the image. Creating a bounding box requires three, selecting the class and placing the top-left and bottom-right corners. Apart from the extra click, placing corners correctly to define a box around an object requires more precision than placing a point somewhere on it. All images are labelled and the time it takes is measured.

Results are presented in *4.1 Annotation effort*.

3.4.2 Evaluating model performance

The implications on model performance were assessed in several ways with experiments described below. Mean average precision (mAP) is the most common metric to measure the performance of object detectors and serves as the main performance metric. To get a good perspective, both mAP@50 and mAP@50:95 will be measured. These metrics will be measured for different training cases, with different model versions and changes to the training data. The difference between mAP@50 and mAP@50:95 is that mAP@50:95 looks at an average of several thresholds between 0.5 and 0.95 for the ground truth overlap (IoU) required for a prediction

to be considered correct, whereas mAP@50 only considers the IoU threshold 0.5. This means that mAP@50:95 required more accurate ground truth overlap than mAP@50.

To gain a fair assessment of the point-guided loss suppression model and the MPT framework, they are compared to the semi-supervised case in every experiment. This means that three models are trained for the experiments: A semi-supervised model, one model trained in the PLS framework, and one trained in the MPT framework. The semi-supervised case refers to training a YOLOv8 model on only box labels. The PLS and MPT models are trained on the same data, in the mixed setting of boxes and point labels in each image.

3.4.2.1 Varying levels of box supervision on CBD datset

The fraction of point- and box labels is varied by synthesizing points from available boxes by sampling a point within the box, or removing boxes in the semi-supervised case. By varying this fraction of label types used in training we can get an indication of how well the framework performs given different amounts of labelling - either generated from an open-set detector or manual labelling.

From the fully labelled dataset, copies with 0.5, 1, 2, 5, 10, 30, and 50% box labels were generated for this experiment. The standard YOLOv8 model is used as a baseline, only training on this fraction of box labels. The point-guided loss suppression and point-teaching models also receive the rest of the objects as point labels, sampled from the box labels for those objects.

Another interesting case to look at for the CBD dataset is binary object detection, where the objective is only to detect berries, and not distinguish between their different classes. The performance in this single-class case gives an indication of how well the model would perform if it was used to count berries in an image for example.

The models are all trained with a batch size of 2 until validation performance has not increased for a hundred epochs, and the best model is then evaluated on the test set to produce results. Results for this experiment are presented in subsection 4.2.1.

3.4.2.2 Varying levels of box supervision on MS COCO 2017

By training the model on a commonly used benchmark dataset such as MS COCO 2017, it is possible to make an evaluation of this framework which is more comparable to other works. This enables a close comparison to the original Point-Teaching framework by [Ge et al., 2023], which was trained on the COCO dataset.

The three model versions were trained on the COCO dataset for different fractions of box- and point labels in the same manner as for the CBD dataset. It was trained with a batch size of 32 for a fixed number of 11 epochs. The number of epochs was selected to be close to 40k iterations which the original Point-Teaching framework

was trained on for comparison. A key difference is that in this setting with mixed label types within images, the strong box labels can't be oversampled as in the original where each batch consisted of 16 images with box labels and 16 images with point labels.

The models trained on COCO are only evaluated on the validation set due to a lack of time.

3.4.2.3 Varying size of the CBD dataset

This experiment was made to test whether the models can learn the CBD dataset using less training data than the 143 images in the original training set. The training set with 10% box labels was used, and each model was trained on a quarter, half, and three-quarters of the training data. This is to give some insight into whether or not some models can learn more from less data. If the performance plateaus with less data than in the full training set, the results can also give an indication of whether the small number of images in this dataset is enough. Given the relatively large number of instances in each image, this question is particularly interesting for understanding the CBD dataset.

3.4.2.4 Hyperparameter tuning

Two hyperparameters, λ_1 and λ_2, were used to balance the image- and point-wise MIL loss, respectively. These were to be tuned in a way that optimizes the performance of the model and should be tested individually without impact from the other. This could be achieved by simply setting one hyperparameter to zero, based on 3.1. The values of the thresholds tested in this experiment were the same as those tested in the original Point-Teaching paper by [Ge et al., 2023].

3.4.3 Optimizing confidence threshold

Each prediction the YOLOv8 model makes has an associated confidence score, a value between zero and one that represents how confident the model is in the prediction. YOLOv8 uses a default confidence threshold of 0.25 when making predictions with a trained model. The threshold is used to filter candidate predictions made by the model so that only the predictions that it is most certain of are returned. However, the mAP is calculated over candidate predictions of all confidence scores, regardless of their magnitude. Predictions made by a trained model with the default confidence threshold might not be representative of the performance indicated by the mAP metrics. A model can for example get a high mAP score but only make predictions with confidence scores below 0.25. To use such a model effectively you need to change the confidence threshold or it will not make any predictions. It is unlikely that the optimal threshold is exactly the default 0.25 for any given model, even if they make predictions with confidence scores covering a larger range.

One way of optimizing the confidence threshold is by looking at the F1 score on the validation set for all confidence thresholds. A good F1 score indicates a balance

between finding many objects in the image and not producing many bad predictions. By using the confidence threshold corresponding to the highest F1 score one could improve the predictions produced by a trained model. This optimization was done for all models trained on the CBD dataset with 5% box labels and is exemplified with predictions on a single image from the test set. The F1-confidence curves used for optimizing the threshold were calculated on the validation set. Results are shown in section 4.3.

4

Results

4.1 Annotation effort

Results from the experiment on annotation effort conducted as described in *3.4.1 Annotation effort*.

The time taken for annotating the one image with many objects and the ten images with few objects, with boxes and points, respectively, is presented in table 4.1. When annotating the same image more than once, the labels produced were not completely accurate. The number of annotated objects differed by a few because some objects were missed or wrongly annotated at some point. However, this should not negatively affect the impact of this test assuming missing or mislabeling is similarly likely for points and boxes. Because of the differing number of labels, the annotation effort is presented as the time taken per label.

	Boxes	Points
Highly populated image	6.97s	2.20s
Sparsely populated images	7.19s	1.48s

Table 4.1: The time per annotation given in seconds. The number of berries labeled was nearly the same for the large and the small images, with around 360 annotated berries.

