Active Learning Techniques for Annotation Efficiency in Detecting Coffee Berry Disease

Evaluating the Impact of Strong and Weak Labels for Semi-Supervised Object Detection

Emma Amnemyr
emma.amnemyr@gmail.com

Daniel Björklund
karldanielisak@gmail.com

A thesis presented for the degree of
Master of Science in Engineering

Center for Mathematical Sciences
Lund University
Sweden
May 2024

Active Learning Techniques for Annotation Efficiency in Detecting Coffee Berry Disease

Evaluating the Impact of Strong and Weak Labels for Semi-Supervised Object Detection

Authors:
Emma Amnemyr
Daniel Björklund

Supervisors:
Aleksis Pirinen & Olof Mogren (RISE)
Karl Åström (Center for Mathematical Sciences, LTH)

Examinator:
Alexandros Sopasakis (Center for Mathematical Sciences, LTH)

Abstract

Arabica coffee production in Africa has declined significantly over the last half century, partly due to Coffee Berry Disease (CBD), caused by the fungus Colletotrichum kahawae. This disease results in substantial economic losses, estimated at USD 350-500 million annually. Recent advancements in machine learning (ML) and computer vision offer powerful tools for disease detection. However, annotating data for training ML models is both time-consuming and costly. Active Learning (AL) aims to maximize annotation efficiency by strategically selecting data points for annotation, thereby accelerating model performance improvement. This thesis evaluates the impact of utilizing both strong and weak labels in AL for detecting CBD. Initially, an AL framework was implemented, and four query strategies using only strong annotations were developed and evaluated. One of these strategies, ALCU Soft-Rank, showed promise and appeared to outperform the baselines. This strategy was then further developed to determine whether the inclusion of weak labels could enhance the performance. The results indicated that, under the chosen conditions, incorporating weak labels was not beneficial, and the original ALCU Soft-Rank utilizing only strong labels performed best. Further exploration of active learning in this setting, especially using other base models, would be interesting.

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

Acknowledgements

We would like to extend our deepest gratitude to our supervisors Aleksis Pirinen and Olof Mogren at RISE for providing us with excellent guidance during the entirety of the project and for engaging in interesting discussions. We would also like to convey our appreciation to our supervisor Karl Åström at the Center for Mathematical Sciences for his guidance in the project, always providing an insight into the bigger picture. Additionally, many thanks to our examiner Alexandros Sopasakis and our opponents Richard Lindholm, Eric Ihre, and Tom Hagander.

Finally, a big thanks to the other master's thesis students at RISE for providing us with an inspiring environment to work in, whether at the office or wherever in the world we might end up during the day.

Daniel Björklund & Emma Amnemyr
Lund, June 2024

List of Acronyms

AI Artificial Intelligence

AL Active Learning

ALCU Adapted Least Confidence Uncertainty

ANN Artificial Neural Network

AP Average Precision

CBD Coffee Berry Disease

CNN Convolutional Neural Network

ER Entropy Reduction

FN False Negative

FP False Positive

IoU Intersection over Union

LCU Least Confidence Uncertainty

LMU Largest Margin Uncertainty

OD Object Detection

PLS Point-guided Loss Suppression

SMU Smallest Margin Uncertainty

TN True Negative

TP True Positive

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Coffee is a well-traded and valuable commodity on the global market. In 2023, the global market revenue from coffee was valued at USD 453 billion, with an expected yearly continuous growth up to USD 532 billion in 2028 [1]. For developing countries, this is also the second-next most valuable commodity, only surpassed by petroleum. Given this, the economic importance of coffee production cannot be understated, as the income given has great societal impacts on both local households and nationally. In Africa, Arabica coffee production has declined during the last half century, accounting for a 27% share of the global production during the 1970s and dropping to 13% in the 2000s [2]. A main reason for the decrease in market share is the impact of the Coffee Berry Disease (CBD). CBD is an anthracnose disease, meaning it belongs to a group of diseases caused by fungi, specifically the fungus *Colletotrichum kahawae*. The disease creates lesions in the otherwise green berries, affecting the growth of the capsuled coffee bean. The costs of the disease are staggering, with an expected loss of USD 300-500 million annually, representing 2.4-4.5% of Africa's coffee revenue [1]. This includes lost coffee yield as well as the cost of preventive actions, such as chemical pollutants [3].

With the advancements of machine learning (ML) and computer vision in the last decade, powerful tools for detecting disease are available. However, training ML models requires a lot of annotated data. Annotation of data is a time-consuming and consequently expensive process. This is why an effective use of the existing data is crucial. Active Learning (AL) is a technique in ML with the goal of maximizing annotation efficiency. It is done by making decisions on which annotations to annotate and in which order, for the quickest model performance convergence. In object detection problems, bounding box annotations are the most common technique. Building upon a previous master's thesis that researched *Weakly semi-supervised object detection for annotation efficiency*, this master's thesis will build upon their point annotation loss suppression model and investigate AL principles applied to a CBD detection dataset.

1.1 Aim/Motivation

The motivation for this thesis stems from Mpendakazi Agribusiness in Tanzania seeking assistance with machine learning methods to address their challenges related

to CBD. Large quantities of well-labeled data would be needed to create a good model. We seek to investigate whether annotation efficiency can be improved by leveraging active learning techniques. This approach is motivated by the assumption that selecting the appropriate data for annotation and training can enable a model to perform comparably to models trained on larger, unsystematically labeled datasets. To further explore annotation efficiency, we will also consider the integration of weak labels. Weak labels are usually less reliable than strong labels but provide a quicker method of labeling data, thereby offering potential for accelerating the annotation process even more.

To explore different implementations of AL, a dataset consisting of 203 images will be used. Despite its relatively small size, the dataset contains over 30 000 berries.

1.2 Research Questions

- Using active learning techniques, how much can be gained in terms of annotation efficiency when it comes to detecting coffee berry disease?
- Can hierarchical active learning (where one can query for either strong or weak annotation) improve upon a baseline active learning paradigm (where one can only choose between strong or no annotation)?

1.3 Limitations

Limitations in this project include that a fixed object detection (OD) model will be used, in this case YOLOv8. This is due to this thesis building upon a Point Suppression Model which is constructed from this version specifically. As for the Active Learning in itself, we will not make use of pseudo-labels and instead all the annotation will be done by an Oracle. Pseudo-labels refer to labels generated by the model itself through prediction.

1.4 Related work

Most research in the field of AL in computer vision has been focused on image classification. However, more recently, there has been a growing effort to apply it to object detection, a considerably more complex problem that presents its own set of challenges. One problem is that there are often multiple objects in an image. Brust et al. [4] explore how to go from an uncertainty score for each object to a score that represents the whole image. They tried different aggregate functions, such as sum, average and maximum, to get an image score. All the functions performed better than random selection, and the most superior performance was observed from sum. Brust et al. attribute its superiority to its tendency to choose images with a lot of objects in them.

Another problem with OD is the annotation effort, which is why weakly supervised object detection has become more prevalent. Vo et al. [5] brings the two areas together by examining AL in a weakly-supervised setting, where a weak label is an image with a label that provides information about which objects are in the image

but not where. The aforementioned articles all evaluate their method on the Pascal VOC or COCO dataset, which are large datasets common for OD research with around 2.4 and 7.2 objects per image, respectively.

Liang et al. [6] explore AL in an aerial OD setting, which has other challenges than the Pascal VOC and COCO datasets represent. For example, the images contain small and blurry objects that are densely distributed, meaning they are spread out over a large area but still concentrated. These characteristics make most existing AL methods inapplicable. This also makes Liang's et al. problem setting more similar to ours since the coffee berries are often several per image and clustered together. They developed their own method where both the image and object uncertainty are combined to create a final object score. The score is then weighted according to class distribution to get class balancing.

1.5 Ethical considerations

When working with projects relating to AI, a nuanced perception of the technology has to be considered, and the risks have to be asserted before initiation. This includes considering the potential for misuse or whether the technology could bring any other unintended consequences. In general, some important key points to consider are fairness, accuracy, transparency, privacy, security, and societal impacts.

The dataset used in this thesis is a dataset of CBD samples. These are used and given courtesy of the photographer and data collector from Mpendakazi Agribusiness in Tanzania. This data does not include any sensitive information that is to be handled under specific regulations. The annotations in this project were created by non-professionals with the assistance of an AI annotation tool. This means that some labels might have label noise, i.e., not a fully tight bounding box, is misclassified, falsely predicted, or missing in some cases. The dataset also has an unbalanced set of scab instances. However since this project is considering berries, it is deemed to not breach any guidelines in terms of fairness or biases.

As for the explainability of the model, ML models are always difficult to explain when they are using black-box functions. In this project, the well-known YOLOv8 OD model has been used. It is deemed to be robust and reliable.

Throughout the development of the code, it has also been made to be as dynamic as possible, approaching it with good coding practice. Documentation exists throughout the code and a read-me is included too. This is to ensure that continual improvement is possible with the future goal of launching it into production.

High accuracy OD techniques can, however, be used for negative purposes, with a risk of infringing on personal privacy and jeopardizing societal structures through surveillance and social control. This requires consciousness when creating frameworks. Hopefully, this project only inspires future development in the field for beneficial impacts for both society and a more sustainable future.

Chapter 2

Background

In this section, the theory surrounding the project will be introduced. This includes more in-depth details regarding CBD and a general introduction to the topic of ML. Subcategories will also be introduced more in depth, as they will touch upon Object Detection and specific model architectures. This will give a broad foundation for the subsequent section about AL and different query strategies.

2.1 Coffee Berry Disease (CBD)

As previously mentioned in the introduction, the economic effects of CBD have a significant impact not only on individual households but also nationally in developing countries. The costs due to the disease are expected to be around USD 350-500 million yearly [2]. The disease is a type of anthracnose, meaning it belongs to a group of fungal diseases. CBD, in particular, is caused by the fungus called *Colletotrichum kahawae*. The disease shows classical anthracnose symptoms, which are expressed as discoloration and lesions on both the leaf and berry of the affected plant. For the coffee berry plant, the most prominent feature is the lesions in the berry. In this study, the lesions are considered to fall into two different classes: Scab, and Active lesions, which are shown in Figure 2.1 in comparison to the healthy berries.

(a) **(b)** **(c)**

Figure 2.1: CBD affected berries. Healthy berries in sub-figure 2.1a (except for a single active) followed by berries with scab in sub-figure 2.1b and active lesions in sub-figure 2.1c for comparison. Examples taken from our own dataset, courtesy of Mpendakazi Agribusiness in Tanzania.

As seen in the images, a healthy berry has a green or red color depending on the

growth stage, while scab- and active-classified berries have different degrees of discoloration. For the berries with scab lesions, they are characterized by only a few sunken spots, in contrast to those with active lesions, which are characterized by being totally dark and mummified.

2.2 Machine Learning (ML)

Machine Learning (ML) is a subsection of the general area of *Artificial Intelligence* (AI). The main goal of ML is to teach machines how to solve certain tasks with the help of data. It is used to automate predictions and decisions using statistics and to find and explore relationships of varying complexity. One of the building blocks for this is features. They are measurable properties or characteristics of the data that the ML uses to find relationships. ML can also be further divided into sub-categories depending on the circumstances of the used data. ML as a topic is divided into *Supervised Learning* and *Unsupervised Learning*. Supervised learning is the field where data, along with its ground truth, is used for training models. This is the most fundamental ML strategy. Unsupervised learning, however, does not have this ground truth label connected to the data but only exploits the intrinsic data structure. There is also a setting that mixes these two learning methods, called *Semi-supervised Learning*. The difference here is that not all data has a ground truth label, but only a subset. There are, however, different Semi-Supervised Learning methods. They can also be divided even further to weakly semi-supervised Learning. Just as previously, the ”weak” suffix here is also related to the properties of the data, more specifically the way it is annotated. A ”strong” label can be seen as complete and provides all the information needed, while a ”weak” label works as a partial label. For object detection, an example of this would be to use a ”weak” point annotation for an object, which provides the whereabouts of the object but not the exact position and size that a ”strong” bounding box annotation would. This will be further explained in the subsection about object detection. However, since weak labels do not capture the full complexity of the data, it can make it harder for the model to learn the true underlying patterns, which can lead to weaker performance. The reason behind using weak labels instead of strong labels in this project is due to the annotation cost, since points are easier and therefore cheaper to annotate than full boxes.