For the image with many objects, the average time taken to annotate boxes and points respectively was 6.97 seconds per box and 2.20 seconds per point. For the images with fewer objects, the time taken was 7.19 seconds per box and 1.48 seconds per point. This shows an average decrease of 5.24 seconds per annotation when annotating with points rather than with boxes. Assuming no preliminary labelling, this would save an estimated 49 hours and 31 minutes on annotating the 34024 objects in the full CBD dataset as an example.

4.2 Model Performance

The performance of the three models will be presented here, along with the hyper-parameter tuning for the models trained with MPT. Results from the training on various fractions of box- and point labels, for both the CBD dataset and the MS COCO 2017 dataset, as well as the effects of changing the size of the CBD dataset are presented here.

4.2.1 CBD dataset

The three models were trained on the CBD dataset with different levels of box labels as described in subsection 3.4.2.1. For the PLS and the MPT frameworks, the remaining objects are labelled with points, while for the semi-supervised case, these are left unlabelled. The resulting performance is visualized in figure 4.1 for multiclass data, and in figure 4.2 for the binary detection case.

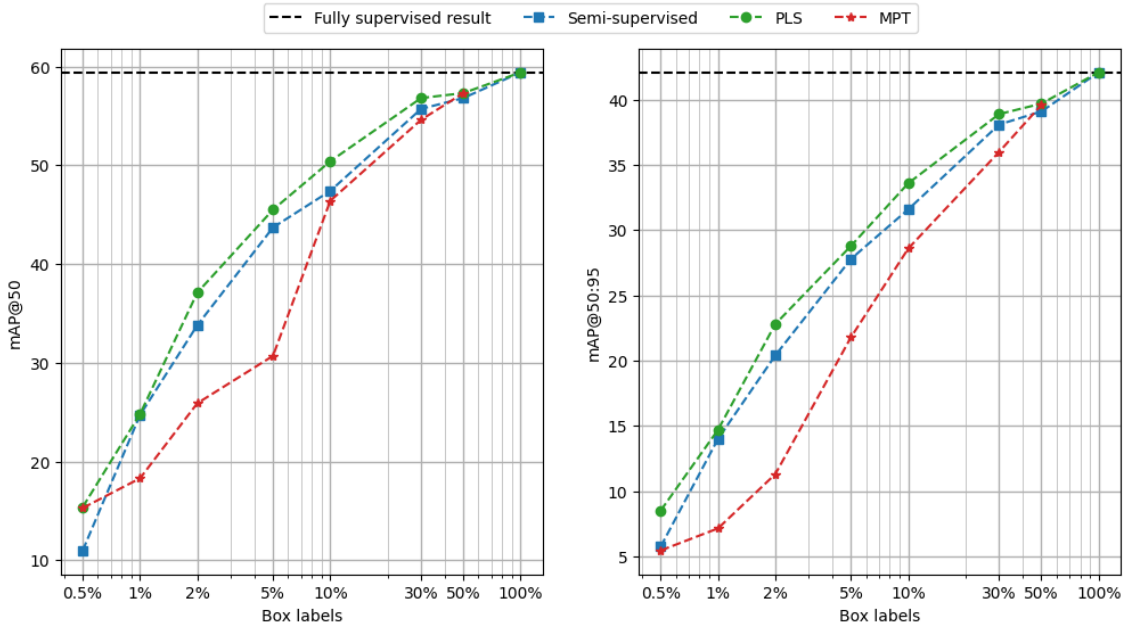


Figure 4.1: The detection performance of each model for multiclass data with different fractions of box labels. The graph on the left shows performance in terms of mAP@50 and the one on the right is mAP@50:95. The PLS framework shows a slight improvement over the semi-supervised case for all fractions of box labels. This means that the model can leverage the point labels for an increase in performance. The MPT framework performs worse than the other models in general but there are scenarios where it performs better. The gap is greater in the mAP@50:95 case. All three models get relatively close to the fully supervised case. Note that PLS and the semi-supervised case at 100% box labels are equivalent to each other. This is termed the fully supervised case.

The results show that using PLS leads to an increase in performance from the semi-supervised case, but only a slight increase. For models trained in the MPT

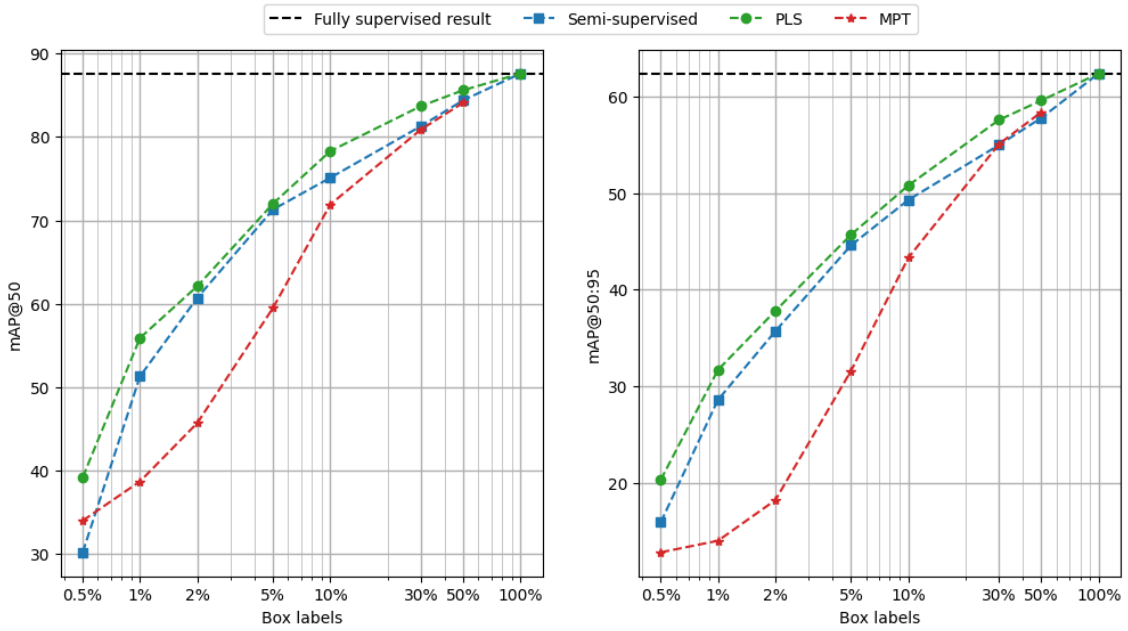


Figure 4.2: Detection performance for models in the binary detection case where all objects are the same class. The trends and results between the models are similar to the multiclass case, except that the scores are generally higher.

framework, the performance is in general worse than the other two methods, even though there are cases where it performed as well as them.

4.2.2 MS COCO 2017 Benchmark

When trained on the COCO dataset, the models produced the results we see in figure 4.3 below. The models trained with PLS performed slightly worse than the semi-supervised models for all levels of supervision. The MPT framework performed worse than both of them, with two exceptions at 0.5% and 1% box labels.

When training these models, an interesting phenomenon occurred for all models and levels of box supervision. In the first few epochs, the loss increased on both validation and training, and the performance deteriorated as well. At around the third epoch, this trend shifted and the models started improving. Two examples of this phenomenon are shown in figure 4.4 below. In some cases, this dip in performance was too significant for the models to improve past within the specified number of iterations. In other cases, the models managed to improve past it with no issues. The implications of this are interesting as the final model is chosen as the one with the best validation performance during training.

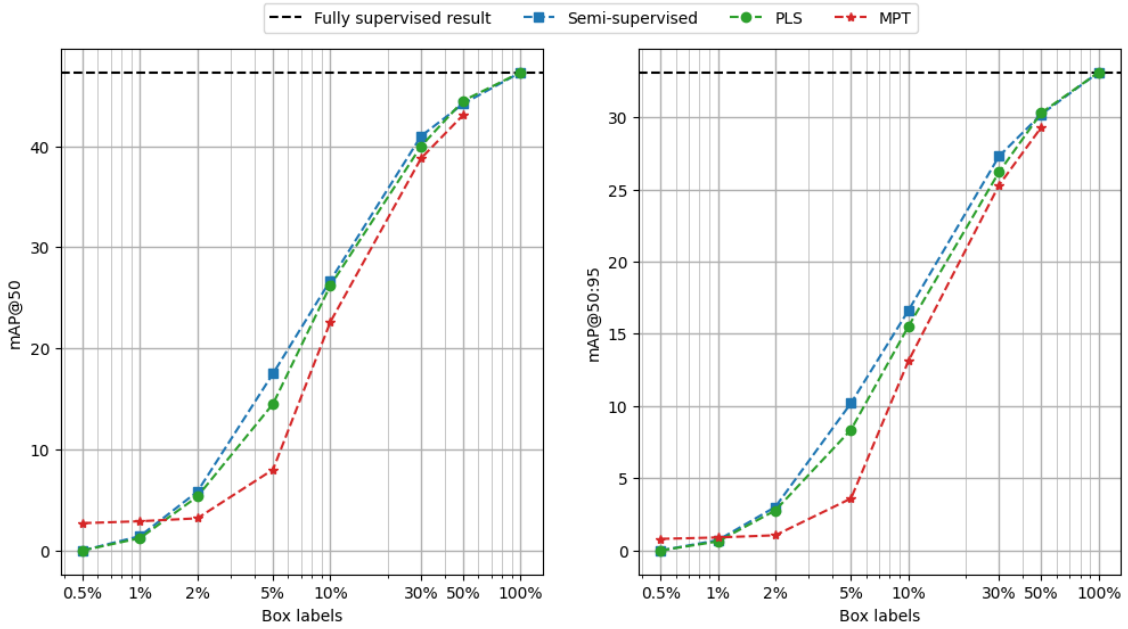
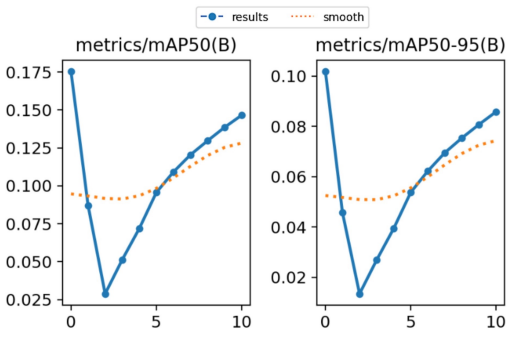
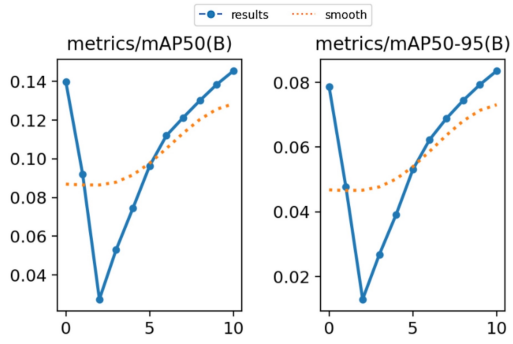


Figure 4.3: The figure shows the performance of the three models on the MS COCO 2017 dataset. The x-axis shows the fraction of box labels used and the y-axis shows the performance measured in mAP. The semi-supervised model generally performs better than models trained with both PLS and MPT, even though the model trained with PLS is very close. The model trained in the MPT framework is better than the other models at 0.5% and 1%.



(a) Validation mAP for the semi-supervised model calculated after each epoch during training. The model failed to improve past the dip from the initial performance within 11 epochs. The model used for evaluation is the resulting model after the first epoch.



(b) Validation mAP for the model trained with PLS calculated after each epoch during training. The model manages to improve past the initial dip in performance, and the resulting model after 11 epochs is used for evaluation.

Figure 4.4: Metrics recorded during training with PLS and the semi-supervised case at 5% box labels.

4.2.3 Efficiency

With the models utilizing points outperforming the semi-supervised models, a natural question to ask is whether this performance increase is worth it based on the time it takes to annotate the dataset. Using the results from section 3.4.1 on annotation effort, the time taken to annotate each dataset can be estimated. The performance of the models can then be compared based on the time taken to annotate the dataset for each model. This gives us an indication of what model gives the most efficient performance increase based on the time required to annotate the dataset for that model. The results for this are provided in figure 4.5 for the CBD dataset, and figure 4.6 below.