The amount of data required for training high-quality ML models depends deeply on the complexity and nature of the problem. For a classification problem with many classes, this requires a lot of annotation and consequently also increases the costs. In 2023, the data annotation market was valued at USD 2.90 billion and is expected to have a compound annual growth rate of 28.9% to 2030 [7]. Due to these costs, a demand for effective annotation tools and methods has emerged. One of these methods is the AL framework, which will be explained in the subsection about AL and also be the main focus of this thesis.

If we delve even deeper into ML, we encounter *Deep Learning*, a type of learning inspired by the functioning of the brain and its neurons. Neurons are nerve cells that exist in our brain, responsible for transmitting and receiving sensory inputs through synapses, which are interconnections that tie neurons together. These interconnections are what give the brain its powerful computational capabilities. During the learning process, new connections form or grow stronger, which consequently

improves the memory [8]. This is the inspiration taken into ML, where these artificial neural networks (ANN) can be built in the same way, creating a network of connections between nodes as visualized in Figure 2.2.

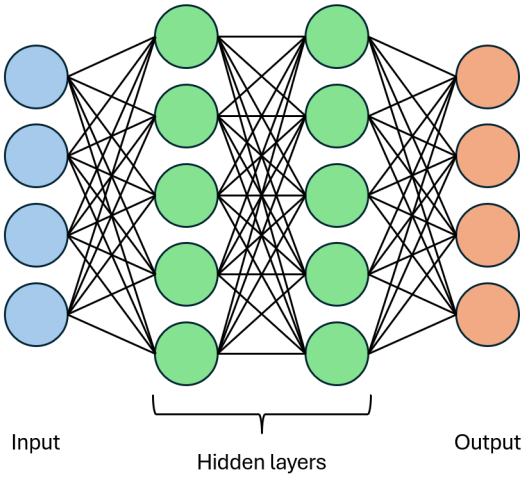


Figure 2.2: Image visualizing an ANN with two hidden layers.

Benefits of deep learning include not having to manually choose features, but having the network choose them itself. This is very advantageous when working with images, as they are often represented by complex relationships. A popular architecture in computer vision is the *Convolutional Neural Network* (CNN). This also builds upon the inspiration of a specific part of the brain, the visual cortex. The CNN has its foundation in the ANN but primarily uses convolutional layers that can handle grid-like data, which means it can preserve the spatial structure of an image. These layers apply filters, also called kernels, to the input image. Each filter can detect specific patterns, like edges, corners and colors. The filter is applied just like when you play the classic game 'Where's Waldo?' You would not be looking pixel by pixel in the image but instead making ageneral sweep of different portions. This is exactly how convolution works, too. A *nxn* kernel sweeps through the image pixels with a set distance and spacing. This can be visualized for a 3x3 setting in Figure 2.3. However, in practice, these types of models can be large and computationally expensive to train. The training process for general ML is an optimization problem, where the goal is to minimize the error or *loss function*. The loss function is used by the learning algorithm during training to evaluate the current predictions compared to the ground truth, update its values, and be guided in the right direction. For large networks with high complexity, this process can be extensive.

An advantageous method when training ML models in general is using already pretrained models. This method is called *transfer-learning*. This means that it is possible to re-use previously trained models on datasets to only fine-tune these networks to specifically fit a problem, saving costs for both computation and annotation as it does not require the same amount of data. It can be done by, for example, re-training only the last layers of a neural network. These open-source networks are often trained on huge datasets, creating a strong foundation for the model.

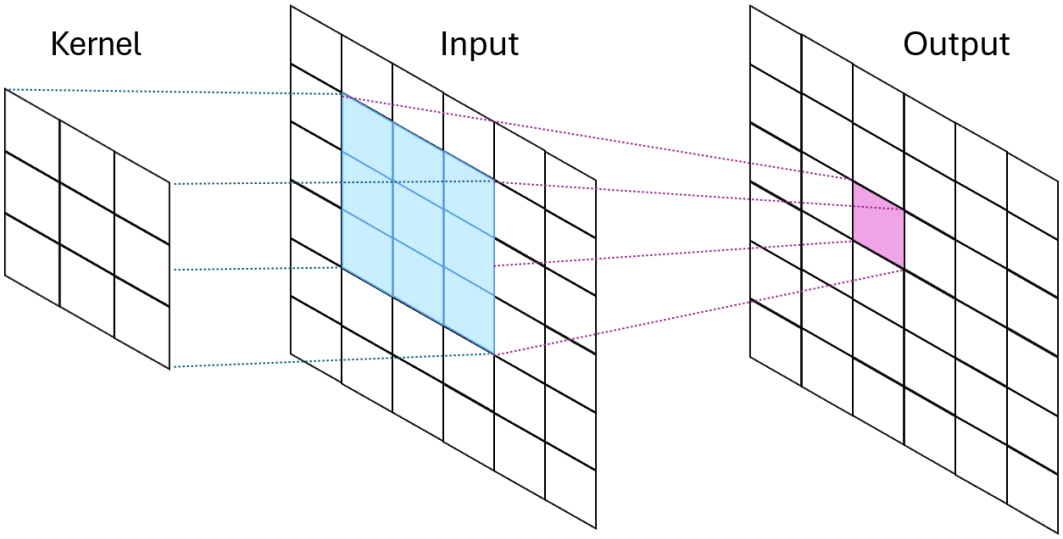


Figure 2.3: Example of a 3x3 Kernel on a grayscale image.

2.3 Object Detection (OD)

Identifying different objects may seem like a trivial problem since we, as humans, can easily and quickly distinguish one object from another from an early age. For the computer, however, the field of object detection (OD) is a challenging problem, highly dependent on the amount and quality of data. The detection itself results in a combination of two properties: a classification and a localization property. The classification gives the class of the object, and localization gives a bounding box around the object. Applied OD can be a very valuable asset in many practical areas, such as security, medical imaging, and autonomous vehicles.

In the process of detecting objects, various neural networks can be used. Two of the main categories of object detectors are two-stage and single-stage detectors. The two-stage detector consists of two modules, where the first module generates candidate object-bounding boxes and the second refines them. This also includes predicting the class label and the precise localization of the box. Single-stage detectors, on the other hand, predict both the class and bounding box in a single pass through the network [9]. In general, two-stage detectors can achieve higher accuracy, while single-stage detectors have a better inference time, making them more applicable for real-time applications. Recent developments have seen a lot of progress in single-stage detectors, and occasionally they achieve higher accuracy than the two-stage detectors [10]. One popular class of object detector models is the single-stage detector called *You Only Look Once* (YOLO).

2.3.1 YOLO

The YOLO algorithm was first introduced in 2015 [11]. The authors treat object detection as a single regression problem, enabling the prediction of the bounding box and class simultaneously. This is also the reason behind the name, since you only look at an image once. YOLO is a deep CNN. The first layers extract features

from the images, and the fully connected layers predict the coordinates and class probabilities.

In overview, the model partitions the input image into an $S \times S$ grid, with the authors using a 7×7 grid for experiments and evaluation [11]. If the center of an object lies within a grid cell, that specific cell is responsible for detecting that object. Each cell then predicts B bounding boxes along with their confidence scores. These confidences indicate both the model's certainty that the box contains an object as well as its confidence in the accuracy of the predicted box. Each grid cell also predicts C class probabilities that are conditional on the cell containing the object [11], with C being the total number of classes. The bounding box confidence and the class probabilities are then combined so that each box gets class-specific confidence scores.

To get into more detail about how YOLO works, the network consists of 24 pretrained convolutional layers and two fully connected layers. The network uses features from the entire image to predict each bounding box. Each predicted bounding box consists of five predictions; x, y, w, h and the confidence. Here, x, y represent the center of the bounding box relative to the grid cell and w, h represent the width and height relative to the entire image size. Consequently, the last layer will have the shape $S \times S \times (B * 5 + C)$ encoding each grid cell's B bounding boxes, their confidence scores as well as the C class probabilities. As described, each grid cell predicts multiple bounding boxes, but during training, only one of these predictions will be responsible for the object. This responsibility is assigned using the highest IoU (which is described more in Section 3.5) with the ground truth. Being "responsible" for an object refers to the loss function only penalizing a box coordinate error if that specific predictor is "responsible" for the ground truth box. This approach leads to each predictor specializing in different things, such as sizes and aspect ratios. When it comes to inference, objects close to a border or larger objects might end up being well localized by multiple grid cells. While this does not critically impact performance, the authors use non-maximal suppression to address this issue [11].

There have been multiple developments of the original YOLO model. Some of the developments build upon each other, while others diverge in other directions. This project has considered YOLOv8 as its base model. The architecture was developed by Ultralytics but does not have an official research article [12]. Even without an official paper, it is one of the latest state-of-the-art models [13]. YOLOv8 used for object detection is pretrained on the COCO dataset and can be used for transfer learning to adapt to a specific task [12].

YOLOv8 directly outputs confidence-scores and not probabilities, the confidences for all classes will not add up to one; $\sum_c^N p_c \neq 1$. This often makes them not directly compatible with the uncertainty measures, which will be introduced in Section 2.4.1. A way to convert the confidences to probabilities and fulfill this requirement is to compute a linear scaling of the confidences for each predicted berry through

$$p_i = \frac{c_i}{\sum_i c_i} \quad , \forall i, \tag{2.1}$$

where p is the probability, c is the confidence, and i is the class index.

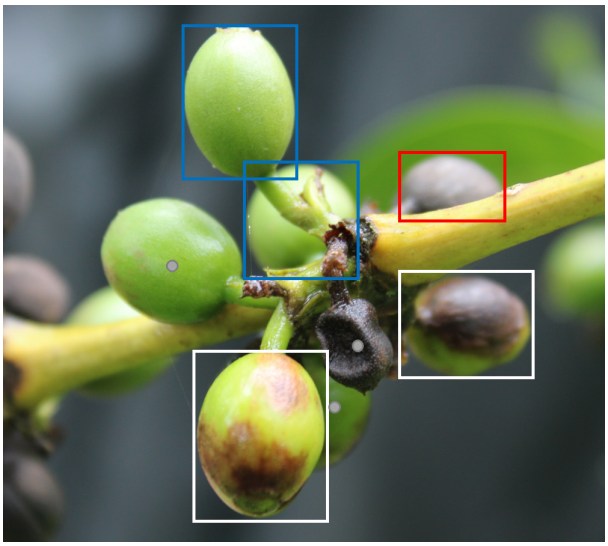


Figure 2.4: Example of the three classes of berries with both box labels and point annotations. Healthy berries is given by blue, scab by white, and active by red. As point labels do not have a corresponding class, they are given by a gray color.

2.3.2 Point-guided Loss Suppression (PLS)

Detection in YOLOv8 is only compatible with box labels, a type of strong label. A previous master's thesis [14] at RISE developed the *Point-guided Loss Suppression* (PLS) framework, which can also handle weak labels. Specifically, it is made to handle point labels, where an object is annotated only with a point within its bounds instead of a fully surrounding box label. The framework is designed to address a key challenge in semi-supervised learning: the absence of labels for certain instances often results in a significant loss. In practice, the PLS works by suppressing the loss function if a prediction encloses a point label, hence the name [14]. Without PLS, if a berry lacks a label during training and a label is predicted there, it generates a loss since the model assumes it should be background. If there is a point label for the berry, PLS instructs the model to ignore anything in that area, thereby ignoring the loss in the surrounding area. No class is assigned to the point.

A point annotation is shown as an example compared to the box label in Figure 2.4. The previous thesis used the PLS framework in combination with the YOLOv8 model [14].

One of the main advantages of PLS, compared to simply using a semi-supervised case is that with PLS, you get predictions with higher confidences. PLS uses the entire confidence range, while the semi-supervised case output prediction with low confidence uses a very small range close to zero, indicating that PLS might be a more robust model. This occurred when the instances not labeled with boxes in the semi-supervised case were labeled with points [14].

2.4 Active Learning (AL)

To train a supervised ML model, you need an annotated dataset to start with. There is often access to a lot of data, but this data usually needs to be labeled, which is

very time-consuming. This is where active learning (AL) can come in. AL is a ML framework that chooses which data to annotate so it will be most useful for the model. The idea is that you can get higher accuracy from the model with less annotated data if you choose the data wisely.

The most common scenario for AL is pool-based. It is when you have access to a large pool of unlabeled data P from the start. There is also a labeled pool T which will be used as the training set for the model. T can either start as empty, called a cold start, or as a smaller initial annotated dataset, called a warm start. This thesis will focus on the latter. The general AL algorithm will start to train the chosen model on T. Then some data from the unlabeled pool will be chosen for annotation. The way this data is chosen is called a query strategy. The queried data will be annotated and added to the training set, and then we will start over again to train the model. This will be repeated until some predetermined condition occurs. In AL, this condition can be that the annotation budget has run out or that we have achieved a certain error bound.