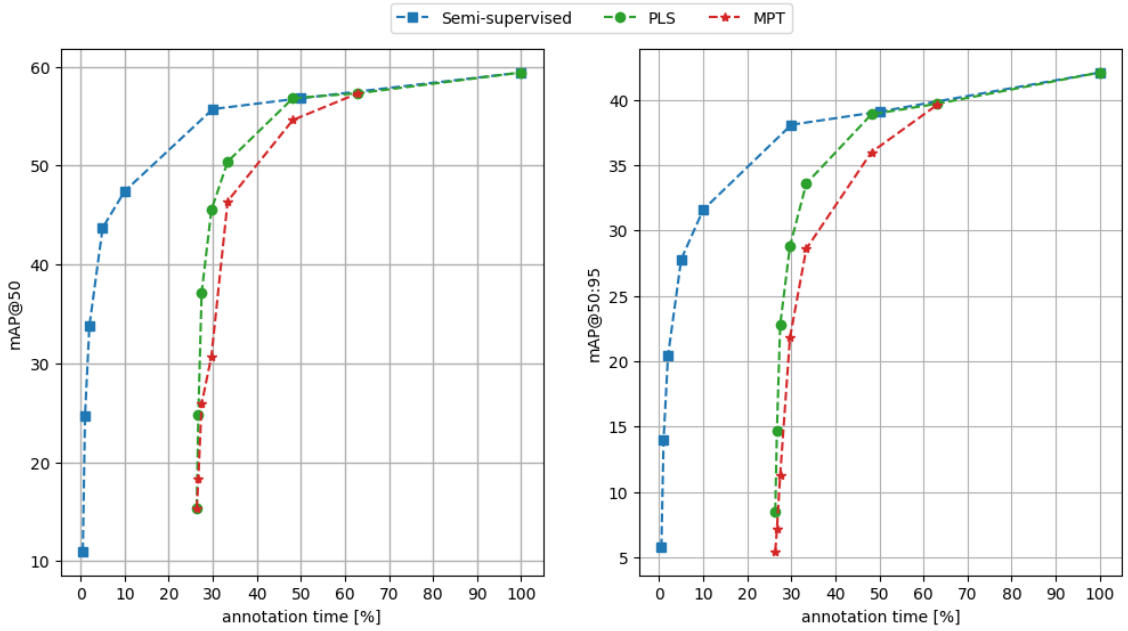


Figure 4.5: Performance for each model on the CBD dataset, with the estimated time for annotation on the x-axis. We see that the model trained with PLS is on par with the semi-supervised model at 50% box labels. As indicated by previous results, the MPT framework is outperformed here as well.

4.2.4 Varying size of the CBD dataset

Experiments were done with the three models, were each of them were trained on different fractions of the CBD dataset. The results of this can be seen in 4.7. Here, the models trained with PLS performed better than both the semi-supervised case and the MPT framework on all sizes of the training set. The semi-supervised model performed better than the model trained with MPT in the mAP@50:95 metric, on all levels. On the mAP@50 metric, the model trained with MPT performed better than the semi-supervised model in the two lowest dataset sizes, while the semi-supervised model was better in the two larger dataset sizes.

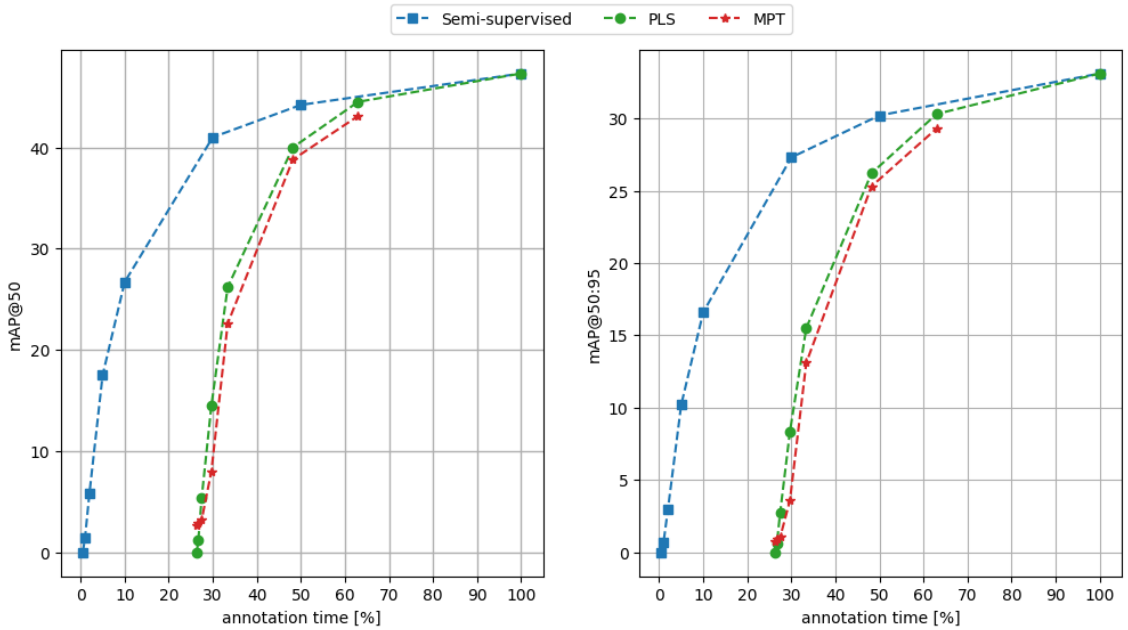


Figure 4.6: Performance of each model on the COCO dataset, with the estimated time for annotation on the x-axis. As the semi-supervised case performed best on this dataset, our models are less efficient as they require more time for annotation and result in lower performance.