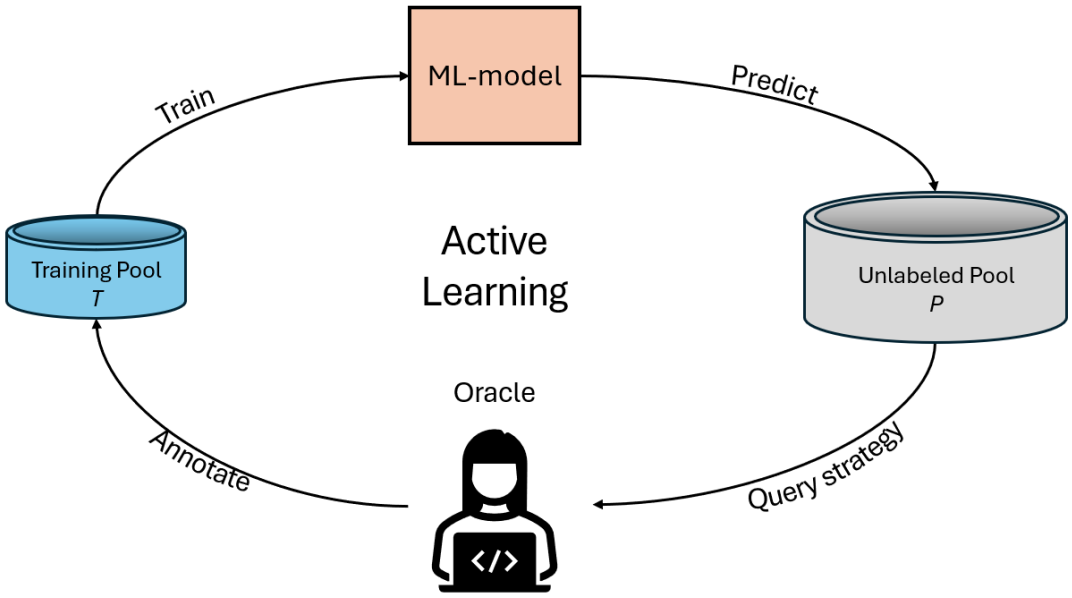


Figure 2.5: Image illustrating the standard Active Learning algorithm.

The main part of AL is the query strategy, which can roughly be divided into two categories: uncertainty-based and diversity-based sampling. The idea of uncertainty-based strategies is that the data that a model is most uncertain about will also be most informative for the model to be trained on. A problem with this type of querying is that it tends to sample data close to the model's decision boundaries, which can cause very similar data to be chosen. Diversity-based strategies instead focus on getting as diverse a dataset as possible. It will solve the issue of choosing very similar samples. However, this can lead to the opposite issue: that samples that are already easy for the model to predict are chosen. Hybrid strategies combine the two former strategies [15].

To compare querying strategies in AL, uniform random sampling is often used as

a baseline. This thesis will mainly explore uncertainty sampling, and some of the common strategies are described below in the following section.

2.4.1 Uncertainty sampling

There are several methodologies when doing uncertainty-based AL. The most common way is to use some kind of uncertainty measure. Some examples that are prevalent in the literature are *Least Confidence Uncertainty* (LCU), *Entropy Reduction* (ER), and Margin Uncertainty types. As the name suggests, LCU sorts by the Least Confident sample. This is done by checking the highest class probability and sorting them in an ascending order. The formula for LCU for one sample is given by

$$\phi_{LC}(x) = 1 - P(y^*|x), \tag{2.2}$$

where y^* is the class with the highest probability for sample x.

Entropy is a way to measure the average amount of information about a random variable's outcomes. Given a sample x, the entropy is given as

$$\phi_{ER}(x) = -\sum_{y} P(y|x) \log P(y|x), \tag{2.3}$$

where y is the class. If the entropy is high, there is high uncertainty, and if the entropy is low, there is low uncertainty. The idea in AL is to calculate the entropy based on the predicted class probabilities and query the instances with the highest entropy first. Given Equation 2.3 these are the instances that are most uncertain about which class they belong to.

Lastly, the method of margin uncertainty can also be used. Here, there are two types; *Smallest Margin Uncertainty* (SMU) and *Largest Margin Uncertainty* (LMU). The former calculates the uncertainty by comparing the first and second highest probabilities through the equation

$$\phi_{SMU}(x) = P(y_1^*|x) - P(y_2^*|x), \tag{2.4}$$

where y_1^* and y_2^* are the two classes with the highest estimated probabilities for sample x. This is then sorted in ascending order. For the LMU, the approach is similar to the SMU, but instead of comparing the first and second highest probabilities, it compares the highest and lowest probabilities. This is given in the equation

$$\phi_{LMU}(x) = P(y_1^*|x) - P(y_{min}|x). \tag{2.5}$$

Both of the margin uncertainty methods are based on the fact that the model is uncertain in its predictions if there is a small difference between the different class probabilities.

Many of these uncertainty measurements are designed for single-point acquisition, where the most uncertain sample is queried one at a time during sampling. However, this is not very applicable to deep learning models, which require large amounts of data for training. In such cases, one sample at a time does not make a significant

impact on the resulting model. Instead, batch acquisition is often preferred, meaning multiple samples are queried simultaneously. The most naive batch acquisition function is to assume independence between samples and to simply pick the top-K most uncertain samples. The assumption of independence will not apply for most datasets and can cause a problem. Imagine if the most uncertain example happens to have a duplicate. Then the top-K acquisition function will choose both of these, which obviously will not be helpful since it will not improve the model further. Some other acquisition functions try to solve this problem, one of which is *Stochastic Batch Acquisition* [16]. This entails a stochastic sampling strategy that takes into account that scores change as new data is acquired, assuming that future scores differ from the current by a perturbation. The noise distribution of this perturbation is modeled as an addition of Gumbel-distributed noise [17] $\epsilon_i \sim \text{Gumbel}(0; \beta^{-1})$. $\beta \geq 0$ is what the authors call, a 'coldness' parameter, and as $\beta \to \infty$, the distribution will converge towards top-K acquisition [16]. Conversely, as $\beta \to 0$, it will converge to a uniform acquisition. The authors ran their main experiments using $\beta = 1$ and showed that it works effectively, but performance could be further improved by tuning β for a particular dataset.

One of the introduced stochastic acquisition variants is called *Soft-Rank Acquisition*. It relies only on the rank order of the scores, meaning it ignores the score values and requires only the relative order of them. The ranking r_i for each score $s(i)$ is created with descending ranks such that $s(r_i) \geq s(r_j)$ for $r_i \leq r_j$, with the smallest rank being 1. An index i is sampled with probability $p_{softrank}(i) \propto r_i^{\beta^{-1}}$ with coldness β. A perturbed 'rank' is then created through

$$s_i^{softrank} := -\log r_i + \epsilon_i, \tag{2.6}$$

where $\epsilon_i \sim \text{Gumbel}(0; \beta^{-1})$, and r_i is the rank as described previously. The authors of [16] show that selecting the top-K samples from $s^{softrank}$ is equivalent to sampling without replacement from the rank distribution $p_{softrank}(i)$. This provides a more nuanced approach to addressing the suboptimal results that might occur from selecting the top-K most uncertain samples.

Chapter 3

Experimental Setup

In this chapter, the experimental result setup will be presented. This includes more in depth information about the dataset and annotation characteristics, as well as the general introduction about the used object detection model and relevant metrics corresponding to the field. Hyperparameters and configurations concerning AL will be described in the relevant section in Chapter 4 and 5, respectively.

3.1 Dataset

A dataset was provided by Mpendakazi Agribusiness in Tanzania. It included 203 images of coffee plants infected with CBD from farms in the Mbeya, Songwe, and Kilimanjaro regions. The images include both healthy and infected berries and are very varied, with different lighting, magnifications, number of berries, and quality. The latter is mainly alluding to the fact that some of the images are out of focus. There are also some instances of photos that were taken of the same branch with small variations.

The labeling of this dataset was part of a previous master's thesis, so it is fully labeled with bounding boxes and classes. The three classes are 'healthy', 'scab lesions' and 'active lesions'. In total, there are 31 323 berries, which means there are, on average, about 154 berries per image. The dataset is quite unbalanced, and the distribution between classes can be seen in Figure 3.1. The dataset has three subsets: training, validation and test set with 143, 30 and 30 images, respectively.

Due to the scarcity of images, another larger dataset was also created from the original one. It was done by splitting each image at the horizontal and vertical center lines, so one image was turned into four tiles. If a box label was located at the split, the box was split as well. To reduce the risk of bad labels, the label was completely removed if less than 15% of the original box was left after the split. The training, validation and test sets were split individually so that tiles from the same image could only exist in the same set. The split was possible due to the high resolution of the images and is the dataset that will be used during all experiments since it gave, in general, higher performance than the other dataset.

When creating point labels, a point was sampled from the bounding box in question. The sampling was done in the same way as in the previous master's thesis [14].

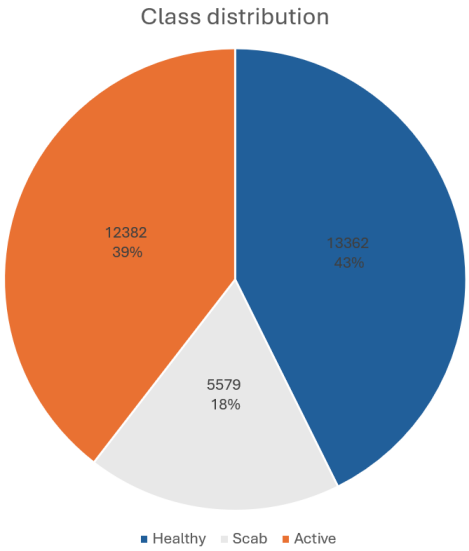


Figure 3.1: A pie chart showing the class distribution of the different classes including the number of instances and corresponding percentages.

Specifically, the points were sampled from a multivariate normal distribution since the dataset primarily consists of berries with a round shape. The mean of this distribution were set to the center coordinates of the box. To ensure most points naturally fall within the box, the variance was selected such that the edges of the box lie three standard deviations away from the mean. Points that ended up outside the edges anyway were clipped to the box edges.

3.2 Annotation costs

As this project focuses on exploring annotation efficiency we have chosen to use time as our measurement, specifically in seconds. Annotation efforts then need to be approximated for both a box and a point label. By assuming that the labels are done in a visual annotation interface that enables the creation of labels, this budget was approximated to 5 s for a box and 0.5 s for a point, as viewed in Table 3.1. This was done by experimenting with annotating different labels on the images and clocking them. Independent of the actual values, the most important factor here is not the absolute values but the relationship between each other, so (s) can be seen as a more general measurement of time. Another important factor was to label background, which is not a class itself but is used to discard predictions that are not valid, for example, predictions where there are no berries. This was approximated at a cost of 1 s and is also included in the table.

Table 3.1: Table showing the annotation costs for different labels per instance.

Label	Cost (s)
Box	5
Point	0,5
Background	1

3.3 Setting

This thesis will also explore active learning in two different settings, a semi-supervised setting and a weakly semi-supervised setting. In the semi-supervised setting, we train an object detection model where images are generally only partly annotated. The annotations that are present are bounding boxes with its corresponding class. In the weakly semi-supervised setting, the images are also only partially annotated, but the existing annotations can be a combination of strong bounding box labels and weak point labels.

3.4 Object Detection Models

To align with the previous master thesis, the YOLOv8 detection model was chosen to apply the active learning principles. YOLOv8 has many different configurations and hyperparameters to be set. As a base model, the YOLOV8n model was used. It is the smallest and quickest model available to both train and run. In addition to version n, there are also four other larger model-versions of YOLOv8 (s, m, l, x), which have been shown to give higher accuracy on other datasets, such as COCO, but do have a higher inference time. When configuring this model, the most important hyperparameters to be set were the number of epochs of each training process, the image size given to the model, and the batch size. Due to time constraints, the behavior of the loss function and mAP scores was evaluated during testing to determine the optimal number of epochs. A plateau was consistently observed before 50 epochs and therefore that number was selected. However, when selecting the final model for the prediction stage, early-stopping was used, i.e., the best-performing model from these 50 epochs was selected. In practice, as an example, this could mean that the 46th epoch model was selected. Secondly, batch size is another important parameter. It controls the trade-off between the quality of the predictions and the training time, but as this project does not take training time into consideration, a low number was selected. This number was decided to be set to 2, since this was the lowest number at which the training time was within a reasonable time frame when conducting our experiments.