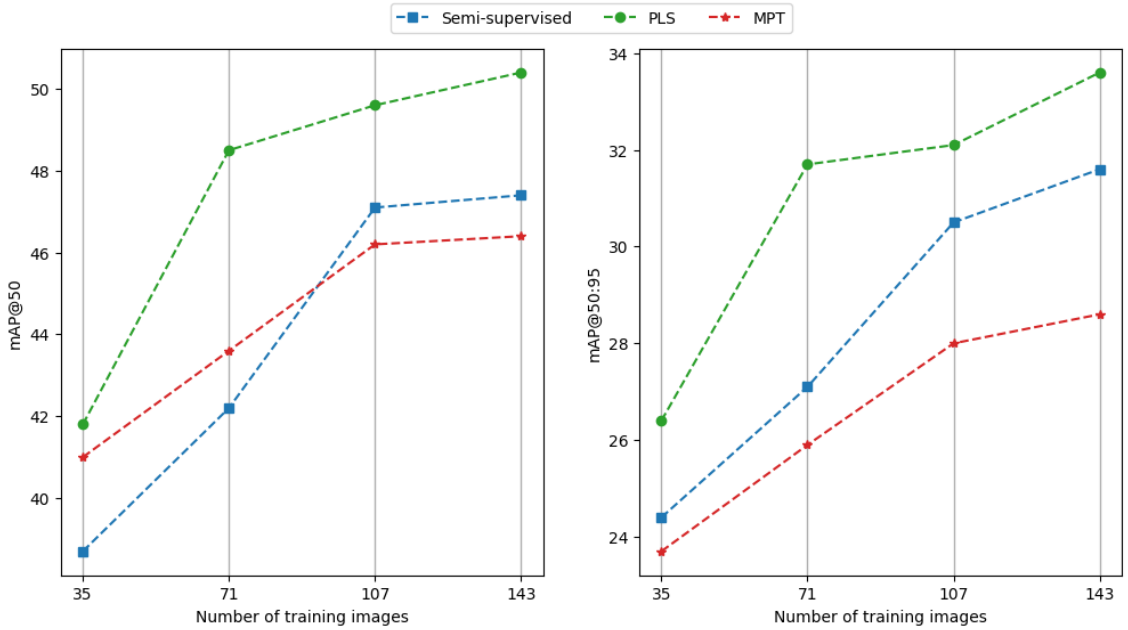


Figure 4.7: Training with all three models on the CBD dataset with various fractions of the training data. All training was done with a box label fraction of 10%. The left plot shows the mAP@50 score, while the right shows the mAP@50:95.

4.2.5 Hyperparameter tuning

The results for the hyperparameter tuning of the image- and point-wise MIL loss gain can be seen in table 4.2a and 4.2b, respectively. From the tables one can see that the best performances were received with $\lambda_1 = 1.5$ and $\lambda_2 = 0.05$.

λ_1	λ_2	mAP@50	mAP@50:95
0.5	0	46.7	27.7
1.0		46.5	28.5
1.5		**47**	**29**

(a) Results from only tuning the image-wise MIL loss gain, λ_1 ,using the CBD dataset with 10% box labels in the training data. This experiment showed that the best performance was when having $\lambda_1 = 1.5$.

λ_1	λ_2	mAP@50	mAP@50:95
0	0.025	47.8	28.8
	0.05	**48.1**	**30.1**
	0.1	47.1	28.6
	0.15	43.7	26.6

(b) Results from only tuning the point-wise MIL loss gain, λ_2, using the CBD dataset with 10% box labels in the training data. This experiment showed that the best performance was when having $\lambda_2 = 0.05$.

4.3 Model predictions

In this section, example predictions made with models trained on 5% box labels are shown, before and after optimizing the confidence threshold. The F1-confidence curves used to optimize the threshold are also presented.

4.3.1 Predictions with default confidence threshold

Using the default confidence threshold of 0.25 from the YOLOv8, the predictions visualized in figure 4.8 were produced. The semi-supervised model produced no predictions despite its relatively high mAP. The model trained in the MPT framework has a lower mAP but manages to make three predictions. The model trained with PLS made the most predictions, with significantly higher confidence scores.

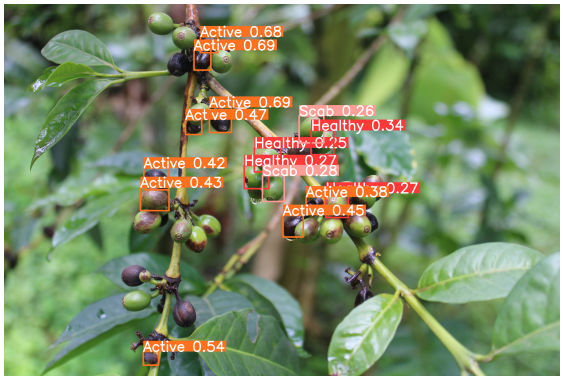
4.3.2 F1 scores

The F1-confidence curves for the models used to create the predictions in section 4.3.1 are presented in figure 4.9. This curve is created by calculating the F1 score for all confidence thresholds. The result shows that the model trained with PLS has the highest maximum F1 score, followed by the model trained with MPT. The semi-supervised model has the lowest maximum F1 score. We also see that these maxima occur at different confidence thresholds. The semi-supervised model has its maximum F1 score at the lowest confidence threshold of the three models, followed by the model trained with PLS, and then the model trained with MPT which had the highest.