Using these general configurations, two separate models were created. One model incorporated only box annotations in the semi-supervised setting, and one model contained the PLS method for the weakly semi-supervised setting. The AL framework was then applied to these two models separately, which is explained further in the following Section 4.2.

3.5 Metrics

There are several different metrics to measure the performance of ML models, as well as OD overall. The most common metric to use in OD is the *mean Average Precision* (mAP). To understand mAP, we first need to introduce some other metrics. To begin with, when working with predictive statistics in general, there are different outcomes that have to be acknowledged. These are True Postives (TP), False Positive (FP), True Negatives (TN) and False Negatives (FN). TP is the number of samples that are correctly predicted to be part of the class, while FP is the number of samples

that are wrongly predicted to be part of the class. Correspondingly, TN represents the number of samples correctly predicted to not be part of the class, and FN represents the number of samples wrongly predicted to not be part of the class. What is a correct prediction then? In OD, the correctness of a prediction is derived from the *Intersection over Union* (IoU), which is the ratio between the area of the overlap between two boxes and the union between the two. This is illustrated for two boxes A and B in Figure 3.2. Visualisations for three different IoU examples are shown Figure 3.3

$$IoU = \frac{A \cap B}{A \cup B} \tag{3.1}$$

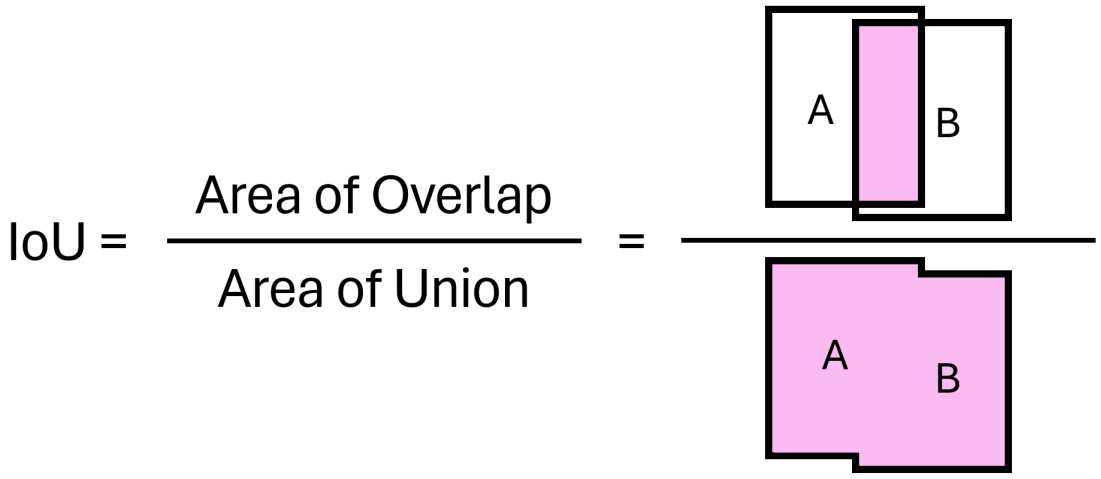


Figure 3.2: The formula for IoU, including a visual representation.

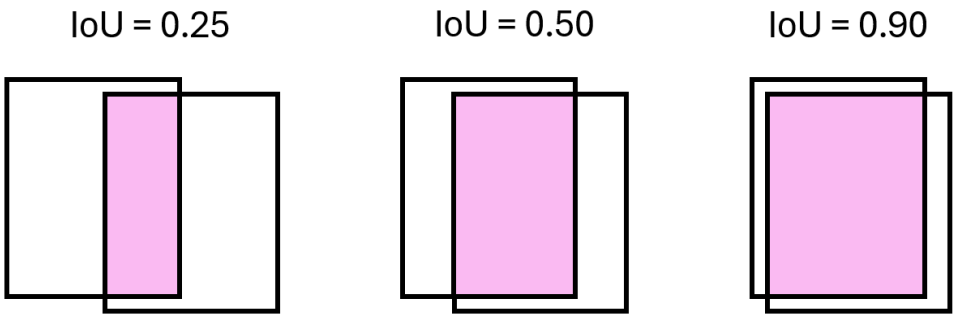


Figure 3.3: Examples of different IoU's.

In practice, for OD, this ratio is calculated using Equation 3.1 between the predicted bounding box and the ground-truth bounding box. For a predicted box to be considered correct, a certain threshold needs to exist. This means that if the IoU is higher than this threshold, the predicted box will be considered a TP. Otherwise,

it is considered a FP. When it comes to object detection, it is not meaningful to talk about TN since that would be all the predictions that did not happen and the number of possible non-object regions is vast and not informative. Therefore, the following metrics do not use TN.

From these metrics, precision and recall can be calculated. Precision measures the percentage of correct predictions and is defined as

$$Precision = \frac{TP}{TP + FP} = \frac{Correct\ Predicitons}{All\ Predictions}. \tag{3.2}$$

Recall instead measures the percentage of how many samples of a class were actually correctly predicted and is defined as

$$Recall = \frac{TP}{TP + FN} = \frac{Correct\ Predicitons}{All\ Ground\ Truths}. \tag{3.3}$$

A good model will have both high precision and high recall, but in practice, it is often a balancing act. The model's predictions will depend on a set confidence threshold. A low threshold will mean more predictions, which often results in a higher recall. Contrarily, a high threshold will often lead to more correct predictions, meaning higher precision. This trade-off can be visualized with a precision-recall curve $p(r)$ which plots the precision against the recall depending on different confidence thresholds. Average Precision (AP) is the area under this curve and is defined as

$$AP = \sum_{r=0}^{1} p(r)dr. \tag{3.4}$$

Equation 3.4 is used to calculate the average precision for each class separately. The mAP is then the mean of all these classes, given by

$$mAP = \frac{1}{n} \sum_{i=1}^{n} AP_i, \tag{3.5}$$

where AP_i is the average precision for a class and n is the number of classes. Depending on which IoU is used, the mAP in Equation 3.5 gives different results. In the field, the standard is to use an IoU of 0.5. This is called mAP50. Another common selection is the mAP50-95, where the IoU is varying between 0.5 and 0.95, capturing both low and high overlap between the inferred ground truth boxes. For this thesis, we have chosen to focus solely on mAP50.

Precision and recall can also be used to calculate another metric that is used to measure the performance of a ML model. The metric is the F1-score and it is the harmonic mean of the precision and recall defined as

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{3.6}$$

Chapter 4

Active Learning in a Semi-Supervised setting

In this chapter, AL in a semi-supervised setting will be presented. An initial exploration phase will be introduced first, followed by the developed active learning framework and the specific query strategies used. The experiments that were conducted will then be given along with their corresponding results.

4.1 Initial Exploration

To start off, we focused on the semi-supervised setting with only ”strong” bounding box labels. Subsequently, given the results of the initial literature study, both basic and more advanced methods were identified for our purpose. These basic methods were chosen because of their occurrence in the literature and the more advanced due to the intrinsic properties of our dataset. Initially, a general full-image-based AL framework was implemented within the given codebase from the previous master's thesis at RISE, making the model compatible with PLS for later usage. The basic methods mentioned in the background in Section 2.4.1 were the uncertainty measures LCU , ER, SMU, and LMU. These were the methods based on uncertainty, which used the probabilities of the predictions to calculate the entropy, the least confidence, or highest margin uncertainties. To make these compatible with YOLOv8 confidence output, the linear scaling of Equation 2.1 was used. An adapted LCU, which we will simply call Adapted Least Confidence Uncertainty (ALCU), was also tested, where the output of YOLOv8 was used directly without scaling. After a phase of exploratory testing of these different methods, the results suggested that the ALCU had the best performance and was decided to be the target strategy to advance with further.

Moreover, the AL framework was changed to query for a batch of individual instances instead of images. The change was made because, although the dataset contained a small number of images, it consisted of a considerable number of individual objects. By adopting this approach, we aimed to amplify the potential of query strategies by providing more options.

It was suspected that if the queried instances were to be spread out between images,

the model might perform worse due to more unlabeled versus labeled parts of an image. To address this, a simple image constraint was created and evaluated. A maximum threshold N was established for the number of new images to be labeled during each iteration of the active learning process. Under the constraint, once the limit is reached, the oracle can keep labeling any instances within all previously labeled images, including both the newly labeled images and the images already in the training pool T. Some experiments were done on different values of N, such as 50, 75, 100 and 125. It showed that it did in fact affect the performance. We observed negative impacts when N was either too large or too small. As long as it was within a moderate range, the specific number did not seem to matter. N was decided to be set to 75.

Consequently, three approaches inspired by the literature study and exploration were created. These methods were ALCU Soft-Rank, Boundary, and Mix. These will be explained in more detail in Section 4.3.

4.2 Framework

The AL framework developed for CBD detection in a semi-supervised setting was created as a warm-start pool-based framework. It was based on the general active learning algorithm described in Section 2.4. To get the initial training pool T, images were randomly chosen and fully labeled until we reached 900 instances, which was about 5% of the berries in the complete training set. A whole image was fully labeled to be moved to the initial pool, hence, the size might slightly exceed 900. For each active learning iteration, the object detection model was trained on the training pool T with the configurations described in Section 3.4 above. For each iteration, it was retrained from the regular YOLOv8 pre-trained weights. The trained model was then evaluated on the validation set, which provided an mAP score as well as an F1-confidence curve with different F1-scores depending on the confidence threshold. This curve was used to optimize the predictions on the unlabeled set P by setting the confidence threshold for the prediction model to where the F1-score is maximized, which, as mentioned in Section 3.5, represents a good trade-off between precision and recall. A query strategy was then used to decide in which order the predictions should be prioritized for annotation. The strategies are described in Section 4.3. The predictions were then iterated through one by one in the determined order. First, it was checked to see if that specific predicted bounding box was not already labeled, in which case we moved on to the next. Otherwise, it was given to the oracle to be labeled. If the predicted bounding box and a ground truth berry had an $IOU \geq 0.5$, the correct class label and bounding box would be provided, and if the $IOU < 0.5$, the box would be considered background. The labeling would continue until the annotation budget ran out for that particular active learning iteration. According to the general active learning algorithm, as explained in Section 2.4, this was the point where all the newly labeled images would typically be moved to T. However, in our case, they were copied to T while still remaining in P. The reason for this adjustment was because it was in a semi-supervised setting, where the images were likely to only be partially labeled. Therefore, the choice was made to retain them in P to potentially acquire more labels if the query strategy suggested doing so. After the new images and labels were added to T, the process moves on to the next active

learning iteration. The pseudocode for the framework can be read in Algorithm 1.

Algorithm 1 Implemented Active Learning Algorithm

Require: Dataset D, Query strategy q, number of iterations N, list of annotation budgets $B = [B_1, B_2, ..., B_N]$, warm start size k
Ensure: Improved model θ for detection

```
Initialize model θ
Initialize labeled dataset T ← ∅, unlabeled dataset P ← D
while Instances in T < k do
    Select data point from P for annotation
    Label selected data point and add it to P
    Remove selected data point from P
end while
Predict instances on P using model θ
for n = 1,2,...,N do
    while B_n > 0 do
        Query an instance from P using Q
        Label queried instance
        Copy the data point containing the instance and add it to T
    end while
    Retrain model θ using annotated data in T
    Evaluate model θ at budget B_n
end for
```

4.3 Query Strategies

The framework utilizes a specific strategy to query predictions for annotation. Below, we describe the developed query strategies for the semi-supervised setting, along with the baselines used for evaluation.

As for most AL experiments, a uniform random query strategy was used as a baseline as this is considered to be the most trivial strategy. This is to show that there is actual value in using the query strategies. We will refer to this strategy simply as Random. Since an image limit in the AL algorithm can restrict the number of images copied from the unlabeled pool to the training pool in each iteration, as explained in Section 4.2, another random baseline with the same limit was also implemented, referred to as Random Limit. This is to make sure that the strategies themselves had an effect on the performance and not only the image limitation.

As was described in Section 4.1, the first most basic strategy that was implemented was ALCU with an image limit of 75. It queries the most uncertain predicted bounding boxes while making sure that no more than 75 images are added to the training pool T in each iteration. Due to the effectiveness of the image limitation of 75, all other developed strategies, which will be described below, will also incorporate the same setting.