mAP@50=41.4

(a) Semi-supervised

mAP@50=44.4

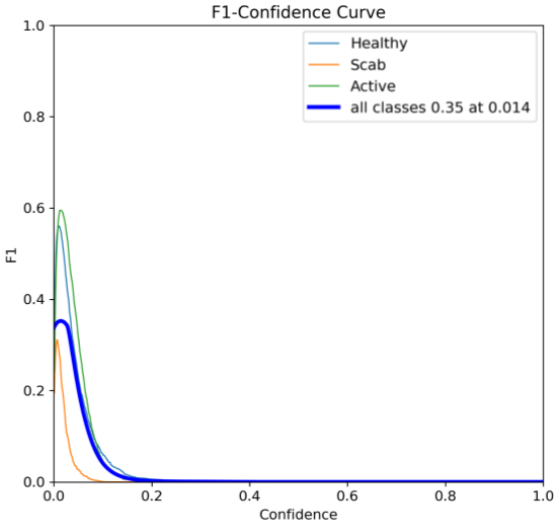


(b) PLS

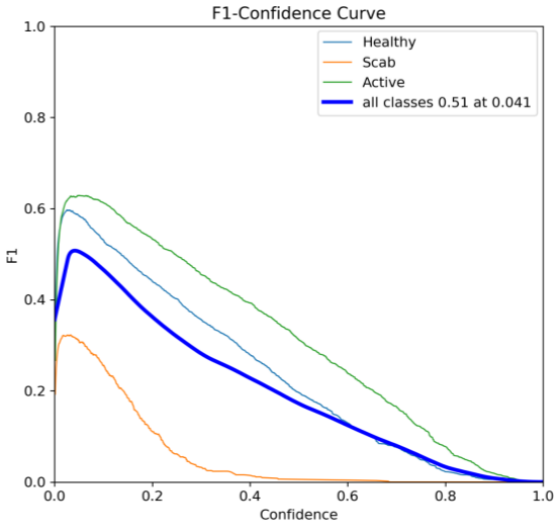
mAP@50=40.4

(c) MPT

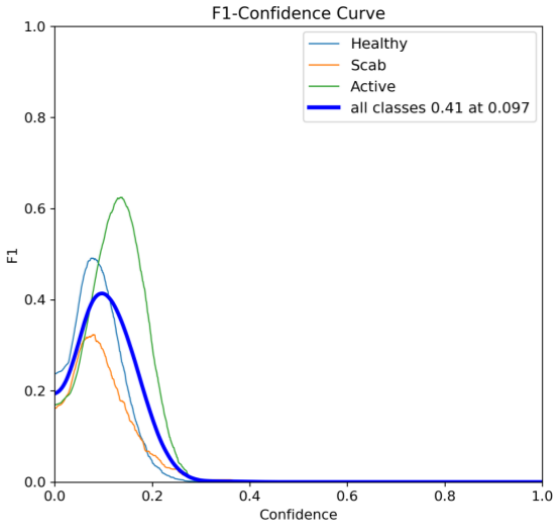
Figure 4.8: The subfigures show the predictions for each model using YOLOv8's default confidence threshold of 0.25. The models were trained with 5% box labels and their respective performance in terms of mAP@50 is presented above each image. All predicted boxes made by each model are shown in their respective image with the class label and corresponding confidence score for that prediction. One could observe that the semi-supervised model did not make any predictions, despite its relatively high mAP. The model trained with MPT performs worse in terms of mAP but produces three predictions. One could also notice that the model trained with PLS made the most predictions and also much higher confidence scores on its predictions than the model trained with MPT.



(a) Semi-supervised



(b) PLS



(c) MPT

Figure 4.9: Each subfigure shows the F1 scores for all confidence thresholds for their corresponding model. The models were trained with 5% box labels and the F1 scores are calculated on the validation set. The F1-confidence curve for the semi-supervised model is shown in subfigure 4.9a. This model had the lowest maximum F1 score of 0.35 at the threshold of 0.014. The F1-confidence curve for the model trained with PLS is shown in subfigure 4.9b. It covers the largest range of confidences and has the highest maximum F1 score of 0.51 at an optimal confidence threshold of 0.041, which is higher than for the semi-supervised model. For the model trained with MPT shown in subfigure 4.9c, one can see that it covers a larger confidence range than the semi-supervised model and has a maximum F1 score of 0.41 at the highest optimal confidence threshold of all three models at 0.097.

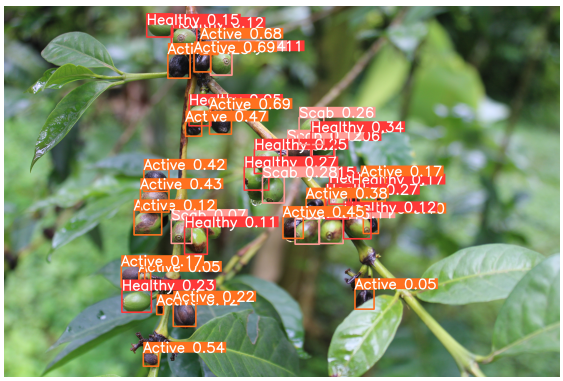
4.3.3 Predictions with optimized confidence threshold

Predictions using the confidence threshold optimized for the model's maximum F1 score are presented in figure 4.10. This shows how the models can make more predictions when the confidence threshold is changed.

mAP@50=41.4

(a) Semi-supervised

mAP@50=44.4



(b) PLS

mAP@50=40.4

(c) MPT

Figure 4.10: The subfigures show the predictions for each model using a confidence threshold optimized for the F1 score. The models are the same as the previous example in section 4.3.1, but with a confidence threshold optimized for the F1 score. All predicted boxes made by each model are shown in their respective image with the class label and corresponding confidence score for that prediction. We see that all three models produce more predictions than with the default threshold. These additional predictions all have confidence scores below the default threshold of 0.25.

5

Discussion

The developments in this project show promising results, but as it stands now these methods are likely not worth using in practice. The PLS method worked well considering its simplicity, utilizing the point labels to get an increase in performance on the CBD dataset. It did not perform as well on the more complex COCO dataset. Models trained in the MPT framework generally performed worse than the semi-supervised models, but we believe the method has potential for improvement.

5.1 Datasets and model performance

The choice of classes and the annotations in the CBD dataset were based on a small amount of research and without significant involvement of experts in the coffee industry. As it is now, the dataset is likely not optimal to use with ML models for tracking or understanding CBD. We annotated all the berries in each image, and their class was decided by whether we thought they showed the symptoms we found in our research (scab lesions or active lesions). Optimally, the dataset would be annotated by experts with proper knowledge in this field and how the disease is usually handled, but this was unfortunately not possible. In conversation with Mpendakazi Agribusiness, it became clear that it would be important to take more symptoms into account for a model to understand the disease properly, such as markings on branches and berries that have fallen off the branch due to infection. However, the applications of object detection models are many. The annotations made for the dataset may be useful for tasks other than only tracking the disease. For example, when tracking or diagnosing the impact of the disease it may be important to keep track of the number of berries in each image. This is a task that models trained on this dataset can do quite well, as seen in the single-class case in figure 4.2 where a fully supervised model reached an mAP@50 of 87.6 which is quite high for object detection.

For the CBD dataset, the performance increase with the PLS method compared to the semi-supervised case is rarely worth the time investment required to annotate the remaining objects with points. As indicated in figure 4.5, it is in most cases more efficient to increase the number of box labels than to provide point labels. There are several variables in the experiments that may result in different outcomes in other cases. The time cost was estimated based on a short test annotating part of the

CBD dataset with only three different classes. An interesting finding in figure 4.5 is that the performance improvement is quite sharp in the beginning and plateaus substantially at 30% box labels. This in combination with the estimated annotation time required leaves very little room for possible improvements gained from utilizing point labels. Figure 4.6 shows the corresponding result on the COCO dataset. Here we see that the difference between the semi-supervised case and the fully supervised upper bound is larger, meaning that there is more room for improvement by utilizing point labels.

The CBD dataset contains many instances of berries in each image, enough to provide good model performance with a small fraction of these as box labels. Especially using the model trained with PLS, we see in figure 4.7 that it achieves better performance than the semi-supervised case with only half as much training data. This suggests it could be useful in applications with especially small datasets. In such a setting, there may be room for more efficiency improvements as well.