Since ALCU is a completely uncertainty-based AL strategy, we also tried adding some diversity to it by creating a mix of ALCU and random, mainly for the first iterations. For the first iteration, 80% of the budget was spent on random instances

and the remaining 20% on instances chosen by ALCU. This quota was then decreased by 20% each iteration until it reached 0% where it then was fixed. This strategy will be referred to as Mix.

ALCU is created for single-point acquisition, but the framework will query multiple instances, so we took this into account by implementing the stochastic batch acquisition function Soft-Rank, from Equation 2.6, to be combined with ALCU. It was adapted to work with ALCU by switching the ranking order so that $s(r_i) \leq s(r_j)$ for $r_i \leq r_j$ since a lower confidence means a higher uncertainty.

As the YOLO architecture was not really adaptable towards visualization and deeper analysis, another independent image encoder was also incorporated to give more of an insight in the AL-process and to build upon. By using transfer learning, the Huggingface CLIP vit-batch-32 model was used to encode every predicted bounding box and to extract its embeddings as this was not given directly in YOLO. This made it possible for us to understand the underlying data instances better and gave us tools to build more dynamic query strategies with regard to uncertainty and diversity. Firstly, the embedding space was visualized to understand the data more thoroughly. This was done by picking a subset of the instances of the data and plotting them in this space. As CLIP returns a vector representation of 512 dimensions for each sample, t-SNE was used to lower the dimension and making it possible to visualize in 2D. A subset of 250 points is shown in Figure 4.1.

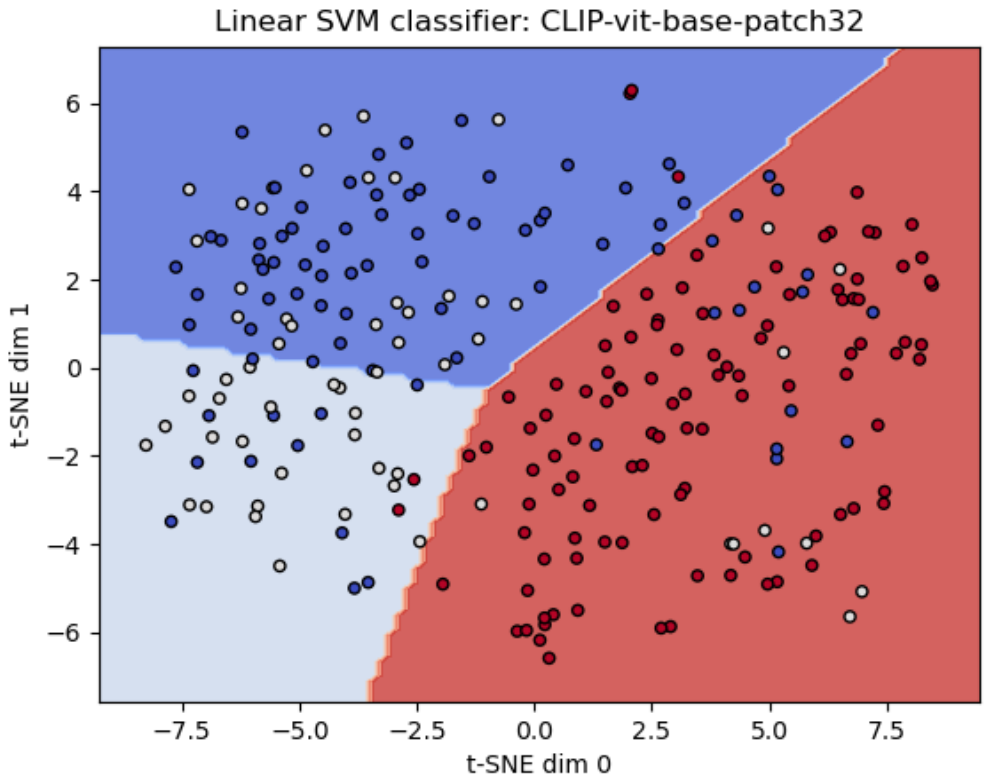


Figure 4.1: Linear SVM classifier on berry instances embedded with OpenAI's CLIP vit-base-patch32 encoder transformed to a t-SNE visualization in two dimensions. Healthy is represented by blue points, scab by white, and active lesions as red.

With this visualization, it seemed that the model had the most difficulties deciding whether a berry was healthy or a scab. This is expected since these are very similar visually. By using a linear SVM classifier for the complete embeddings, boundaries between these clusters were computed for each AL iteration, as were their individual distances from the closest boundary. To find a balanced sampling technique, both uncertain and diverse samples was chosen. This was done by selecting points depending on their distance from the boundary. This was computed for the 512 dimension embedding. Instead of only sampling the points which were closest to the boundary, a mixed sampling technique was created. This was done by taking three percentage intervals of the total amount of points depending on their distance. These were chosen to be 0-10%, 40-60%, and 90-100% from the boundary, which in practice means the top 10% points closest to the boundary, the top 10% furthest from the boundary, and 20% of the points which is around the median of the distances. The resulting points from these intervals was then shuffled and put first in order to be sampled. As for the remaining points outside of these intervals, they were also shuffled but put secondary in the order. Since this strategy is based on decision boundaries, it will be referred to as Boundary.

4.4 Experiments

The semi-supervised setting has been evaluated with four different query strategies, which are the methods ALCU, ALCU Soft-Rank, Boundary and Mix. The metric for evaluation of the performance was selected to be mAP50. The AL-progression was initialized with 5% of the total berries, or 900 berries, in T at budget $= 0$ seconds. The next iteration was chosen at 2500 seconds with the reasoning that it corresponded to about 450 berries labeled with bounding boxes, or another 2.5% of the total amount of berries in the training set. This number was not exact since it depends on the ratio between berries and the background being queried. 2500 seconds was kept as the budget interval for four iterations, where the steps then resulted in: 2500, 5000, 7500 and 10 000 seconds. The next two steps were chosen at 20 000 and 45 000 seconds, corresponding to about 25% and 50% of all instances, respectively. In addition to these query strategies, two baselines of random with and without image limit, Random and Random Limit, were also run.

All of the query strategies above, including the baselines, have been included in the same graph shown in Figure 4.2, with a dotted line to represent the fully supervised result. To ensure a higher certainty of the predictions, every strategy has been run ten times with different seeds, and the result that is shown is the mean of these ten. This experiment has been evaluated on the test set.

The standard deviations of these results are shown in Table 4.1, where for every method and budget insertion, there is a corresponding variation in its value. The best result for each budget is given by a bold number, which is represented by the highest mean of the ten seeds for that column. This is not considering the possible variation of each number, however. The mAP50 for each specific class across every method is also visualized in Figure 4.3.

The method with the best performance for the semi-supervised setting is the ALCU Soft-Rank method, which has the overall highest mean for four of the six possi-

ble budget points. Compared to the random baselines, ALCU and Mix also show promise of beating the random limit baseline at times.

The average gain of the ALCU Soft-Rank method in comparison to the baselines are $+1,9$ mAP50 for Random, and $+0,73$ mAP50 for the Random Limit.

Some notable points are that ALCU Soft-Rank at a budget of 5000 s outperforms Random at 7500 s. Additionally, ALCU Soft-Rank achieves the same mAP50 at a budget of 7500 s as Random Limit does at 10 000 s. In both cases, this results in saving 2500 seconds for the same or better performance.

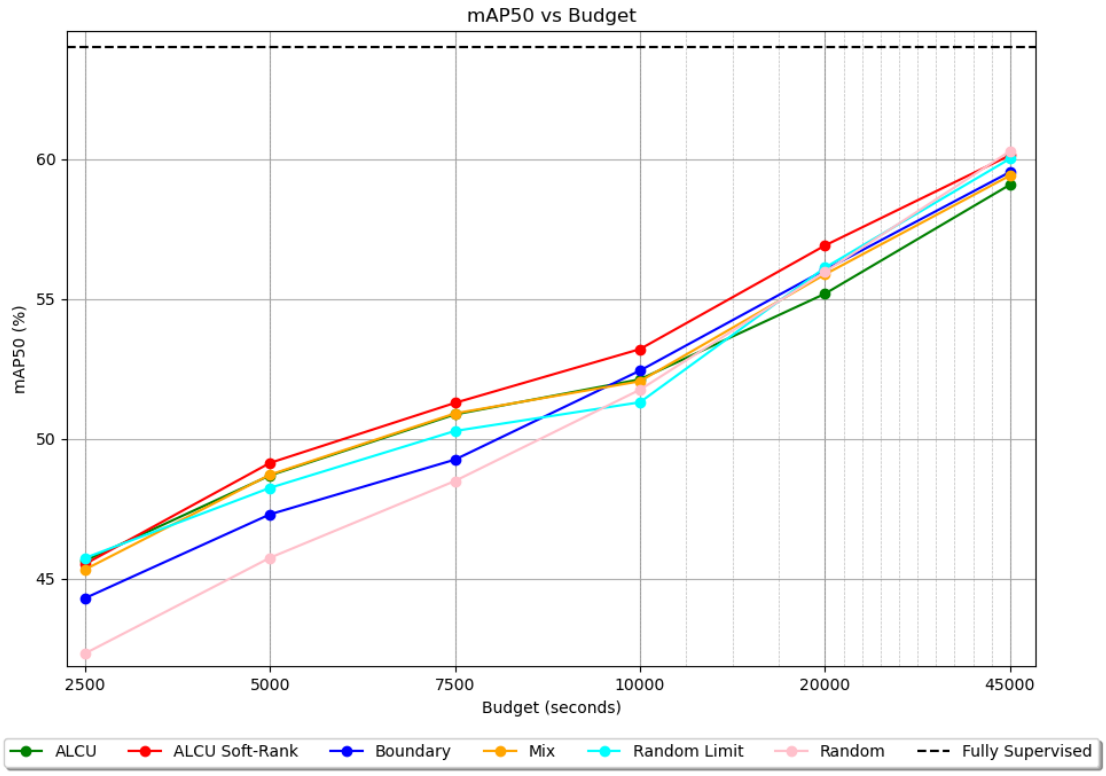
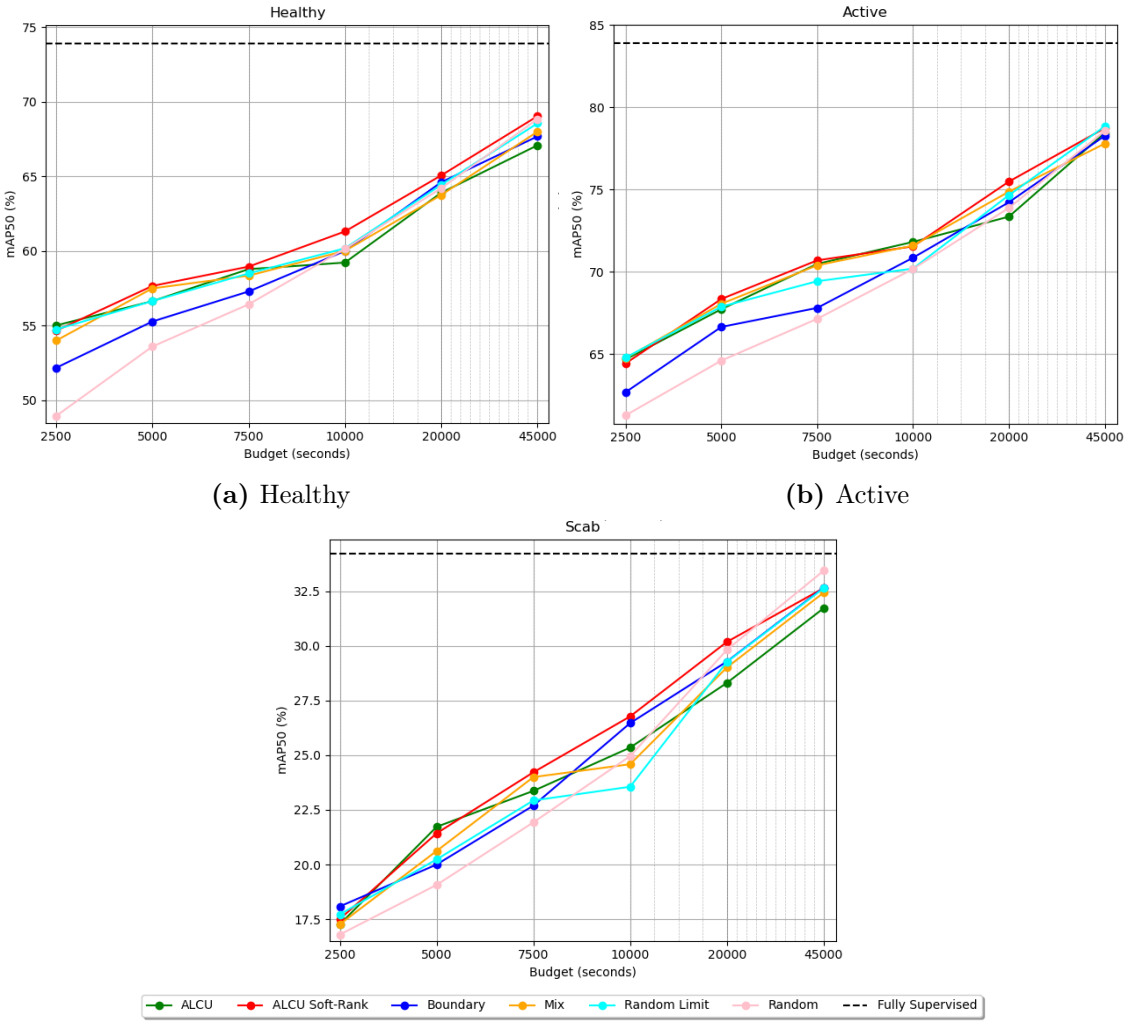


Figure 4.2: Figure shows the mAP50 score in percentage on the y-axis, and budget in seconds on the x-axis when evaluated on the semi-supervised setting for the test-set. Green line represents the ALCU method using only confidences from YOLO, red shows the previous ALCU method but with a Soft-Rank selection and not single-point acquisition, blue shows the boundary method using an independent embedding for computing the confidences based on different distances from the boundary, orange shows the method using both a part randomness and confidence mix decreasing for each learning iteration. These are then compared to the turquoise and pink lines which is showing the random query strategy including an image limit of 75 and a fully random strategy respectively. A black dotted line is also included to show the result when training the model fully supervised with all of the available data.

Table 4.1: The different query methods for AL in a semi-supervised setting (only box labels) and its mAP50 performance using different amount of labeling budget. As this is a warm-start pool initialization, 0 s represents 5% of the labelled data, incrementing 2,5% for each step until 10 000 s which represents 15%. Two additional points at 20 000 s and 45 000 s also represent 25% and 50%. The numbers in bold indicate the highest average mAP for each budget level, not taking the standard deviation into account. The methods has been run on ten seeds.

Method	**mAP50 on portion of budget**						
	0	**2500**	**5000**	**7500**	**10000**	**20000**	**45000**
Random	33.1±7.6	42.3±2.0	45.8±1.2	48.5±2.0	51.8±1.3	56.0±1.2	**60.3**±1.0
Random Limit	33.1±7.6	**45.7**±1.7	48.3±1.6	50.3±1.2	51.3±1.7	56.1±1.3	60.0±0.7
ALCU	33.1±7.6	45.6±1.5	48.7±1.0	50.9±1.3	52.1±1.3	55.2±1.1	59.1±0.8
ALCU Soft-Rank	33.1±7.6	45.5±1.5	**49.1**±1.3	**51.3**±1.5	**53.2**±1.0	**56.9**±1.2	60.1±0.8
Mix	33.1±7.6	45.3±1.4	48.7±1.4	50.9±1.5	52.1±1.2	55.9±0.8	59.4±0.8
Boundary	33.1±7.6	44.3±2.4	47.3±1.5	49.3±2.2	52.4±1.4	56.0±1.4	59.5±0.8



(a) Healthy

(b) Active

(c) Scab

Figure 4.3: This figure shows the class performance differences of Healthy, Active, and Scab for the mAP score in percentage on the y-axis, and budget in seconds on the x-axis.

Since the dataset is quite unbalanced, as shown in Figure 2.1, we also wanted to see

if any of the query strategies would affect the class balance. Therefore, the class labels are tallied for each AL iteration and strategy. The average percentage for each class during each budget stage of the AL framework is presented in Table 4.2. The numbers presented are the averages of ten seeds.

Table 4.2: Table showing the class balance in percentage for different query methods, and different budgets. Healthy is represented by H, Scab by S, and Active by A.

Method	**Budget and Class**																	
	2500			**5000**			**7500**			**10000**			**20000**			**45000**		
	H	**S**	**A**	**H**	**S**	**A**	**H**	**S**	**A**	**H**	**S**	**A**	**H**	**S**	**A**	**H**	**S**	**A**
Random	38	22	40	38	23	39	38	23	39	39	23	38	39	23	38	39	22	39
Random Limit	39	22	39	39	22	39	40	22	38	39	22	39	40	22	38	39	22	39
ALCU	39	22	39	40	22	38	40	23	37	40	23	37	41	24	35	38	22	40
ALCU Soft-Rank	40	21	39	40	22	38	40	23	37	40	23	37	40	23	37	40	22	38
Mix	39	22	39	39	22	39	40	22	38	40	22	38	40	23	37	40	23	37
Boundary	39	22	39	39	22	39	38	22	40	39	22	39	40	23	38	40	22	38

Chapter 5

Active Learning in a Weakly Semi-Supervised setting

This chapter will begin by introducing the initial exploration, which was done given the results from the previous chapter for the semi-supervised setting. Here, the specific framework and query strategies will also be introduced, which have been adapted to use point labels in addition to box labels. Lastly, the experiments and corresponding results will be shown.

5.1 Initial Exploration

Firstly, the weakly semi-supervised setting was evaluated. The goal of a weakly semi-supervised setting was to see if it could perform better than a semi-supervised setting with an equivalent budget. To see if annotating berries with points instead of boxes could ever be beneficial, a few experiments were made where the points were considered to be free. A few different box-point ratios were tried and it was concluded that different query strategies using a 95/5 split, where 95% of the loop budget was spent on boxes and 5% was spent on points, were explored. Even though it might sound like quite a big difference, it was important to consider that points were ten times cheaper, as seen in Table 3.1, which means there were only approximately double the number of boxes as there were points. The potential of this split is visualized in the in Section 5.4. After the split was decided, a few different query strategies as well as labeling strategies were explored, which are described in more detail in Sections 5.2 and 5.3.

5.2 Framework

To adapt the existing AL framework to a weakly semi-supervised setting with point labels, the annotation budget during each loop was divided into two parts: one allocated for bounding box labels and the other for point labels. Two extensions of the framework were developed for the point annotation process. In both extensions, a point label could be replaced by a box label if suggested by the query strategy. Furthermore, only predictions for images already in the training pool could be queried.

This decision was made to prevent images in the training pool with only point labels and no box labels.

The first extension is a direct continuation of the labeling of bounding boxes described previously. Once the budget allocated for bounding box labels ran out, the same labeling process would continue, but with point labels instead. However, due to the cost ratio between different labels, it was suspected that a significant portion of the point labeling budget might be allocated to background points. To address this issue, the second extension was developed.

In the second extension, each image was partitioned into an $X \times X$ grid. Depending on the query strategy, a score was calculated for each grid cell based on the predictions within it. The cells are then sorted based on these scores, and entire grid cells were queried for point labeling in order of their scores. As for the grid annotation cost, if there were instances to be labeled within the grid, the annotation cost was calculated by multiplying the number of point labels by the point cost in Table 3.1. To simplify matters, we retained the background cost from Table 3.1 if there were no additional instances to be labeled in the grid cell or if the grid cell was entirely background. In reality, this would naturally vary depending on the chosen grid size.

5.3 Query Strategies

As PLS is an extension of the semi-supervised model, the same query strategies can be reused for the box labels and further developed for point labels. We have chosen to focus solely on the best-performing query strategy in the semi-supervised setting, which is Soft-Rank ALCU. Regular semi-supervised Soft-Rank ALCU will also be used as a sort of baseline that the following implemented strategies aim to outperform. As described in Section 4.2, two different extensions were created for the point labeling. For the first labeling system, where points are annotated in the same manner as boxes, there are two implemented strategies.

Firstly, the most simple adaptation was to extend Soft-Rank ALCU to use points as well, where the predictions will be queried in the same order for both point and box labeling. Though, the box budget will be used first so the most uncertain samples get box labels. After it runs out, the queried predictions will continue in the same manner, but the oracle will label them with points instead. This strategy will be referred to as Instance ALCU Soft-Rank since we query the object for point annotation instance by instance.

A second strategy was to simply choose the points randomly while the box label was chosen with Soft-Rank ALCU. Corresponding to the aforementioned strategy, this strategy will be referred to as Instance Random. The strategy will both work as its own strategy as well as like a baseline for Instance Soft-Rank ALCU.

As for the second point labeling extension described in Section 4.2, there are three more strategies. As a reminder, the framework extension would query a grid cell of predetermined size and label it as a whole to potentially save costs associated with background annotation.

To get a score for a grid cell, we chose average as the aggregate function. Taking the average can diminish resolution and uncertain predictions may be overshadowed

by highly certain ones, potentially compromising the overall assessment. Because of this, some simple experiments with one seed were done where the average uncertainty was calculated over all the predictions, the 50% most uncertain, the 25% most uncertain, the 10% most uncertain as well as having the score being represented by the single most uncertain prediction in the cell. These were queried in the order of lowest to highest score, meaning the most uncertain to the least until the point budget ran out. The strategy of using the average of the 25% most uncertain predictions in a grid cell as the score appeared to yield the best performance, thus it was adopted as an official strategy together with using Soft-Rank ALCU for the box labels, and it is referred to as Grid Min Average.

The opposite was also implemented, referred to as Grid 25 Max Average, so that the score was calculated using the 25% most certain predictions, instead of uncertain, for each cell. The strategy was then to query the highest or most certain score first. As the PLS suppressed the loss function where point labels were present, the underlying idea with this strategy was that the loss for certain points might be greater compared to uncertain points. Thus, the model could potentially gain more by suppressing the loss of certain points.

Lastly, the grid cells were queried randomly for annotations, referred to as Grid Random. Again, this method was used both as its own strategy and as a baseline for the effectiveness of the grid-cell sampling strategy.

5.4 Experiments

To first explore the potential of ALCU Soft-Rank if you add point labels, we started with the following experiment. Initially, boxes were selected according to the method described previously. Once the box budget was exhausted, all unlabeled instances in the training set were labeled with points. The box budget was set to 95% of the previously used budget steps: 2500, 5000, 7500, 10 000, 20 000 and 45 000 seconds, reflecting the 95/5 split between box and point labeling. For this experiment, we assigned the cost of zero to points to estimate the potential gain with this setting. The results, which have been evaluated on the validation set, are shown in Figure 5.1.

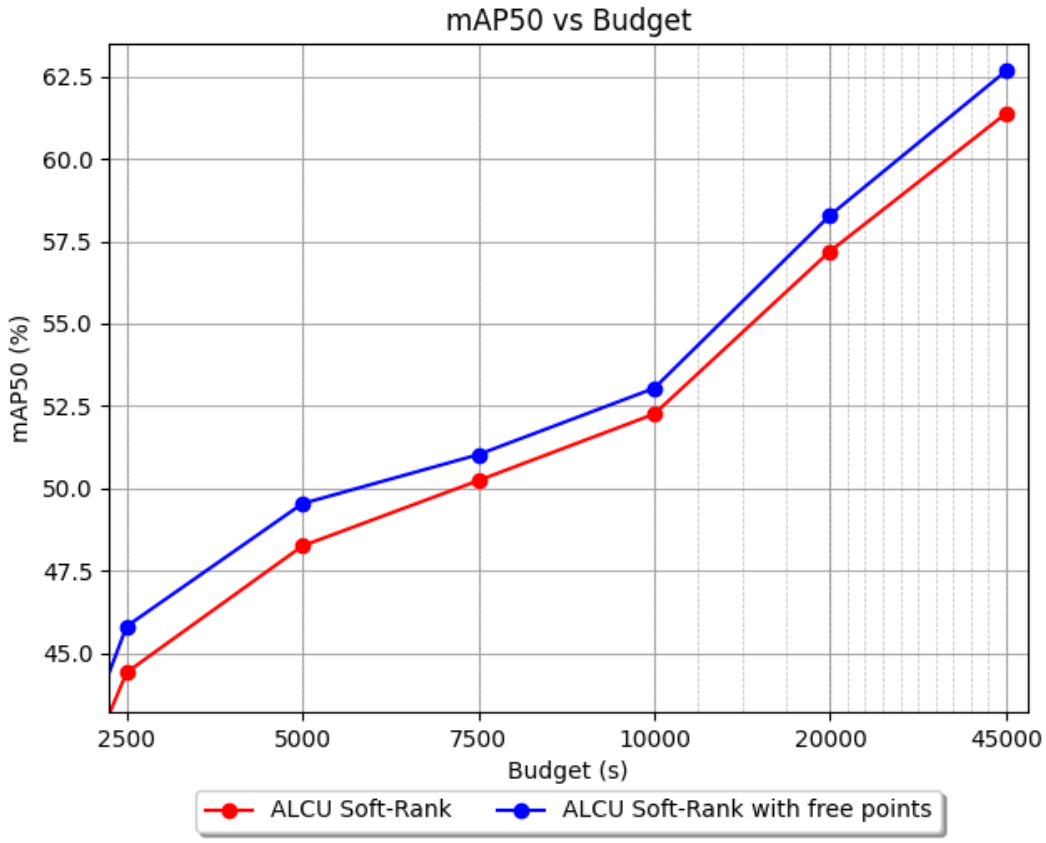


Figure 5.1: A comparison of performance between the ALCU Soft-Rank model with only boxes compared to the ALCU Soft-Rank where free points is also added.

To evaluate the five different query strategies in a weakly semi-supervised setting, we conducted the following experiment. We aimed to keep the experiment as similar to the semi-supervised experiment as possible to ensure an easy and fair comparison. This involved setting the loop budget initially to 2500 seconds, although this no longer corresponded to about 450 berries due to the differing costs for points. Again, this budget interval was maintained until 10 000 seconds, where it was increased, which means that the budgets used were 2500, 5000, 7500, 10 000, 20 000 and 45 000 seconds. To again account for the randomness we have in our strategies, the framework was run with ten different seeds for each strategy, consistent with the same seeds that were used for the semi-supervised experiment. The results of this experiment were evaluated on the test set and are visualized in Figure 5.2 as well as presented in Table 5.1 along with their standard deviations.

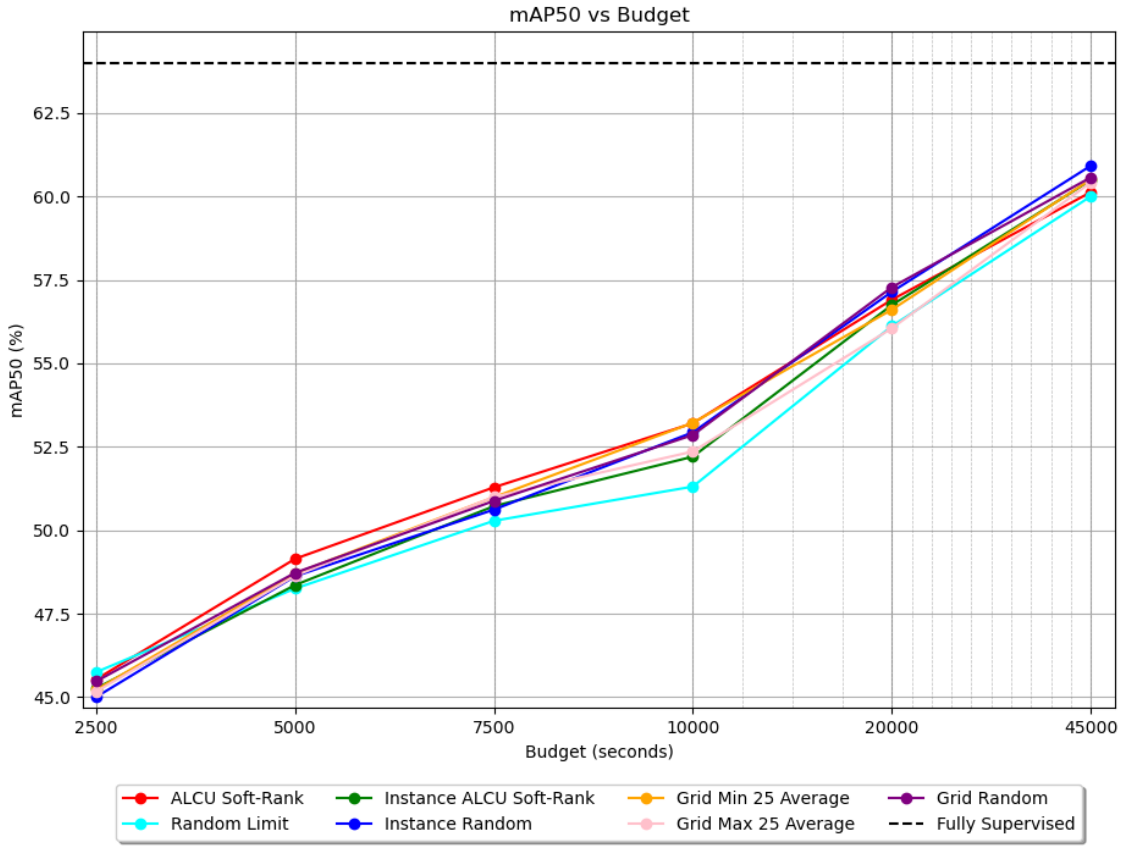


Figure 5.2: Figure shows the mAP50 score in percentage on the y-axis, and budget in seconds on the x-axis when evaluated on the weakly semi-supervised setting for the test-set. Red and turquoise lines works as baselines and represent the ALCU Soft-Rank and Random Limit in the semi-supervised setting which were also included in the corresponding Figure 4.2. The remaining lines split the budget between boxes and points by 95/5% and use ALCU Soft-Rank for box selection, but differs in point selection. The Instance ALCU Soft-Rank and Instance Random query one instance at the time for point labeling, and are represented by the green and blue lines. The rest of the strategies use 2×2 grid querying. These are Grid Min 25 Average, Grid Max 25 Averige, and Grid Random and are represented by the orange, pink, and purple line. A black dotted line is also included to show the result when training the model fully supervised with all of the available data.

Table 5.1: Table showing the different query methods for AL and its mAP50 performance using different amount of labeling budget for both a semi-supervised (only box labels), and weakly semi-supervised setting (both box labels and point labels). The numbers in bold indicate the highest average mAP for each budget level, not taking the standard deviation into account. The methods has been run on ten seeds.

Method	**mAP50 on portion of budget**						
	0	**2500**	**5000**	**7500**	**10000**	**20000**	**45000**
SS							
ALCU Soft-Rank	33.1±7.6	45.5±1.5	**49.1**±1.3	**51.3**±1.5	53.2±1.0	56.9±1.2	60.1±0.8
Random Limit	33.1±7.6	45.7±1.7	48.3±1.6	50.3±1.2	51.3±1.7	56.1±1.3	60.0±0.7
Weakly SS							
Instance ALCU Soft-Rank	33.1±7.6	45.3±1.7	48.4±1.2	50.7±1.7	52.2±1.2	56.7±1.3	60.5±0.7
Instance Random	33.1±7.6	44.0±1.4	48.6±1.1	50.6±1.4	52.9±1.2	57.3±1.1	**60.9**±0.6
Grid Min 25 Average	33.1±7.6	45.2±1.3	48.7±1.2	51.0±1.2	53.2±1.1	56.6±0.8	60.5±0.6
Grid Max 25 Average	33.1±7.6	45.2±1.3	48.6±1.3	51.0±1.1	52.4±1.1	56.1±0.7	60.4±0.3
Grid Random	33.1±7.6	45.5±1.2	48.7±2.1	50.9±0.8	52.9±1.2	57.3±1.1	60.6±0.5

To clarify the performance differences between the regular ALCU Soft-Rank and the five developed strategies for the weakly semi-supervised setting, the average mAP gain was calculated and is shown in Table 5.2. From the table, we can see that the weakly semi-supervised strategy Grid Random's performance comes close to ALCU Soft-Rank but does not surpass it.

To verify whether the second point labeling extension strategy with grid cells worked as intended, i.e., saving costs associated with background annotation, the point annotations will be counted in each AL iteration. Subsequently, the average of all the seeds for each query strategy is computed and presented in Table 5.3.

Table 5.2: The average mAP gains between semi-supervised ALCU Soft-Rank and five different strategies in the weakly semi-supervised setting.

Method	**Average mAP gain**
Instance ALCU Soft-Rank	-0.38
Instance Random	-0.30
Grid Min 25 Average	-0.15
Grid Max 25 Average	-0.4
Grid Random	-0.03

Table 5.3: Table shows the number of point labels for different methods, at different budgets.

Method	**Number of point labels at Budget**					
	2500	**5000**	**7500**	**10000**	**20000**	**45000**
Instance ALCU Soft-Rank	157	179	185	174	755	2103
Instance Random	180	213	211	207	842	2216
Grid Min 25 Average	265	265	247	248	1010	2493
Grid Max 25 Average	257	251	260	247	994	2494
Grid Random	255	267	266	261	994	2462

Chapter 6

Setting Comparison

In this chapter, the additional experiments aimed at obtaining a more nuanced result between the semi-supervised setting and the weakly semi-supervised setting will be presented. These experiments will include results regarding the F1-score performance metric as well as the prediction evolution after each AL iteration. For comparison, the experiments will be conducted using the best-performing models from each setting, namely ALCU Soft-Rank for the former setting and Grid Random for the latter setting. Random Limit will also be included as a baseline.

6.1 Experiments

One of the main advantages of PLS as given in the previous master's thesis, mentioned in Section 2.3.2, was that it computes predictions with confidences covering the entire confidence range. In their case, however, all remaining instances were labeled with a point. This is not valid for us. To assess whether this still applies when there are fewer point labels, the subsequent experiment was done. For each AL iteration, the F1-confidence curve is calculated on the validation set. The curves are averaged across the seeds for each strategy, and the results are shown in Figure 6.1.

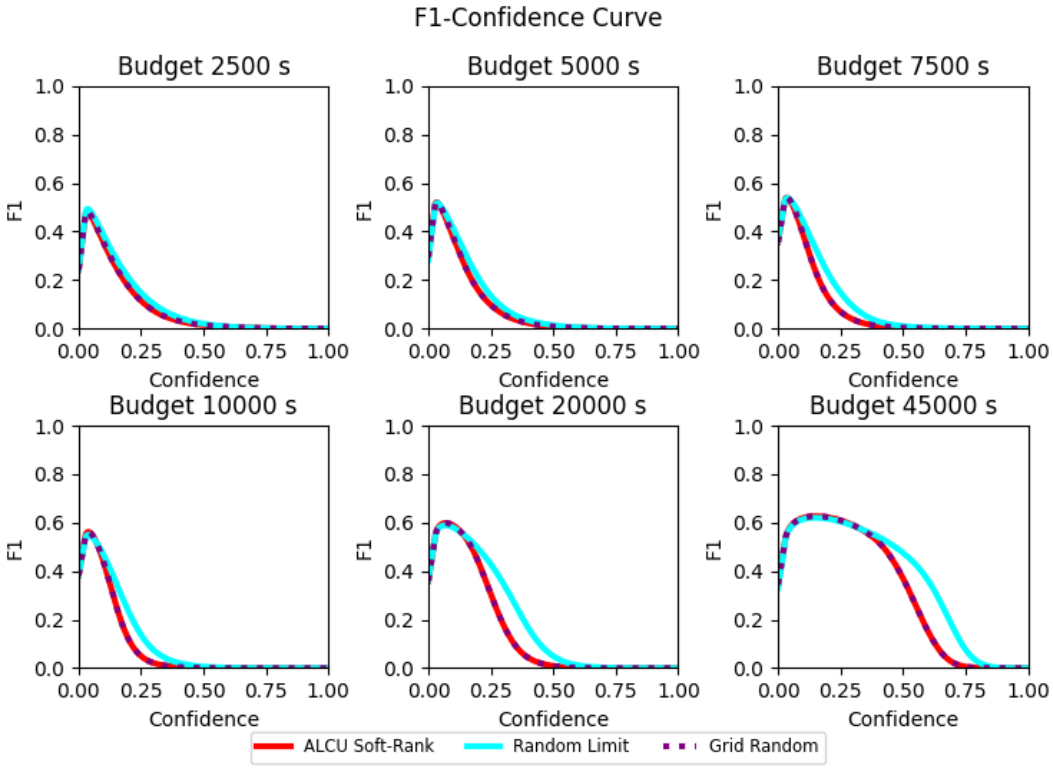


Figure 6.1: Figure showing the F1-Confidence-curve for different AL iterations for the semi-supervised strategies ALCU Soft-Rank (red) and Random Limit (turquoise) and the weakly semi-supervised strategy Grid Random (purple).

The F1-confidence curves for one of the seeds will also be utilized to optimize the confidence threshold when employing the corresponding model for prediction. To observe how the models evolve over the AL budget in practical scenarios, predictions will be made on a single image using the trained model at the different budget intervals. The image shown in Figure 6.2 will serve as an example for prediction, accompanied by the ground truth labels. The prediction evolution for ALCU Soft-Rank, Grid Random, and Random Limit is visualized in Figures 6.3, 6.4 and 6.5, respectively.

(a) Raw image **(b)** Ground truth image

Figure 6.2: Example image from test set, and the corresponding ground truth labels.

(a) Budget 2500 s

(b) Budget 5000 s

(c) Budget 7500 s

(d) Budget 10 000 s

(e) Budget 20 000 s

(f) Budget 45 000 s

Figure 6.3: Semi-supervised Soft-Rank model predictions on a test image for different budgets during the AL process. Note that predictions has an increasing confidence and with more accurate predictions especially after 7500 s.

(a) Budget 2500 s

(b) Budget 5000 s

(c) Budget 7500 s

(d) Budget 10 000 s

(e) Budget 20 000 s

(f) Budget 45 000 s

Figure 6.4: Weakly semi-supervised Grid Random model predictions on a test image for different budgets during the AL process. Note that the confidences increase with a bigger budget but also that this method seems to have more FP in the background compared to the other.

(a) Budget 2500 s

(b) Budget 5000 s

(c) Budget 7500 s

(d) Budget 10 000 s

(e) Budget 20 000 s

(f) Budget 45 000 s

Figure 6.5: Random Limit model predictions on a test image for different budgets during the AL process.

Chapter 7

Discussion

The performance of the semi-supervised framework shows promising results for a couple of methods. As mentioned in the results section, the ALCU Soft-Rank strategy is the best-performing strategy. When comparing it to the baselines, it outperforms the Random strategy and the Random Limit strategy at 5 budget points each. However, upon examining the averages and standard deviations in Table 4.1, overlap between the strategies is observed, particularly with the Random Limit strategy, when considering the standard deviations. The initial pool has a substantial impact on the starting mAP, as indicated in Table 4.1 with a standard deviation of 7.6, which suggests that it may continue to influence the scores in subsequent iterations. Therefore, it would be beneficial to further evaluate the results, perhaps through rank tests to determine if ALCU Soft-Rank consistently outperforms the other methods or if there is considerable variation, as it currently appears. Despite the lack of strong evidence indicating that ALCU Soft-Rank is superior to Random Limit, it is an easy strategy to implement and may therefore be worth considering, as it does not appear to have any negative effects.

The impact of the image limit on the results is notable, particularly evident when comparing Random and Random Limit in Figure 4.2, which displays the mAP50 versus budget graphs. We suspect that this is due to our model choice for the semi-supervised setting, YOLOv8, which was not designed for such conditions. Instead, it was developed for fully-supervised scenarios where it has access to all object labels. Imposing an image limit makes the scenario more similar to fully-supervised training. However, this does not explain why a lower number would yield even better results. One possible reason is that more images allow the model to train on more background, which can also provide valuable information.

Looking at the mAP50 versus budget graphs for each class in Figure 4.3, a substantial difference in mAP-score among the different classes is evident across all the strategies. This is consistent with the unbalanced dataset, visualized in Figure 2.1. We created Table 4.2, presenting the class balance for each strategy and budget, to investigate whether any of the strategies inadvertently brought improved class balance, but we found that this is not the case, the distribution remained mostly consistent with approximately 40% healthy berries, 22% berries with scab lesions, and 38% berries with active lesions, which is about the same distribution as the whole dataset. The mAP for the ”scab” class is notably lower compared to both

the ”scab” and ”healthy” classes, indicating the potential benefits of improved class balance. One thing to note is that the mAP-scores for the scab class come fairly close to the fully supervised case, suggesting there might not be much more to gain. Another alternative could be that the small difference in class distribution, about 22% compared to the actual datasets 18%, is actually helpful for the model.

In the weakly semi-supervised setting, the best-performing strategy is Grid Random, as evident from the average mAP gains shown in Table 5.2. We believe it performs best partly because it is a grid strategy, which worked as intended, labeling more points, as seen in Table 5.3. Additionally, it spreads out the point annotations, which may reduce the likelihood of the same berry being annotated first with a point and then with a bounding box label if the model is still uncertain about it, thereby ”wasting” fewer point labels.

However, even if Grid Random is the best performing weakly semi-supervised strategy, it does not surpass ALCU Soft-Rank when looking at the average mAP gain, which only uses box labels. This outcome is not surprising, considering that the setting did not demonstrate significant potential from the start when utilizing free points, as depicted in Figure 5.1. Except for the figure showing the potential with PLS, the presented results are dependent on the set annotation costs outlined in Table 3.1. These costs were determined based on simple experiments and are more of an educated estimate than precise figures. Furthermore, they are likely to vary depending on the chosen annotation tool. As mentioned previously, the crucial aspect is the ratios between them rather than the actual number. Currently, point labels are not considered worthwhile, as the semi-supervised AL framework with the ALCU Soft-Rank strategy outperforms it with the same budget. However, if the cost ratio were to shift, such as if box labels were to become even more expensive, it might be worthwhile to explore the weakly semi-supervised AL framework further.

There is a scenario where further exploration of the AL framework for the weakly semi-supervised setting might be worthwhile: when both experts and non-experts are involved in the labeling process. Some instances, such as the transition between healthy and scab lesions, and between scab and active lesions, can be challenging to label. In such cases, non-experts could handle the point-labeling since it does not require a class label. This approach might reduce noise in the dataset, potentially compensating for the performance gap between the two methods. However, further investigation is needed, as all testing has been conducted using ground truth labels.

As seen in Figure 6.1 showing the F1-scores, the lines showing both the Soft-Rank method and the Grid Random seem to be the same. This is interesting as the purpose of the PLS was to increase the F1-score, and the confidence of predictions, but here it does not seem to have affected it at all.

Looking at the actual predictions that the different models make on an example image, as shown in Figures 6.3, 6.4, and 6.5, there are some similarities and differences. Firstly, all of the models handle background well, with very few, if any, FP. In general, however, the random limit model has higher confidences overall, especially for the more expensive budgets, which is also seen in the comparison of the F1-scores in Figure 6.1. When looking at the instances compared to the ground truth labels in Figure 6.2, there are wrongly classified berries with high confidences here too, for example, the most left-centered scab berry, which it predicts to be active with

0.65 confidence for the budget of 45 000 s. This prediction has not been changed throughout the other budgets either. As for the Soft-Rank strategy, it does classify this berry wrong for the first budget at 2500 s and then shifts between right and wrong until 7500 s, where it classifies it correctly and retains that prediction for the rest of the learning process. Worth noting is that the ground truth image labels in Figure 6.2 also may have some labeling bias, meaning that some berries that are in the background or only slightly showing themselves behind branches have not been labeled. There is also a scab which could also be predicted as a healthy berry. In general, regarding the results, it is important to consider that the dataset has been labeled by non-experts which might affect the overall performance.

7.1 Future Work

Moving on with this project, it would be interesting to first evaluate the current results more in depth. This could mean running more seed tests than the ten that were done for this thesis, but also by computing statistical significance tests as suggested in the previous section.

As this thesis has been more directed towards uncertainty query strategies, it would also be interesting to see more experiments including a diversity-based query strategy and class balancing. Or even better, try to incorporate it into ALCU Soft-Rank to create a hybrid strategy. We believe that, when evaluating the class performances in Figure 4.3, there is still gain to be found in the scab class, which performs much lower compared to the healthy and active.

Moving on, even though YOLOv8 is regarded as one of the best one-shot detectors for real-time applications, it is not optimal for training in a semi-supervised setting. It also lacks a bit of adaptability for development as it, for example, does not provide very good insight into individual instance class probabilities, which makes it inapplicable with most AL query strategies that are based on this. Therefore, it would be interesting to see how another object detection model would perform, which is more adjusted to semi-supervised data.

Chapter 8

Conclusion

In conclusion, there are many key takeaways from this project. Starting with the research questions, there is value to be found in AL for annotation efficiency for detecting different stages of CBD using the provided dataset. As shown in the results section, the semi-supervised ALCU Soft-Rank strategy consistently beats the random baselines and has better overall performance. Using these methods, can potentially lead to easier and cheaper annotation in the future.

As for the weakly semi-supervised setting, there is value to work towards, which is shown in the potential figure from the results. However, with the current point/box cost quota based on realistic values, this method does have challenges to overcome before being a valuable contributor for annotation, as it did not show any huge improvement compared to the semi-supervised setting. This does not take into account that the weakly semi-supervised point method is still a strong tool for adaptability since it does not require constant expert knowledge due to the fact that it is based on only localization and does not take stages of disease into account. This would mean that it could still be a powerful annotation tool to use for non-experts or in a berry detection setting if a labeled foundation already exists.

This thesis shows an ambition to strive for easier and cheaper annotation, lowering the expensive initial cost of annotation for machine learning object detection systems. This makes it more attainable for the implementation of small-budget scale computer vision detection systems. This in turn could benefit small operators, in our case, local farms looking to implement such systems on a small scale without any good alternatives for 'off the shelf' solutions. As images could also be region-specific, a slight adaptation of solutions also becomes easier.

In the future, it would be interesting to see how another model detector would perform, as well as how diversity and class balancing strategies would work together, possibly also on a bigger and higher-quality dataset. We believe that these additions would make active learning even more beneficial, with the possibility of gaining even more in annotation efficiency.

Bibliography

[1] Statista. *Coffee - Worldwide Revenue | Statista Market Insights*. URL: https://www.statista.com/outlook/cmo/hot-drinks/coffee/worldwide#revenue (visited on 03/05/2024).

[2] Girma Adugna. "Coffee berry disease: a century-old anthracnose of green berries of Arabica coffee (Coffea arabica L.) in Africa". In: *Journal of Plant Diseases and Protection* (2023), pp. 1–14.

[3] HAM Van der Vossen and DJ Walyaro. "Additional evidence for oligogenic inheritance of durable host resistance to coffee berry disease (Colletotrichum kahawae) in arabica coffee (Coffea arabica L.)" In: *Euphytica* 165.1 (2009), pp. 105–111.

[4] Clemens-Alexander Brust, Christoph Käding, and Joachim Denzler. "Active learning for deep object detection". In: *arXiv preprint arXiv:1809.09875* (2018).

[5] Huy V Vo et al. "Active learning strategies for weakly-supervised object detection". In: *European Conference on Computer Vision*. Springer. 2022, pp. 211–230.

[6] Dong Liang et al. "MUS-CDB: Mixed Uncertainty Sampling with Class Distribution Balancing for Active Annotation in Aerial Object Detection". In: *IEEE Transactions on Geoscience and Remote Sensing* (2023).

[7] Grand View Research. *Data Collection And Labeling Market Size, Share Trends Analysis Report By Data Type (Audio, Image/ Video, Text), By Vertical (IT, Automotive, Government, Healthcare, BFSI), By Region, And Segment Forecasts, 2023 - 2030*. URL: https://www.grandviewresearch.com/industry-analysis/data-collection-labeling-market (visited on 03/20/2024).

[8] Ross Cunnington. *Neuroplasticity: How the brain changes with learning*. School of Psychology and Queensland Brain Institute, University of Queensland, 2019.

[9] Syed Sahil Abbas Zaidi et al. "A survey of modern deep learning based object detection models". In: *Digital Signal Processing* 126 (2022), p. 103514.

[10] Tausif Diwan, G Anirudh, and Jitendra V Tembhurne. "Object detection using YOLO: Challenges, architectural successors, datasets and applications". In: *multimedia Tools and Applications* 82.6 (2023), pp. 9243–9275.

[11] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[12] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLO*. Version 8.0.150. Jan. 2023. URL: https://github.com/ultralytics/ultralytics.

[13] Dillon Reis et al. “Real-time flying object detection with YOLOv8”. In: *arXiv preprint arXiv:2305.09972* (2023).

[14] Nils Eickhoff and Axel Eiman. “Weakly semi-supervised object detection for annotation efficiency”. Chalmers University of Technology, 2023.

[15] Shiryu Ueno et al. “Benchmarking of Query Strategies: Towards Future Deep Active Learning”. In: *arXiv preprint arXiv:2312.05751* (2023).

[16] Andreas Kirsch et al. “Stochastic Batch Acquisition: A Simple Baseline for Deep Active Learning”. In: *arXiv preprint arXiv:2106.12059* (2021).

[17] E. J. Gumbel. *Statistics of Extremes*. New York: Columbia University Press, 1958.