The time it takes to label boxes and points respectively is an important variable in determining the efficiency. The experiment carried out in this project is a very rough estimation, and is specific to annotating the CBD dataset. This may not be very representative for other cases, as there are only 3 classes in the dataset and the shapes and sizes of the objects are often quite homogenous. It would be interesting to look at how fast the annotation process is on other datasets as well.

5.2 Mixed label setting

The setting with both strong and weak labels within the same image is the main focus of the developments in this project. While it does enable the use of open-set detectors to efficiently label data, it prevents the use of other smart solutions. The most clear problem that the setting causes is that the strong labels can't be oversampled, which is a common practice when working with unbalanced data. In a normal setting, with images either having only strong labels or only weak labels, the images with strong labels can be sampled more frequently to utilize them to greater effect. The original Point-Teaching framework by [Ge et al., 2023] trained with batches of 16 images with box labels and 16 images with point labels. This way the model always has access to strong labels during training and can make more use of these than if they were sampled uniformly. In the mixed setting, oversampling strong labels can not be done in the same way, and it can therefore be argued that models developed for this setting need to utilize the point labels to a greater extent, which is a more difficult task than relying on box labels.

Although generating box labels with an open-set detector is an important step in utilizing this setting, this was not explored in detail in this project. The models are quite new, and while they gave rise to the idea for this mixed label setting, it was decided that how well this works was dependent on too many variables to test properly for this project. The amount of box labels they are able to generate likely depends heavily on the dataset, and measuring the effort to somehow compare it to

manual annotation is difficult. As they have shown promising performance it was taken for granted that these models can be useful and more efficient than manual annotation from scratch. It would be interesting to see how many box labels these models generate for images in different datasets, and how the required effort to sort through and correct them compares to manual annotation from scratch. Using these models or more accurately simulating their output would give interesting insights into what applications this setting would be useful for, and what fraction of box labels can be expected.

5.3 Confidence

By default, the YOLOv8 model uses a confidence threshold of 0.25 when making a prediction outside of training. A trained model will therefore only output predictions with a confidence score that exceeds this threshold. As exemplified by the predictions and F1-confidence curves shown in section 4.3, some models produced predictions with quite low confidence scores. This problem was generally less significant when the models had more box labels available, as this is more similar to the standard fully supervised case. For the lower levels of box supervision, such as for 5%, we see that the semi-supervised case has especially low confidence levels in its predictions. The confidence scores of predictions cover only a very small range close to zero. This is understandable, as the model would receive a high loss for 95% of its predictions if it is optimal. The very small range of confidence scores means that the robustness of models trained in this mixed semi-supervised setting is questionable.

In contrast, the models trained with PLS and MPT, which uses a mixed weakly semi-supervised data, produce predictions for a greater range of confidence levels. We believe that producing predictions with confidence scores covering more of the available range, makes a model more robust. The results indicate that the model trained with PLS is the most robust, using the entire confidence range when trained with only 5% box labels, although very few predictions have a confidence score close to the maximum. The model trained with MPT produces predictions with a wider range of confidence scores than the semi-supervised model, but is far from utilizing the whole range. Despite producing few predictions over the standard confidence threshold at 0.25, we believe that the larger range this model covers makes it more robust than the semi-supervised model.

By looking at the maximum F1 score on the validation set for each model, and at which confidence threshold this is achieved, we see similar results as with robustness. The semi-supervised model has the lowest F1 score of the three models at 0.35, and it is reached at the lowest confidence threshold at only 0.014. The model trained with PLS achieves its maximum F1 score of 0.51 at a confidence threshold of 0.041. This is the highest F1 score of the three models, which tells us that the PLS method results in the best balance between finding all objects and not making unnecessary predictions in this case. The confidence threshold is still quite low, but significantly better than the semi-supervised model. The model trained in the MPT framework achieves its maximum F1 score of 0.41 at the highest confidence threshold of all

models, at 0.097. It is interesting to note that while this model performs worse than the semi-supervised model in terms of mAP, it produces predictions with higher confidence and with a better balance of precision and recall as indicated by the higher F1 score. This means that the model trained with MPT can be considered better if you want to optimize for a metric such as the F1 score, despite a lower mAP.

The confidence threshold used by the teacher is a crucial factor when training in the MPT framework. In our experiments, the teacher model had the threshold set at the default value of 0.25. When trained on data with fewer box labels, we have seen that very few predictions exceed this confidence threshold. This means that during training, it is likely that the teacher model barely outputs any predictions and that the generated pseudo-labels rely too much on the copy-paste method. Copy-pasted labels are likely less accurate than predictions the teacher could make with an adjusted confidence threshold.

Reflecting on the performance in terms of mAP achieved with the MPT framework, we see some signs of the framework not utilizing the teacher model and relying heavily on the copy-paste method. With few box labels, when the teacher rarely generates predictions, the MPT framework is significantly worse than the alternatives. As we increase the fraction of box labels, the MPT framework is generally on par or at least much closer to it. We believe this is because the teacher model produces predictions with higher confidence so that the system works more closely together, as intended. We can similarly argue that overusing the copy-paste method results in a lack of accuracy in predictions. Comparing performance in mAP@50 and mAP@50:95, the MPT framework is noticeably worse in the mAP@50:95 metric. Since mAP@50:95 places stricter requirements on high ground truth overlap, this tells us that the resulting models produce less accurate boxes.

Changing the confidence threshold used by the teacher could enable the mutual learning process between the student and teacher models that is intended in the MPT framework. This can help for all levels of box supervision but is likely more impactful for lower levels. Throughout the training process, setting the confidence threshold for the teacher model to the optimal confidence threshold according to the F1 score on the validation set would let more predictions through to the point-matching step. The quality of pseudo-labels would likely increase as a result of this, and this would enable the teacher and student models to mutually benefit each other as much as possible.

5.4 Future work

As this mixed setting is new and the models created in this project are quite unrefined due to time constraints, we believe that there is great potential for future development in this area. Generating more diverse and realistic datasets, preferably using pre-trained open-set detectors would be an interesting comparison to our simulated case with a roughly equal fraction of boxes and points in each image.

In a more realistic scenario, objects annotated with points were too difficult for an open-set detector to find. Easily detected objects are more likely found by the open-set detector. Using an open-set detector to generate preliminary box labels for a dataset creates a bias where objects labelled with boxes are more likely to be easily detected. This bias was not accounted for in our experiments.

To improve the PLS method, incorporating the class labels of points and predictions in deciding where to suppress the loss would be interesting. We believe this could help the model learn more complicated distributions and learn more efficiently than it does now. For example, on an image of a truck annotated with a point, the model may predict a good box but classify it incorrectly as a car. As it is now, the model suppresses loss for this case, but we believe it could benefit from the loss in this case. Suppressing the loss only when the class label is correct would be an interesting change to look into.

The MPT framework did not perform very well in terms of mAP, but it may also have the greatest potential for improvement. Firstly, the teacher model is likely not confident enough to produce many predictions with a confidence score over the required threshold of 0.25, especially when training on data with fewer strong ground truth labels. Dynamically changing this threshold so that the teacher can produce more predictions for pseudo-labels would likely improve the exchange between the teacher and the student dramatically. In many cases, the copy-paste method seems to produce the large majority of pseudo-labels. This is not optimal, as copy-pasted labels are likely not very accurate for the object in the image.

Exploring alternatives to the point-guided label copy-paste method would also be interesting. Copying patches of the image together with the label, as in the original point-guided copy-paste suggested by [Ge et al., 2023], could result in better performance despite our concerns about obscured and overlapping objects exemplified in figure 3.6. It would also be interesting to try a pre-trained open-set segmentation model such as SAM [Kirillov et al., 2023] to replace the copy-paste method by using point labels as input to SAM.

Conducting the annotation process of the CBD dataset in close collaboration with experts in handling the disease is an important part of making the CBD dataset and any models trained on it useful. This refers to choosing what classes are of interest, making decisions on how a model would be used in an end product, and annotating the images accordingly. As it is now, we annotated these images based on some short research. After discussing with experts, it became clear that there are important signs of the disease that do not show on the berries themselves that are of equal interest. This can be things like symptoms showing on the branches and signs that berries have fallen off the branches due to the disease.

6

Conclusion

Creating models for object detection to train on a mix of box and point labels in images is a difficult task, but we believe that the setting has potential. The semi-supervised model turned out to be better than we thought in terms of mAP, but the model produced predictions with very low confidence and F1 score. The loss suppression method we propose is a simple solution that provides a performance increase on the CBD dataset, even when the model is provided with only small amounts of training data. We consider this model to be more robust than the semi-supervised model in this setting, as it removes the problem with low confidence. When applied to the more diverse COCO dataset the results were not as impressive, generally falling below the baseline. We believe this is because the loss suppression is agnostic to classes, and that it removes important learning opportunities for the model when there are more classes involved.

The MPT framework did not perform as well as the PLS method. Comparing it to the semi-supervised model, it decreased performance in most cases on both the CBD and COCO datasets. Similarly to the model trained with PLS, the issue with resulting confidence in predictions is mitigated compared to the semi-supervised case. The confidence is still generally lower than for the model trained with PLS. On the COCO dataset, the model trained with MPT was the only one with a nonzero mAP for 0.5% box labels. However, in most cases, it was outperformed by the other models.

For future work, it would be interesting to develop and test the models further. We think the MPT framework in particular has great potential for improvement. Changing the confidence threshold for the teacher when generating pseudo-labels, and trying alternatives to the point-guided label copy-paste are two examples we think could improve the model dramatically. Creating more realistic datasets with varying fractions of boxes and points in each image would also be interesting, preferably involving open-set detectors in the process. This could also provide more insight into the efficiency comparison. Finally, involving experts more closely in the annotation process, as well as choosing the classes for the CBD dataset, would make it more interesting for use in the coffee industry.

Bibliography

[Bilen and Vedaldi, 2016] Bilen, H. and Vedaldi, A. (2016). Weakly supervised deep detection networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[CABI, 2022] CABI (2022). *CABI Compendium*, CABI Compendium.

[Carion et al., 2020] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *Computer Vision – ECCV 2020*, pages 213–229, Cham. Springer International Publishing.

[Eiman and Eickhoff, 2023] Eiman, A. and Eickhoff, N. (2023). YOPO. Code will be made available at https://github.com/AxelEiman/msc-coffee-berry-disease.

[Ge et al., 2023] Ge, Y., Zhou, Q., Wang, X., Shen, C., Wang, Z., and Li, H. (2023). Point-teaching: Weakly semi-supervised object detection with point annotations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):667–675.

[Harman, 1986] Harman, D. W. (1986). An experimental study of factors important in document ranking. In *Proceedings of the 9th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '86, page 186–193, New York, NY, USA. Association for Computing Machinery.

[Jocher et al., 2023] Jocher, G., Chaurasia, A., and Qiu, J. (2023). YOLO by Ultralytics. Open source software available from https://github.com/ultralytics/ultralytics.

[Kirillov et al., 2023] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., and Girshick, R. (2023). Segment anything. *arXiv:2304.02643*.

[Kuhn, 1955] Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

[Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in

context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

[Liu et al., 2023] Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J., and Zhang, L. (2023). Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv:2303.05499v4*.

[Miceli et al., 2020] Miceli, M., Schuessler, M., and Yang, T. (2020). Between subjectivity and imposition: Power dynamics in data annotation for computer vision. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW2).

[Michaelis et al., 2020] Michaelis, C., Bethge, M., and Ecker, A. S. (2020). Closing the generalization gap in one-shot object detection. *arXiv:2011.04267v2*.

[Prihastuti et al., 2009] Prihastuti, H., Mckenzie, E., Hyde, K., Cai, L., H, M., and Hyde, E. (2009). Characterization of colletotrichum species associated with coffee berries in northern thailand. *Fungal Diversity*, 39.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

[Sultana et al., 2020] Sultana, F., Sufian, A., and Dutta, P. (2020). A review of object detection models based on convolutional neural network. *Intelligent computing: image processing based applications*, pages 1–16.

[Tarvainen and Valpola, 2017] Tarvainen, A. and Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[Tkachenko et al., 2022] Tkachenko, M., Malyuk, M., Holmanyuk, A., and Liubimov, N. (2020-2022). Label Studio: Data labeling software. Open source software available from https://github.com/heartexlabs/label-studio.

DEPARTMENT OF PHYSICS
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY