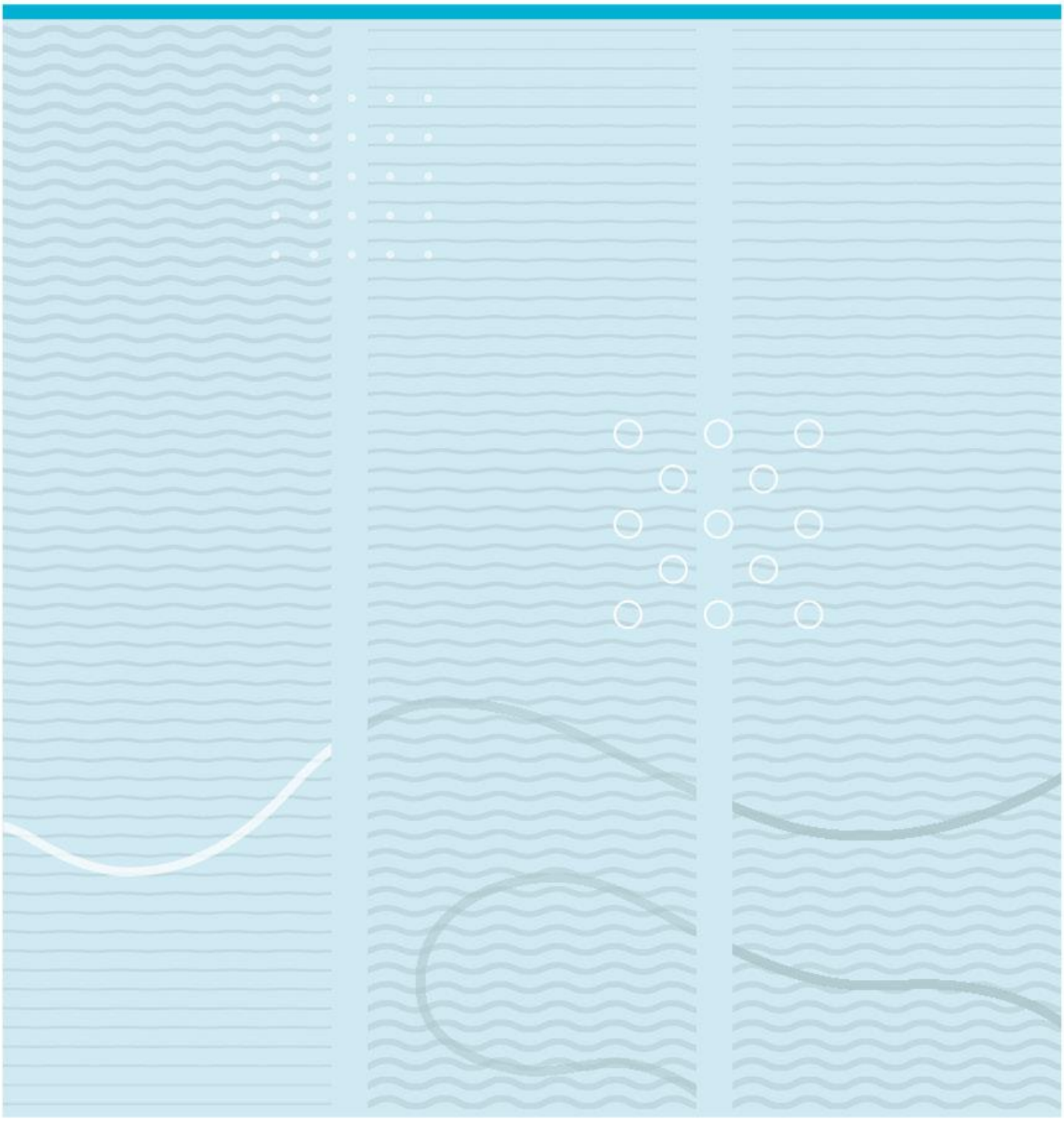
University of South-Eastern Norway
Faculty of Technology, Natural Sciences, and Maritime Sciences
–
Master's Thesis
Study programme: Master of Science in Micro and Nano Systems Technology
Spring/Autumn 2023

Amin Mavaddatkhelejani

DRCAP : Detection, Regression, Classification, and Processing of Apple trees



University of South-Eastern Norway
Faculty of Technology, Natural Sciences and Maritime Sciences
Department of Microsystems.
Raveien 215
NO-3184 Borre, Norway

http://www.usn.no

Summary

Yield prediction is a crucial aspect of crop management in the fruit farming industry. In recent years, a variety of methods have been increasingly utilized to address this challenge, in order to enhance the technical capabilities both in the hardware and software domain.

To tackle this challenge, this thesis, as a part of research project aimed at crop management called “Earlier and more precise yield forecasts in fruit production – development of the digital tree” (translated from Norwegian), we designed a pipeline with separate sections called DRCAP. the sections cover Detection, Classification, Regression, and Pixel Processing of apple tree images taken by RGB sensors. The primary objective of this pipeline is to identify apples at the first stage and then send to Analysis part in order to make a yield prediction.

We implemented YOLOv7 for apple detection and used a custom convolutional model as the classifier and a custom regressor for the regression task. Indeed, an image processing technique has been applied as an auxiliary method alongside a classifier and regressor. We implemented the coding of this project partially in the Google Collaboratory service and partially in the local machine with Jupiter Notebook.

The main data set for this study comes from an apple orchard in Gvarv, Telemark (Norway), taken over a growth season in 2021 and 2022 which only 2021 dataset was used in this study.

Preface

Without any doubt, the journey to completion of this master's thesis has been an enlightening experience with a profound sense of accomplishment and gratitude. My pure apology for the ones omitted inadvertently in this acknowledgment.

First and foremost, I express my appreciation to my supervisors and co-supervisors for their constant help throughout this work. Dr. Erik Andrew Johannessen, with his unhesitating commitment to work ethics and his support by any means, from providing the facilities to managing unforeseen challenges, would like to extend my thanks to Karl Thomas Hjelmervik for his unstoppable struggles with his best, which were an inspiration for me in facing challenges. I offer a special thanks to Christian Hovden for his support in providing the data set essential for this study, and his broadening support pushed the study a step further.
I am equally grateful to Dr.Fabio Augusto de Alcantara Andrade for his indispensable technical support and perseverance, which gave me the courage to overcome the obstacles.
Furthermore, my greatest thanks go to all of my esteemed professors at USN for their unforgettable help in bringing this thesis from an idea to a more realistic realm.

Last but by no means least, I am profoundly grateful to my family for their support throughout my academic journey. Their encouragement and belief in me have been invaluable in reaching this milestone.

Vestfold, Norway, September 2023
Amin Mavaddat

Contents

Table of abbreviations

DRCAP	Detection, Regression, Classification, and Processing of Apple trees
4IR	Fourth Industrial Revolution
SAGUIN	space–air–ground–undersurface integrated network
DSSAT	Decision Support System for Agro-technology Transfer
APSIM	Agricultural Production Systems Simulator
ANN	artificial neural networks
VGG-16	Visual Geometry Group
RESNET-50	Residual Neural Networks
F-VGG16	Fine-tuned VGG16
CNN	convolutional neural networks
F-RESNET50	Fine-tuned RESNET50
CCNN	Custom Convolutional Neural Network
YOLO	YOLO You Only Look Once
COCO	Common Objects in Context
IOU	Intersection over Union
XGBoost	extreme gradient boosting
SVC	Support Vector Classifier
SVM	Support Vector Machines
RBF	radial basis function
Selu	Scaled Exponential Linear Unit
RELU	Rectified Linear Unit
ELU	Exponential Linear Unit
Leaky ReLU	Leaky Rectified Linear Unit
SGD	Stochastic Gradient Descent
Adagrad	Adaptive Gradient Algorithm
ADAM	Adaptive Movement Estimation algorithm
ADAMx	Adaptive Moment Estimation optimize
Nadam	Nesterov-accelerated Adaptive Moment Estimation
MSE	GNU Image Manipulation Program
GIMP	mean average precision
mAP	mean average precision

Chapter 1

Introduction

1.1 Motivation

Incontrovertibly, the use of appropriate technology will enhance the results of systems on any scale, regardless of the field of study. Herein lies the importance of raising the sustainability of agricultural products and their production lines from A to Z. One example is the growth and handling of apples, which are widely planted not only in Norwegian fruit gardens (orchards) but in most countries sharing a similar climatic habitat. The knowledge of handling agricultural products lies with the tacit knowledge of the farmer, and a more reliable data-driven decision-making system will help secure a healthy production and harvest. Within this lies the optimization of sensor parameters that can monitor soil moisture, nutritional level, potential disease, and the presence of pests, where any deviation from the norm would offer an early warning regime and timely intervention that stops or prevents any further detrimental effects to the crop. By timing the harvesting period, one can negate bottlenecks in distribution and prevent crops designated as food from being wasted. Hence, the digitalization of the orchard could play a crucial role in both food security and sustainability. There are several examples where the integration of some “intelligent” detection systems has proven to yield a multitude of benefits for crop management, fruit grading, resource optimization, disease and pest management, quality control, and sustainability [8],[9],[10],and[11].

Based on the idea of continuous monitoring of the orchard, this will identify and track apples through their growth season, from blossoms to harvest. This will form a central part of any crop management system since the growth of apples represents the output of all parameters affecting the orchard. The data will be based on images taken from two different apple trees situated in two different orchards as representatives for the rest. The model will, in addition, be tested on random trees to check its validity. Based on this data, a decision will be made concerning the harvest time prediction, and the earlier this can be done, the better prediction for distribution into the market can be made.

1.2 Background

1.2.1 Review

The growing pace of sophisticated integration of modern equipment has led the apple industry to become more technology-based than the general farming industry [8],[9]. The region of Vestfold and Telemark, a main apple producer province in Norway, plays a prominent role in the usage of technology to reduce apple waste and improve the management of crops.

As part of the Fourth Industrial Revolution (4IR), Agriculture 4.0 uses technology to improve farming methods. This cuts down on wasteful resource use and supports long-term infrastructure for farming [12]. The Fourth Industrial Revolution (4IR) and Agriculture 4.0 play a crucial role in the agriculture domain, as does the fusion of technologies in healthcare or industry [12] (Fig.1.1). Also, the influence of the fourth industrial revolution becomes prominent if the consequences of climate change show more invasive effects in terms of rainfall and temperature patterns [13]. Unlike traditional methods, 4IR, regardless of its invasive approaches to threatening the farmers' job market, plays a crucial role in optimizing crop management [10].

In addition, the technical benefits help farmers' economies grow because the technology makes harvesting more efficient and accurate by letting them track key growth parameters in real-time or find diseases or pests early. Additionally, the utilization of intelligent detection systems will fine-tune any measures required to meet deviations from "normal" growth conditions and thereby optimize the use of natural resources. This not only provides economic benefits but also promotes more sustainable agriculture and facilitates an environment-friendly approach to resource management. Moreover, the implementation of intelligent apple detection systems will consequently provide a viable solution to improve the overall feasibility of the apple farming industry as a part of the food industry[11].

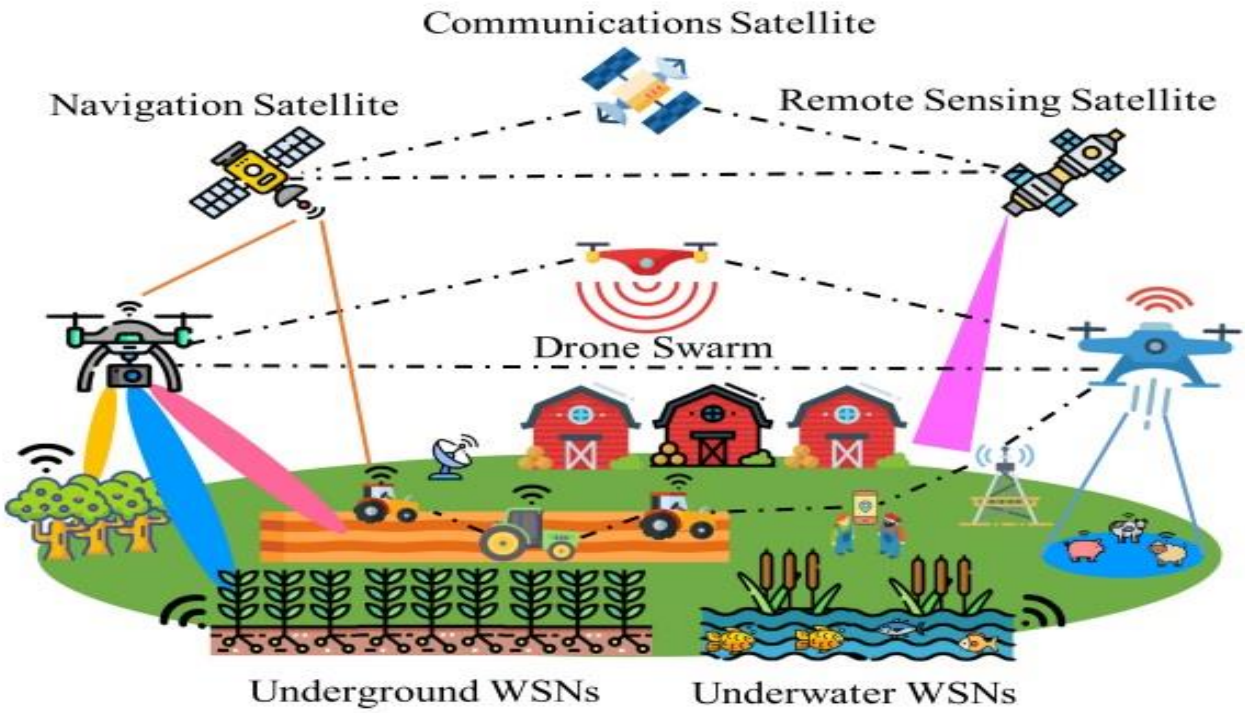


Figure 1.1 space–air–ground–undersurface integrated network (SAGUIN) in agriculture 4.0 [37].

On the other hand, manual yield estimation methods employed by farmers present numerous challenges. For instance, since different people may measure the same plant characteristics differently, visual estimation is prone to intrinsic errors. Weather conditions and even the person's experience can also have an impact on the estimation. However, some methods, like the Penman-Monteith formula, showed relatively satisfying results but still had challenges [14]. These challenges are due to the need for costly experts and time-consuming estimation procedures [15]. Therefore, it is crucial to implement precise and objective solutions that are less biased by the subjective opinion of the individual farmer. The development of more accurate and efficient methods for estimating crop yields is also a necessary step toward meeting the growing demand for food production[16].

Before the widespread use of machine learning and deep learning methods for fruit yield estimation, some computer technology-based methods were used to develop a variety of techniques for estimating yields. For instance, statistical models, crop simulation models, remote sensing-based methods, and farmer surveys.

In modeling, historical data from previous yields, such as weather conditions or other correlated data, has been employed to make predictions for future estimations. Linear regression, multiple regression, and time series are the most common of them. For the world's top six produced crops (wheat, rice, maize, soybean, barley, and sorghum), statistical modeling has shown that just a simple measurement of growing season temperature and precipitation can account for roughly 30% or more of the yearly fluctuations in worldwide average yield volume. However, all of the crop-yielding models are influenced by scale and empirical models on a global level, which may not provide accurate predictions for sub-global scales [17].

In crop simulation models, crop yield is evaluated based on environmental and physiological factors. DSSAT (Decision Support System for Agro-technology Transfer) is a collection of computer software that operates together, and the most recent version of it covers almost 40 different crops [18]. In another capability of DSSAT, the impact of climate change on wheat yield in the Huang-Huai region of China, the results have been shown under different climate scenarios in the long, mid, and short term [19]. APSIM (Agricultural Production Systems Simulator) in crop simulation models showed results in terms of providing a yield estimation with a bias [20].

In recent years, there has been remarkable progress in prediction methods. Specifically, in above-mentioned models have shown promise in classifying objects such as apples in orchards. In order to train these deep learning systems, researchers utilize datasets consisting of sophisticated image labeling processes [21]. In some cases, high-tech data acquisition systems have been utilized to collect clear images of as high quality as possible, but the additional labor intensity involved in doing so would be another issue for farmers and producers [22]. Furthermore, several factors must be considered in the above-mentioned methods of yield estimation for crops. The majority of the methods are based on complex models that take into consideration the underlying mechanisms of crop growth based on the data supplied to the models. Not only that, but also, the required input data and its acquisition methods in the mentioned models will make them less transparent for the unskilled user and less likely to be adopted by farmers and producers. Furthermore, having a huge data set for training makes them less susceptible to real-time capability. It is therefore required that complex systems harbor a great deal of autonomy combined with a simple user interface if they are to be used successfully.

1.3 Outline

This study is divided into 6 different chapters. The first chapter introduces the goal or aim of this work combined with a briefly addressed review of comparable work in order to introduce the reader to this field of study. In second chapter the dataset will be explained. The third chapter goes through the methods and materials of work by introducing the fundamental requirement to understand the study like the computer vision structures, Deep learning systems, and Image processing technics. Chapter four provides the results of study to the reader. Chapter fives covers the discussion part. Chapter six went through the conclusion of the thesis.

Chapter 2

Dataset

Good datasets are essential in successfully developing any data driven model and highlights to the high dependency of the quality of the output of a system on the input. This entanglement becomes crucial when struggling to enhance a system's accuracy, especially if it is not well-optimized. In order not to suffer from this slang, a crucial amount of time was spent on the dataset, from choosing pictures to labeling in DRCAP; therefore, one of the most important and time-consuming steps in this study was dedicated to the dataset, and giving a brief description of it would be beneficial.

2.1 Data Collection

The images in this study come from four RGB camera sensors that were mounted at two different locations in two separate apple orchards in Gvarv, Telemark (Norway) (Fig. 2.1). Pictures were taken for 2 consecutive years in 2021 and 2022 in the daytime and nighttime, but only the nighttime pictures were used in DRCAP because of the better contrast and illumination in the nighttime pictures (Fig. 2.2). Moreover, the pictures in the dataset were taken every night from blossoming time, May 14, to harvesting time, September 25.

Fig 2.1: RGB sensor's locations in Gvarv, Telemark (a) Apple orchard of Eirik Årnes, (b) apple orchard of Henning Lindheim.

Furthermore, several tagged apples were chosen to scale during the season as well as their pictures for later use in analysis part of DRCAP.

These images come from a Raspberry Pi HQ Camera equipped with an Arducam Lens offering a 6mm Focal Length with Manual Focus and Adjustable Aperture. Additionally, several pictures were added to the test data set from different apple variants with 3024*4032 resolution from different locations (Fig. 2.3).

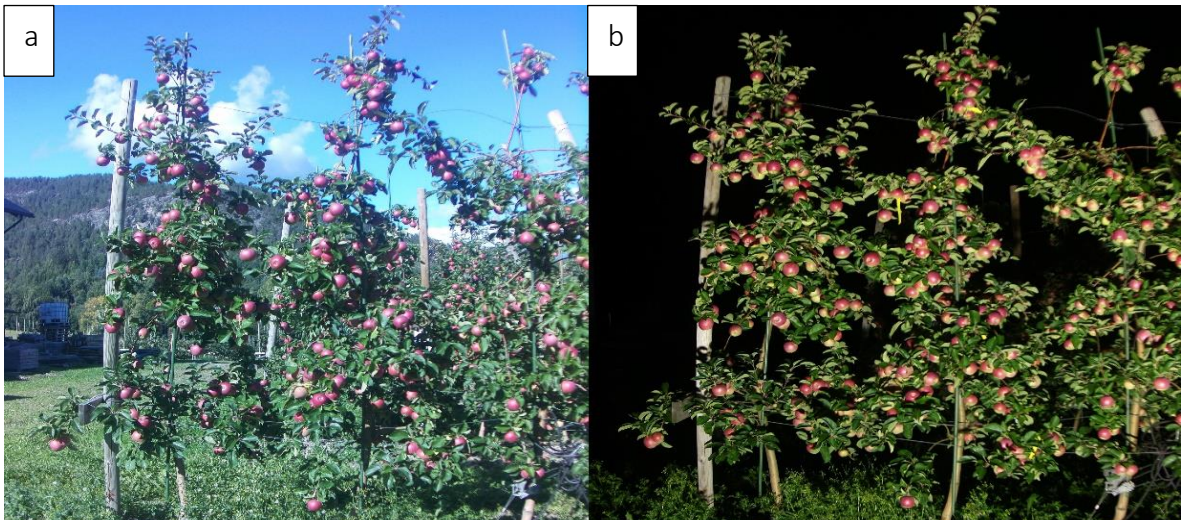


Fig 2.2: a: day-time picture, b: night-time picture contrast comparison

*Figure 2.3: Three apple tree images were taken from three different locations from Drammen and Lier in Norway and different apple variants with an iPhone 14 with a 3024*4032 resolution camera.*

2.2 Data labelling and pre-processing

DRCAP contains two datasets for the detection and analysis parts (Fig. 2.4). There are three different sub-datasets in Dataset 1: Sub-dataset 1, Sub-dataset 2, and Sub-dataset 3. Each has its own augmentation, labeling strategy, and temporal sampling. It is crucial to note that sub-dataset 1, sub-dataset 2, and sub-dataset 3 were made separately from the main dataset with a separate labeling process, and the 2022 collection was just used for testing and remained intact by DRCAP. For 2021 Dataset 1 and Dataset 2, we labeled the images into 3 classes: class 0 for unripe apples (up to July 10), class 1 for semi-ripe apples (from July 11 to August 15), and class 2 for ripe apples (from August 16 to September 14).

The existence of three different sub-datasets was due to its evolving effect. More precisely, we made sub-dataset 1 first and decided to enhance its quality of labeling and augmentation after observing the system's results. This led to the creation of sub-dataset 2. After observing a significant improvement in outcomes, we decided to create a more precise and goal-oriented sub-dataset. This sub-dataset, labeled and augmented with more consideration than the previous ones, was generated based on the revolutionary effects of visiting the previous sub-datasets.

In Dataset 1, the sub-dataset 1 includes 241 images chosen with a one- or two-day interval between two consecutive images that reached 811 images after randomly chosen augmentation. Additionally, the labeling of apples for this sub-dataset was 1900 apples, of which every visible apple was considered. Figure 2.5 shows the labeling strategy for sub-datasets. Apple visibility was the main consideration in this strategy; apples with full visibility were considered for this sub-dataset (Fig 2.5(a)).

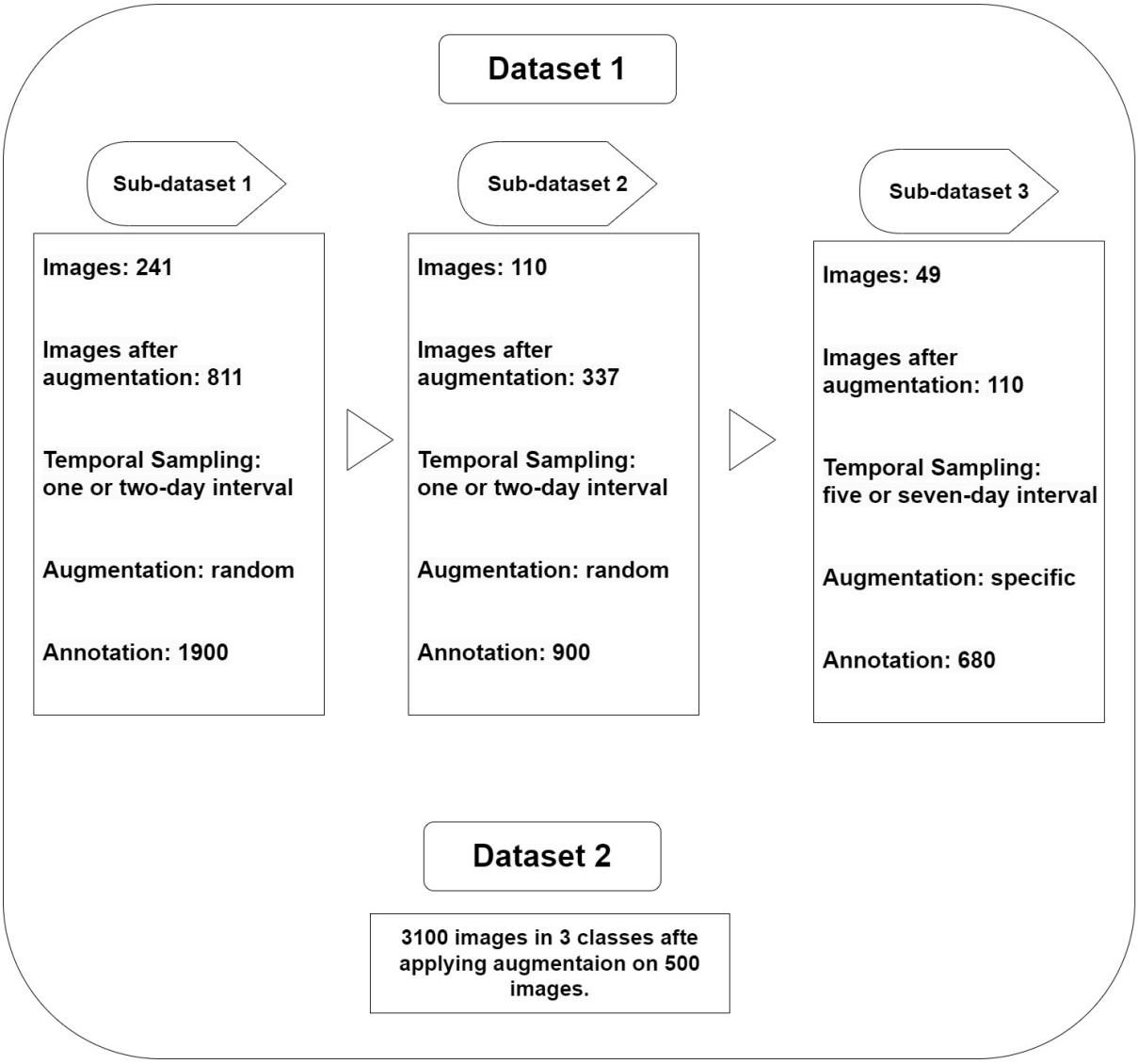


Figure 2.4: DRCAP datasets and sub-datasets. Dataset 1 for detection and Dataset 2 for classification

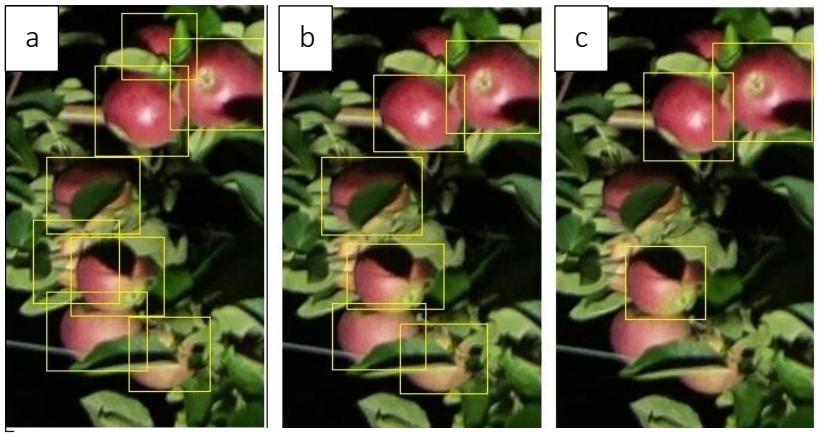


Figure 2.5: labelling method for a: sub-dataset 1, b: sub-dataset 2, and c: sub-dataset 3

The sub-dataset 2 comprised 110 images chosen with a one- or two-day interval between two consecutive images that reached 337 after augmentation. For sub-dataset 2, we labeled apples with semi-full up to full visibility, considering random augmentation (Fig. 2.5(b)). The main difference with sub-dataset 1 was mostly in trying to have a more inter-class balance as long as it had a similar class distribution to the main dataset (Table 1). Since most of the apple-growing process is in class 2, which is the ripening stage, this stage must be included in the dataset as a representation of the apple-growing process. So, this inter-class bias weight has been considered in sub-dataset 2 in terms of picking the same percentage of images of each class from the main dataset.

The sub-dataset 3 has 49 images picked by a five- or seven-day interval between two consecutive images from the main dataset. After augmentation, the number of images reached 110, and more consideration was given to bias and intrinsic bias compared to sub-dataset 2. Moreover, the augmentations in this part were cropping, rotating, and blurring, which is closer to the image-capturing condition in the real orchard. Less desirable illumination and noisier, blurry images because of rain in the real-weather condition of the orchard and foggy weather in the orchard were the reasons for specific augmentations. We labeled the apples in this sub-dataset that had full visibility or more than 75% visibility while ignoring those with less visibility (Fig. 2.6).

Having less desirable illumination and noisier, blurry images because of rain in the real-weather condition of the orchard were the reasons for augmentations. In the labeling process, the apples with full visibility or more than 75% visibility were labeled, and the apples with less visibility were ignored (Fig. 2.6).

It is crucial to note that sub-dataset 1, sub-dataset 2, and sub-dataset 3 were made separately from the main dataset with a separate labeling process, and the 2022 collection was just used for testing and remained intact by DRCAP.

Table 1 Class distribution in sub-data sets and main data set

	Number of images in Sub-dataset 1	**Number of images in Sub-dataset 2**	**Number of images in Sub-dataset 3**	**Class distribution in main dataset**	**Temporal segmentation of classes**
Class 0	37	16	7	15%	up to July 10
Class 1	168	34	34	70%	July 11 to August 15
Class 2	36	14	8	15%	August 16 to September 14

Fig2.6: apple image a: labeled as a class b: not labeled as a class

Dataset 2 belongs to the analysis part used in classification and regression and contains almost 500 images in 3 classes that reached 3100 images after augmentation chosen from main dataset (Fig. 2.7 (a), (b), (c)). In order to make the results as reliable and general as possible, approximately 35% of pictures were collected from the internet (Fig. 2.7 (d), (e), (f))).

Images were labeled in 3 classes for 2021 Dataset 1 and Dataset 2: class 0 for unripe apples (up to July 10), class 1 for semi-ripe apples (from July 11 to August 15), and class 2 for ripe apples (from August 16 to September 14). Indeed, all the labeling was done manually for all datasets and sub-datasets.

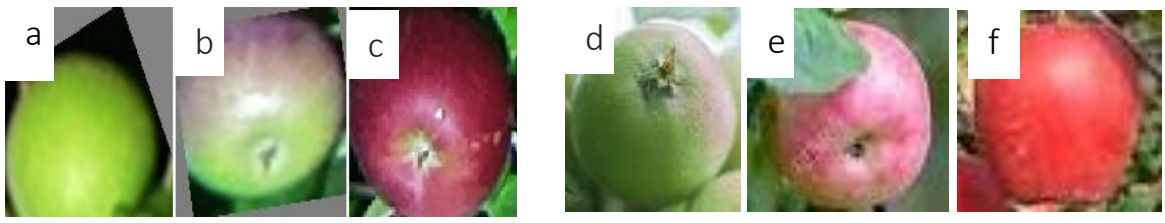


Fig2.7: (a) class 0, (b) class 1, (c) class 2 (d) class 0, (e) class 1, (f) class 2 [1] [6] [7]

Chapter 3

Methodology

3.1 Introduction

In this chapter, more details of DRCAP, including the detection part, analysis part, and image processing part, were explained. This part covered apple detection and cropping, classification, regression, and image processing, as shown in Fig. 3.2. In the first section of DRCAP, the detection part, it has been tried to detect apples and crop them from the tree picture. The automatically cropped pictures were then saved in a folder as input for the second section (Fig 3.1). In the second section, the analysis section, the cropped apples went through three parallel methods to make yield predictions. The blue boxes in Fig. 3.2 were considered the final techniques for each section and sub-section.

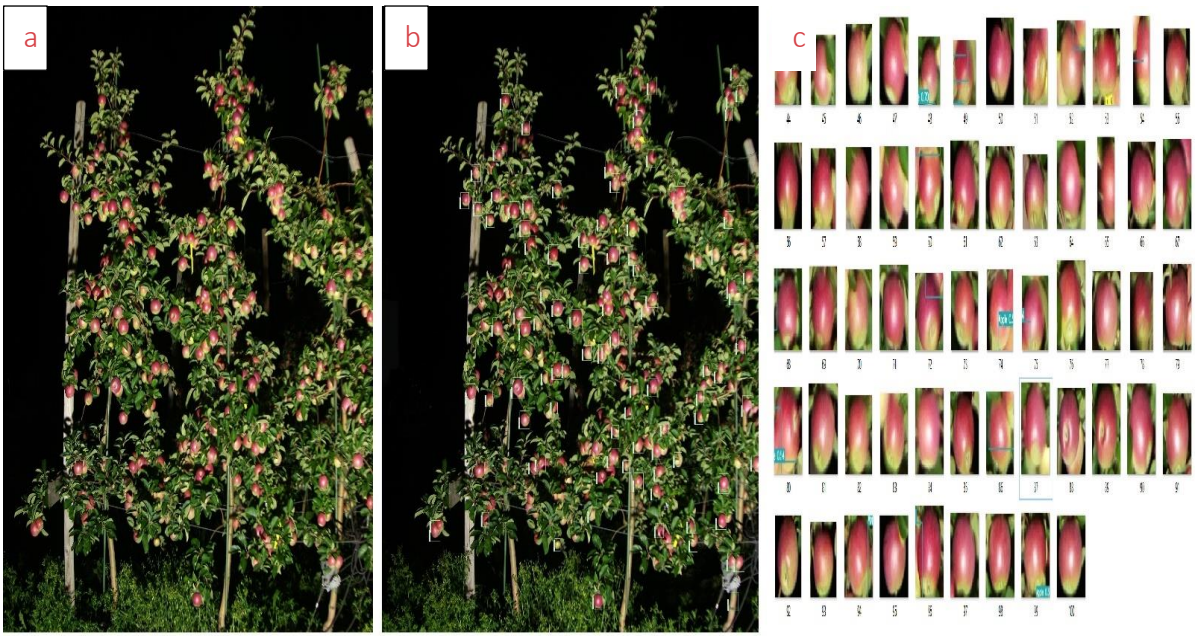


Figure 3.1: a: Input tree image to detection part b: output of detection part before cropping c: output of detection part after cropping

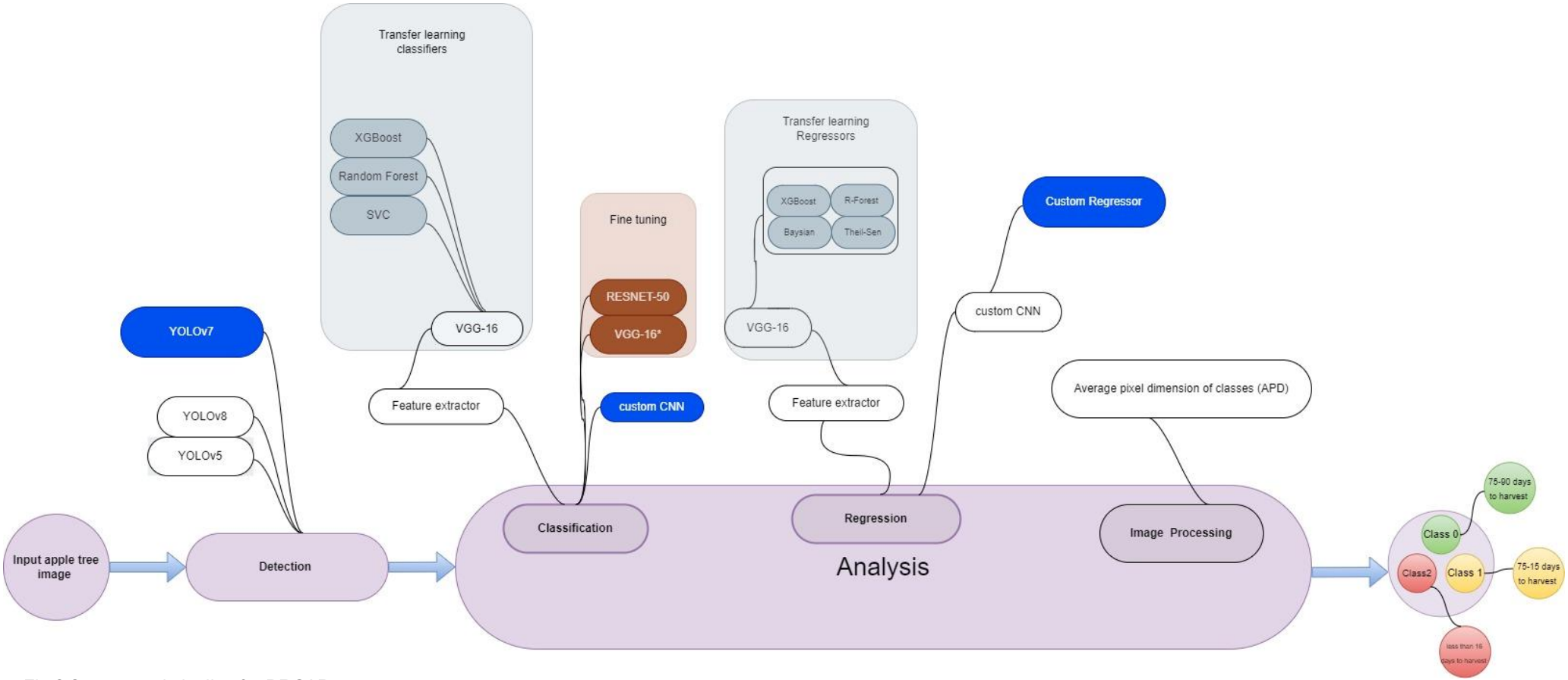


Fig 3.2: proposed pipeline for DRCAP

3.1.1 Introduction to Neural Networks

Neural Networks or artificial neural networks, ANNs, are the building blocks of deep learning and machine learning concepts that try to imitate the human brain's interconnected processing units called neurons [23] (Fig.3.3(a)). Moreover, the elemental constituents of neural networks include input layers, hidden layers, and output layers that each node in those layers are linked to the node in the subsequent ones (Fig.3.4).

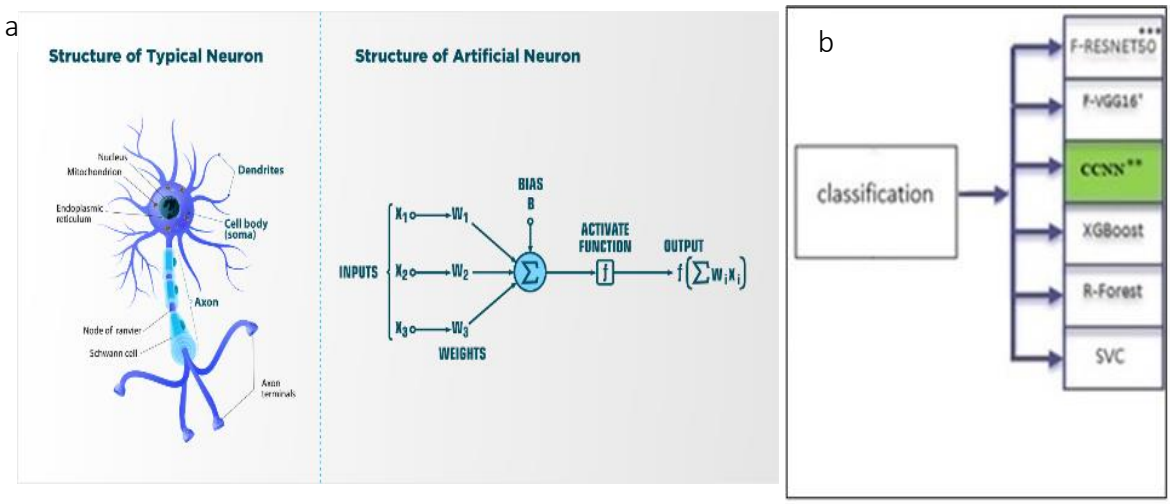


Fig 3.3: Fig 3.3: a: Human neuron similarity with a single neuron [3], b: Classifiers in DRCAP

F-VGG16*: Fine-tuned VGG16
CCNN**: Custom Convolutional Neural Network
F-RESNET50***: Fine-tuned RESNET50

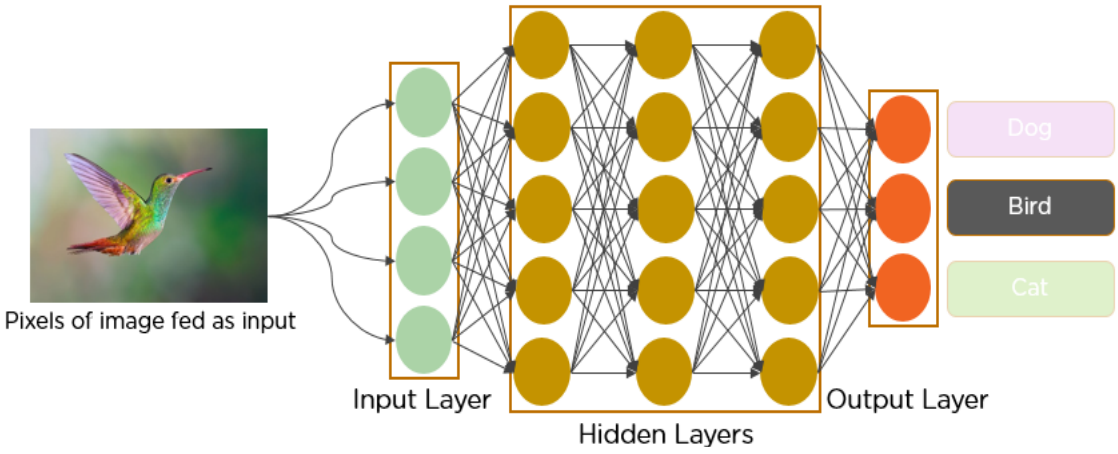


Figure 3.4: A typical Convolutional Neural Network with input, hidden, and output layers[2]

Furthermore, the weight parameters, adjusted in the training phase of the network, control the optimization of the network. Another crucial hyper-parameter, the activation function, in neural networks, introduces a non-linearity to the system by controlling the weights of inputs [23] (Fig.3.3(a)) (section 3.3.4).

Convolutional layers in neural networks, particularly convolutional neural networks (CNNs), are spatial feature extractors that apply convolutional operations to the input data, allowing the network to recognize patterns in images [23]. These layers use learnable filters or kernels to convolve across input data, which captures local patterns and enables the network to efficiently learn the features (Fig.3.5). In this figure, a 3*3 filter convolves with a 7*7 input image to encode a representation (feature map) in which the receptive field is highlighted in pink and the corresponding output value for the position is marked in green. To find the output, the kernel slides over the input image and passes through each pixel position. In this process, the weights of the kernel are multiplied by each pixel value in the corresponding region covered by the kernel. The multiplication of the weights of the kernel with each pixel value in the corresponding region covered by the kernel produces the output.

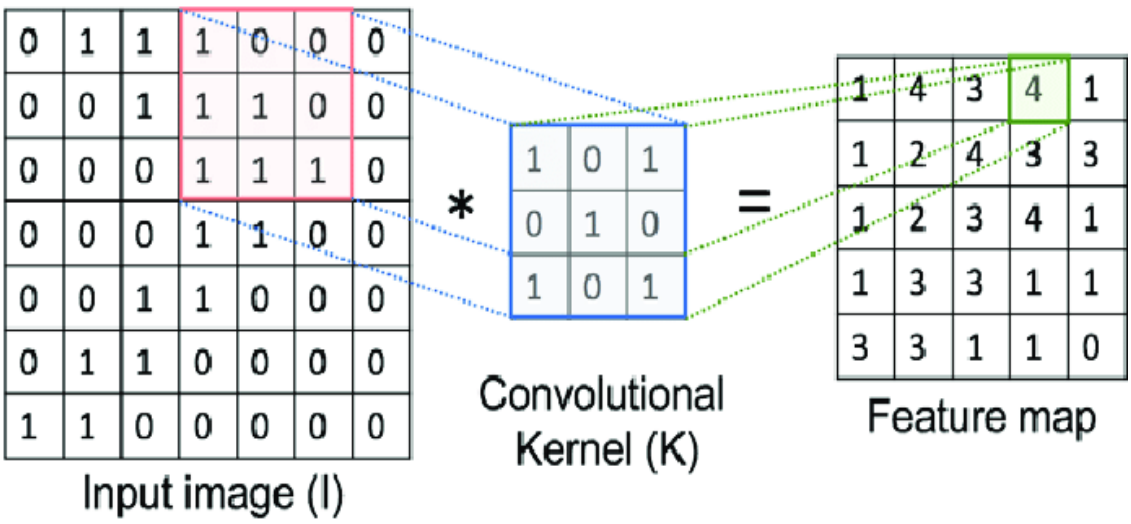


Figure 3.5: An example of an input image I 7x7 convolved with a filter k 3x3 with weights of zeros and ones to produce a feature map[5]

Additionally, there are extensive applications of Neural Networks in a variety of domains from speech recognition [24] , and natural language processing [25] to those relevant to this study in object detection and classification. In our study, the CNNs shown in Fig.3.3(b) were explored.

3.2 Detection

Detecting and localizing an object in a picture stands for object detection, and various methods have been employed so far to detect objects (in this case, apples). Some methods rely on traditional image processing techniques such as space conversion (e.g., RGB, HSV), image binarization, and segmentation to detect [26] or a combination of texture and shape features [17]. Recently developed neural networks like YOLOv5 [27], YOLOv7 [4], and YOLOv8 [28], which stand for You Only Look Once, have been used for this section by pre-trained weights of YOLOv5, v7, and v8.

In the fields of machine learning and deep learning, pre-trained weights are the learned model parameters obtained via training on larger datasets. Weights can be considered as the numerical values of the inner connections of nodes in a neural network or a machine learning model parameter. Additionally, Pre-training would be a beneficial step in enhancing the model's capability and optimization. In most cases, because of the lack of affluent data, as in our case, the transferring of weights from pre-trained models of YOLO played a crucial role in covering the data deficiency [29].

Yolo versions distinguishing performance in real-time applications, especially the YOLOv7, which is 120% faster compared to the YOLOv5 (Fig. 3.6), makes it a proper candidate for fast and efficient detections, particularly for DRCAP.

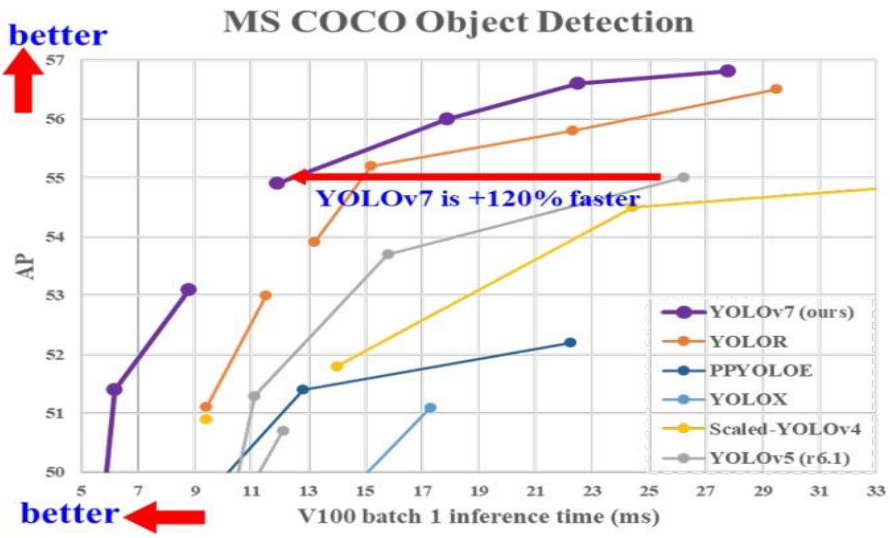


Figure 3.6: YOLOv7 comparison with other real-time object detectors on COCO dataset [4].

Owing to the presence of anchor boxes in bounding box prediction in the YOLO structure, the detection of objects of different sizes and shapes is optimized [25]. As shown in Fig. 3.7(a), anchor boxes are some predefined boxes with different features like aspect ratio and shapes, which help the model predict objects. YOLOv7 takes advantage of nine anchor boxes, which allows it to detect a wider range of objects compared to other versions. Alongside with the anchor boxes importance in the object detection part the Intersection over Union (IOU) should not be forgotten since it is a measure of the overlap between the predicted bounding box and the ground truth bounding box, or actual bounding box (Fig. 3.7(b)).

In DRCAP, the YOLO versions played a crucial role in detecting the apples. However, even using such a well-performed version of a neural network was not enough to boost the apple detection quality without proper data pre-processing and data cleaning methods.

As mentioned before in the chapter 2, there were three sub-datasets in the detection part as the elemental constituents of the detection section, and practically, the major boost in the YOLOv7 performance occurred after the data cleaning and preprocessing methods came across. As shown in Fig. 3.8, to achieve a proper weight function for finding best candidate from YOLO versions, three sub-datasets have undergone 108 training cycles ((number of epochs=4) * (number of batches=3) * (number of yolo versions=3) * (numbers of sub-datasets=3) = 108) with 10% test set considerations for all sub-datasets.

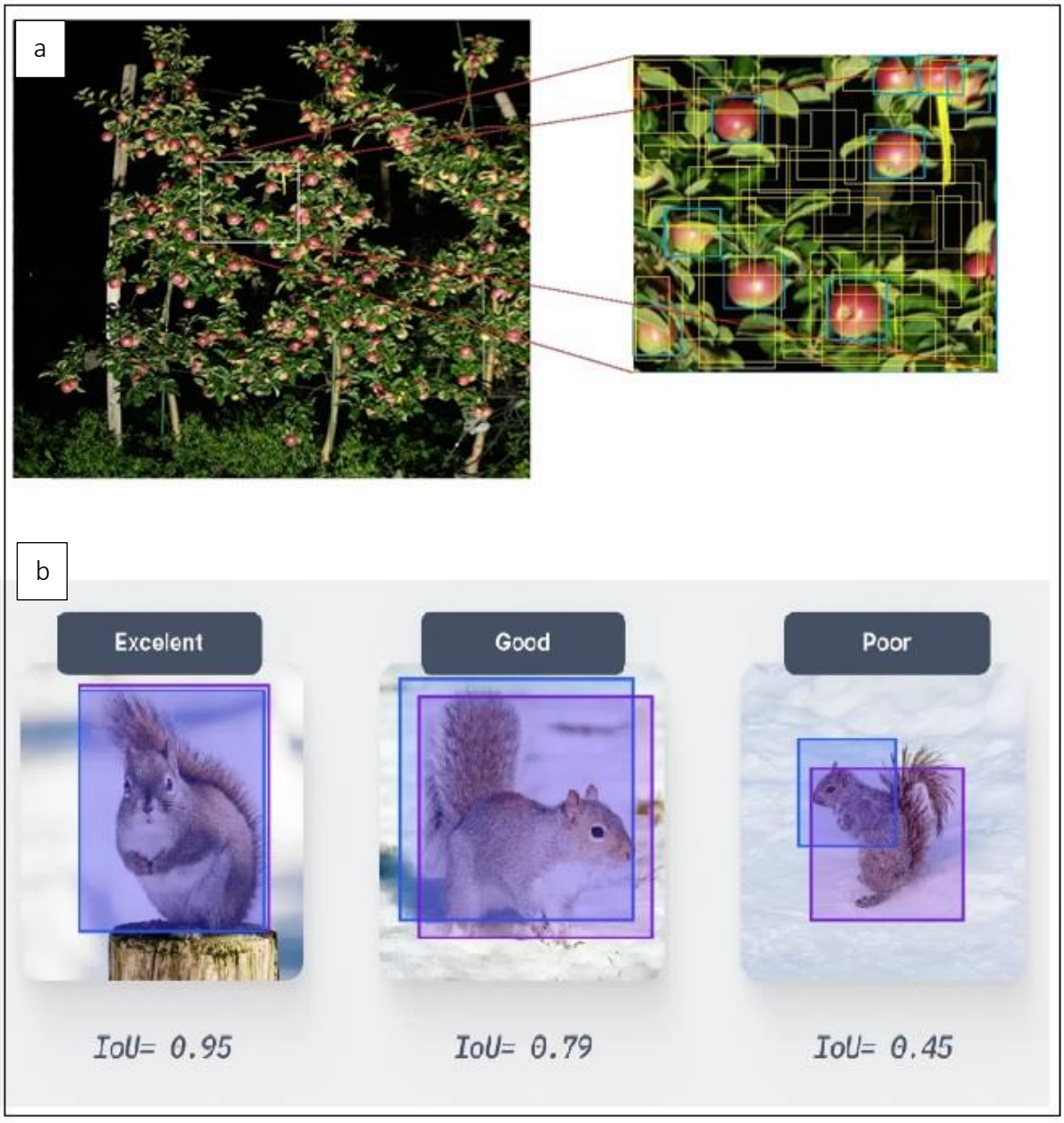


Figure 3.7: a: Anchor boxes in bounding box prediction in the YOLO structure, b: Intersection over Union example in different values for an object [5]

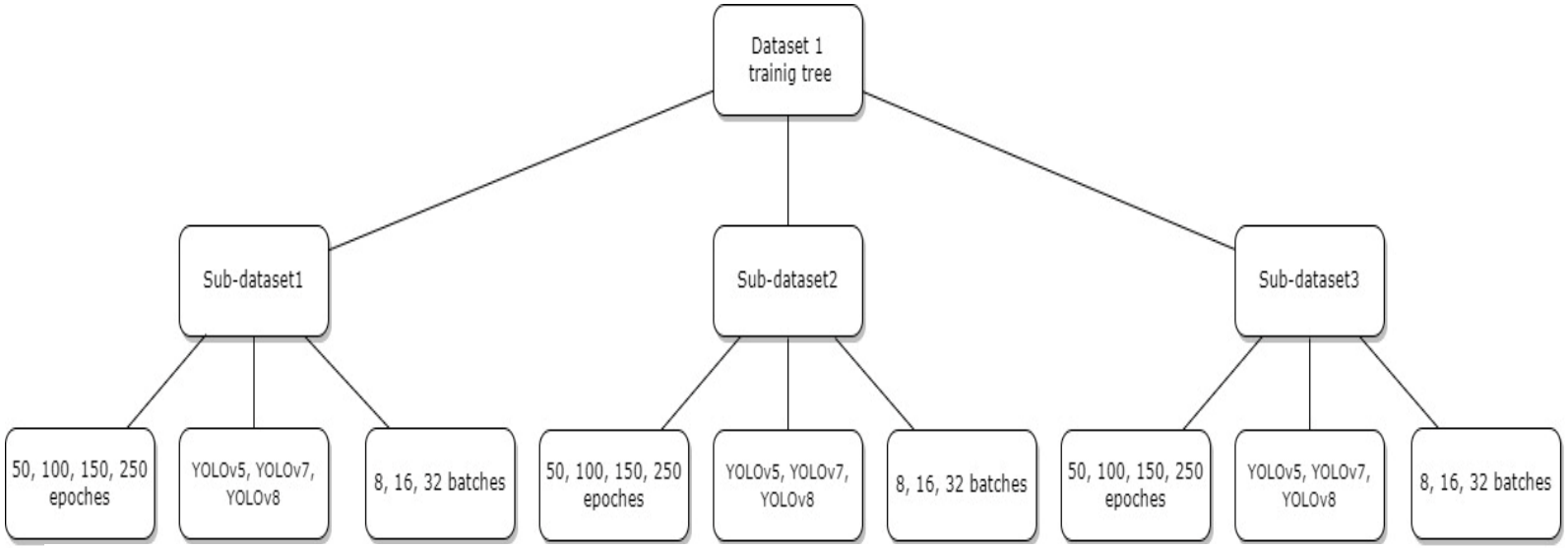


Figure 3.8: training tree cycles description for Dataset1 in the detection part

3.3 Analysis

In this section, the second part of DRCAP, which consists of classification, regression, and image processing, is discussed in detail. To give a brief overview, the cropped apple picture from the detection part of the DRCAP (Fig 3.1(c)) goes through three parallel methods. The first method is a classification technique that attributes a class to the image based on its ripeness including class0, class1, and class3. The second technique is regression, which gives a continues number to the input image from zero to three, which enables the operator to have a better understanding of the apple's ripeness, especially when the apple is in the inter-class transient stage. Finally, in the image processing section, the image was labeled based on its dimension in terms of pixel dimensions. Indeed, an average and class-specified pixel density of pictures has been calculated.

3.3.1 Data collection and processing

First of all, the data set for all three sections in the analysis part is the same. By applying a crop filter to the dataset, the most important pre-processing of the input data for the classification part was already done during the augmentation. Since the output of the detection part was cropped apple images, which could be fully cropped or partially cropped based on the detection output, having a similar augmentation was a necessity (Fig. 3.9). Indeed, the dataset used in this section, Dataset 2, was split into 5% and 10% as test sets.

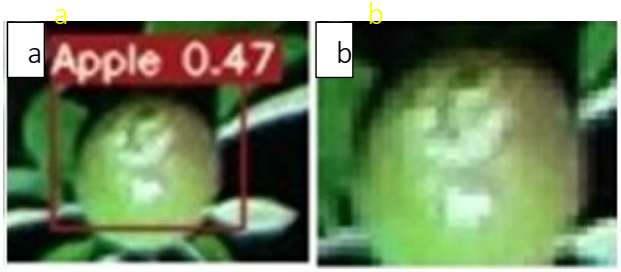


Fig 3.9: a: output of yolov7 b: output of detection part of DRCAP

3.3.2 Classification Model

Classification is a machine learning model or algorithm which attributes the given input, images in this case, to different classes or categories, and its main obligation is to predict the classes of unseen data based on the extracted features during the training process from training and validation data. As shown in Fig. 3.10, six classification algorithms were tested in the classification part to find the best classifier for this part in three different sections.

First, the Visual Geometry Group,VGG-16 [30], which used with the transfer learning method, was the feature extractor in three classifiers: extreme gradient boosting or XGBoost [31], Random Forest [32], and Support Vector Classifier [33]. VGG-16, is a deep convolutional neural network architecture developed by the Visual Geometry Group at the University of Oxford, VGG-16 consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. The VGG-16 model's pre-trained weights on the ImageNet [32] were included in the feature extraction process for the transfer learning part in the classification section in Fig. 3.1. A feature extractor is a convolutional neural network (CNN) model that extracts important features and patterns, such as texture or edges, from the input data. To be more specific, first the VGG-16 architecture's output up to the first convolutional layer in the last convolutional block (block5_conv1) has been taken, which is a tensor. Then, after reshaping this tensor into a one-dimensional vector, it went through the classifiers for training (Fig. 3.11).

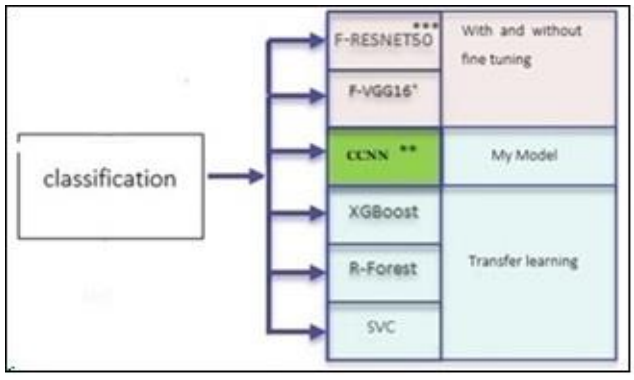


Fig 3.10: Classifiers with more detailed implemented methods

F-VGG16: Fine-tuned VGG16*
*CCNN**: Custom Convolutional Neural Network*
*F-RESNET50***: Fine-tuned RESNET50*

The feature vectorization process involves flattening a feature map (Fig. 3.12), the output of VGG-16 in block5_conv1, to a vector. Finally, the extracted features from images were used as the input for the XGBoost, Random Forest, and Support Vector Classifiers to make predictions.

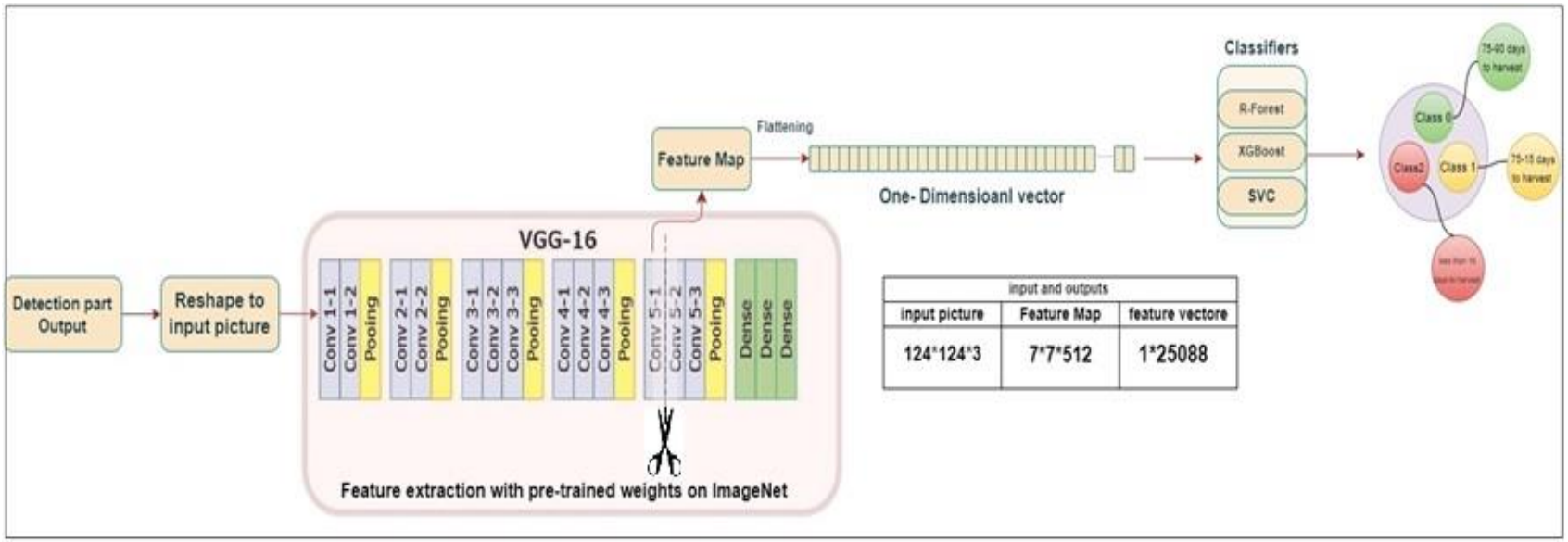


Figure 3.11: feature extraction process for classifiers in the transfer learning part of the classification step in the analysis section

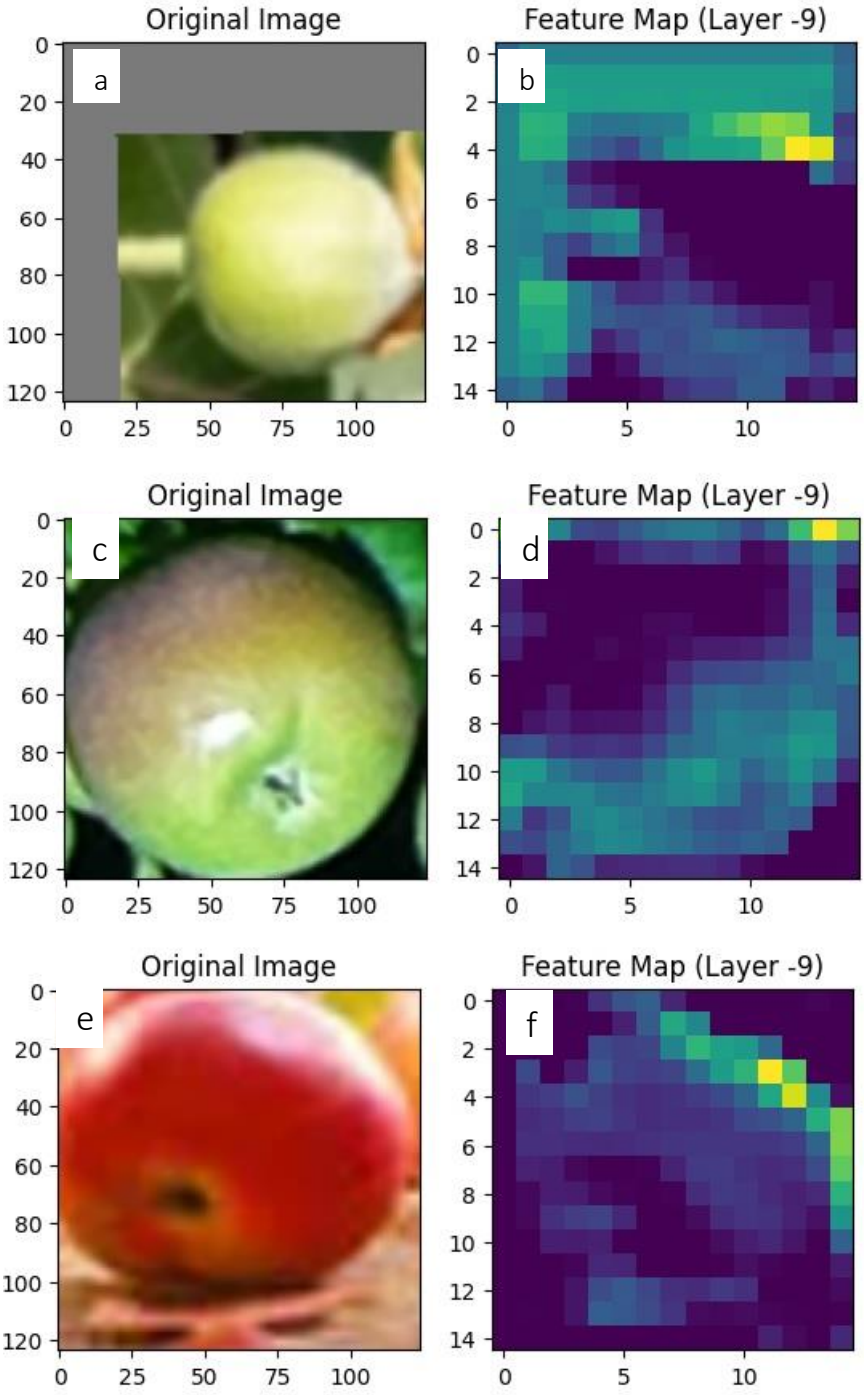


Figure 3.12: Feature map examples for different classes. Rows a-b correspond to class 0, c-d to class 1, and e-f to class 2.

After extracting and flattening the features from the input image by the VGG16, it will go through classifiers for finding the best classifier performance. The first one, was extreme gradient boosting. XGBoost is a versatile algorithm capable of performing classification and regression tasks. It is composed of a sequence of decision trees in which each tree corrects the errors of the previous tree (Fig. 3.13 (a)) [31]. A decision tree is a machine learning system that uses a hierarchical model, like a tree, to provide choices. In this model, core nodes correspond to features, branches correspond to decisions based on those features, and leaf nodes correspond to outcomes. The second classifier was , Random Forest which is another learning algorithm that makes multiple decision trees during training and combines the output of each tree to make a reliable and robust prediction (Fig. 3.13 (b)) [34]. The third classifier was Support Vector Machines. SVM are supervised algorithm that learns from labeled data to make predictions or classifications on new data and can be used for classification and regression tasks. The aim of SVM is mostly to introduce a hyperplane that maximizes the margin among different classes to obtain generalizations to unseen data (Fig. 3.13 (c)) [35].

The hyper-parameters used in the above-mentioned classifiers were another crucial point. The XGBoost hyper-parameters for transfer learning part were: 1000 trees, 0.9 learning rate which controls the step size at each iteration while moving toward a minimum of a loss function, maximum depth of seven which sets the maximum depth of each decision tree, and the gbtree as the booster which indicates that the underlying boosting algorithm is a tree-based model. Indeed, for the Random Forest classifier the employed hyper-parameters were: seven maximum depth, 1000 decision tree in the forest, and zero random state, which sets the random seed for reproducibility. Additionally, in SVC classifier used the radial basis function (RBF) kernel with a degree of 3, a gamma value of 'scale,' and a regularization parameter C set to 10, which is suitable for solving classification problems with non-linear decision boundaries.

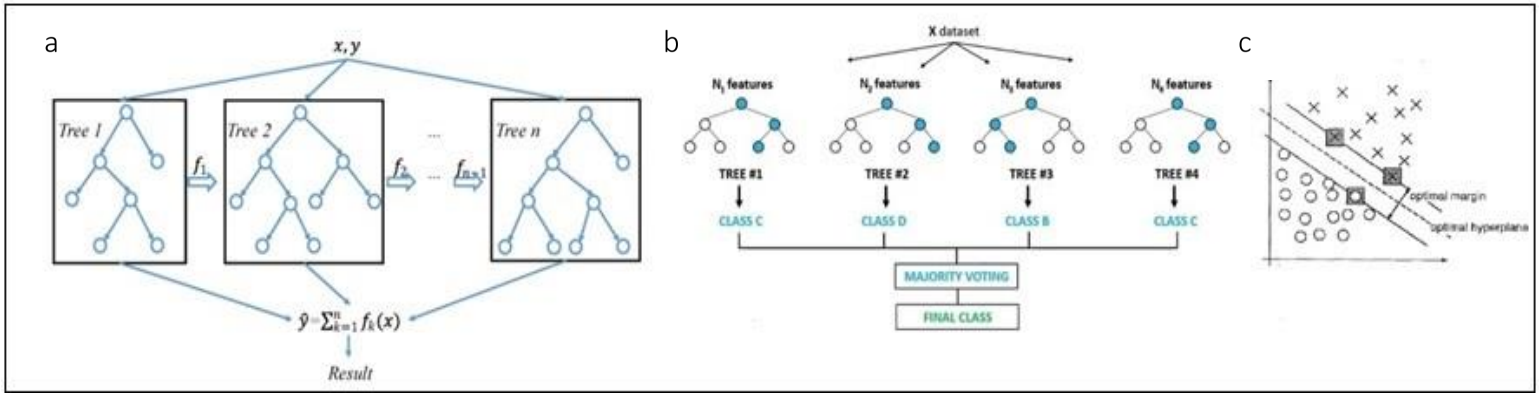


Fig 3.13: from left to right, a: XGBoost. b: Random Forest c: SVC [27] [28] [29]

Furthermore, for the next step, as shown in Fig. 3.10, the VGG-16 and Residual Network, RESNET50 [36], models were fine-tuned. ResNet-50, short for Residual Network with 50 layers, is a convolutional neural network architecture. Developed by Microsoft Research, ResNet-50 is known for its deep structure which allows it to extract more complex features from input data.

In fine-tuning, an important technique in transfer learning, the pre-trained model weights were allowed to be updated on a new dataset. In this case, the weights of VGG-16 and RESNET-50 were updated for our dataset. However, in transfer learning part, the pre-trained weights were frozen, and just new layers or head layers went through training on a new dataset.

In the fine-tuning section of Fig. 3.10, the image from the detection part was reshaped and then sent through the VGG-16 network as the feature extractor. Then the VGG-16 architecture's output up to the last MaxPooling layer in the last block before dense layers (block5_pool1) have been taken, which is a tensor, Then, after reshaping this tensor into a one-dimensional vector, it went through the dense layers for training (Fig. 3.14). Dense layers in a neural network are responsible for fully connecting every neuron in one layer to every neuron in the subsequent layer, allowing for comprehensive information exchange and feature transformation. Five dense layers have been used, with the Selu activation function (section 3.3.4) for the first four layers, and for the last one, a three-node layer with the Softmax activation function has been implemented. Additionally, the same process has been implemented in the RESNET-50 model with the Relu activation function in the first four dense layers (Fig. 3.15). Indeed, the hyper-parameter selection for the fine-tuning of VGG-16 and RESNET-50 was covered in Section 3.3.4 and Table 2.

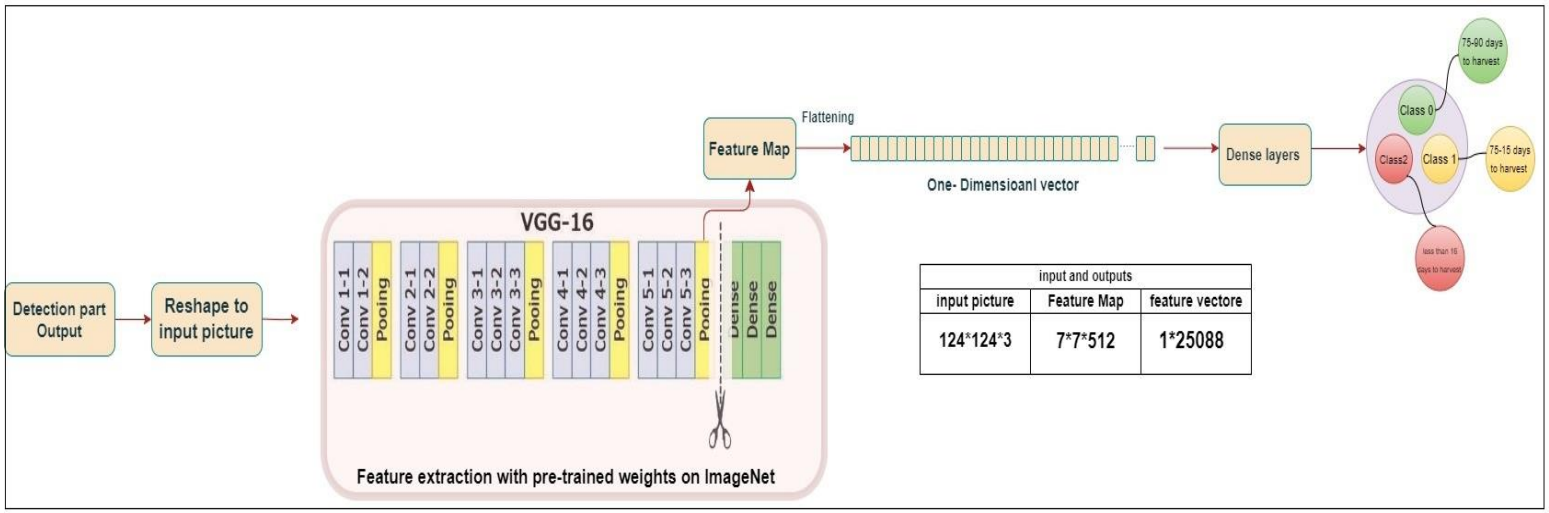


Figure 3.14: VGG-16 fine-tuning

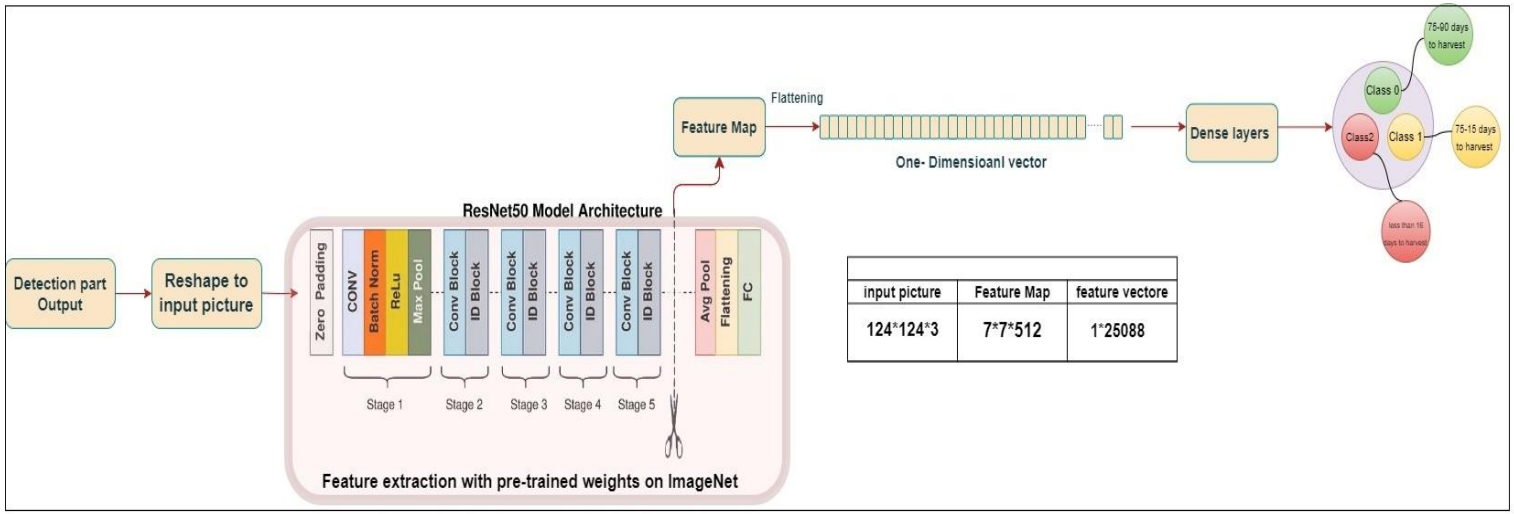


Figure 3.15: RESNET-50 fine-tuning

3.3.3 Custom Model Architecture

In the last step in the Fig 3.10, a custom CNN model was designed. designing a custom convolutional neural network as the last step in classification part of DRCAP has been considered. In this chapter the details and architecture of custom model and the steps in designing process were covered.

Designing a Convolutional Neural Network (CNN) for an image dataset involves several important considerations to create a well-performed model. First of all, dataset size and diversity play a crucial role since the larger and more diverse the data set, the wider the range of patterns and variations in the images, which leads to a better-generalized model. However, most of the time, to reach a rich feature map extraction, it is essential to go for a deep model that requires more data and resources, so a trade-off and balance in the depth of the model must be applied. In other words, if the model uses deeper architecture it can extract complicated patterns and structures but is more likely to overfit and longer running time Furthermore, pre-processing techniques were another factor in terms of applying resizing, normalizing, and data augmentation if required. Additionally, deciding the number of layers, filter sizes, and type and size of pooling layers were challenges in the design process. Furthermore, one of the most important steps was hyper-parameter selection in terms of the activation functions and normalization steps like batch normalization. As a result, with all the mentioned considerations and factors, we came up with a design for the DRCAP classifier as shown in Fig. 3.16. It shows the custom Convolutional Neural Network in the DRCAP classifier with seven convolutional layers attributing the input image to three classes.

The model starts with a 124*124-pixel image in three channels of red, green, and blue and continues with five max pooling layers in five convolutional blocks. The first and second blocks include two convolutional layers with batch normalization after each convolutional layer and a Max Pooling layer at the end of each block. The third and fourth blocks each contain a convolutional layer, followed by a batch normalization layer and a Max Pooling layer. Finally, after implementing a flattening layer to the feature map, the feature vector was achieved as the input to the following dense layers with two dense layers and a Softmax with three nodes

at the end (3.3.5). It is beneficial to consider hyper-parameter selection considerations before illustrating the classifier architecture.

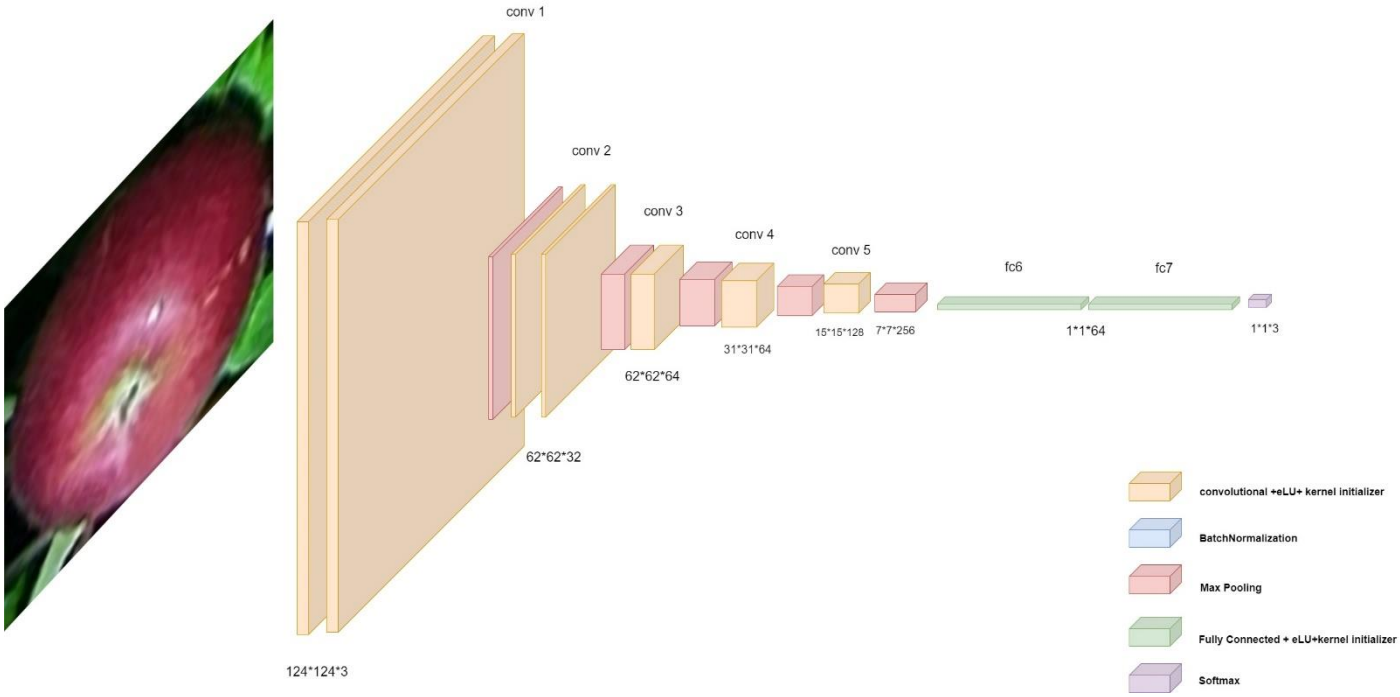


Figure 3.16: Custom Convolutional Network Model Architecture

3.3.4 Hyper-parameter selection

Parameters such as weights and biases in a CNN are internal structural variables acquired from the training data during the training process. In other words, the model's weights and biases are adjusted during training to minimize a mathematical function known as the loss function. The loss function quantifies the model's performance by measuring the difference between the model's output and the given input data. Furthermore, the choice of the loss function is an important factor in designing a model, as different problems may require different loss functions [37]. In the context of DRCAP, categorical cross-entropy is used as the cost function for the system.

Unlike parameters, hyper-parameters are not trainable in the training process but play a crucial role in the performance of a model, such as avoiding overfitting or under-fitting. They are external configurations for the model and are not learned from the data during training.

Overfitting mostly happens when the weight model brings the noises or outliers to attention rather than the patterns and structures of the data. To put in a vivid picture, the model memorizes the coordination of apples instead of understanding the subject matter.

On the other hand, under-fitting just comes across as a superficial model that can only be implemented on simple structures and has no capability of extracting more advanced patterns and structures from data. Therefore, having a balanced approach to dealing with data is the cornerstone of developing an effective and general model. As a result, in DRCAP, a variety of strategies and approaches have been taken to address these issues in terms of hyper-parameter selection. Some considerations in overfitting or under-fitting, class imbalances, resource constraints, and vanishing and exploding gradients have been taken as much as possible.

In the DRCAP classifier model, to mitigate the issues mentioned above, several key factors have been taken into account. Firstly, we applied weight initializers, specifically variance scaling in convolutional layers with fan average mode and the He uniform in dense layers, to ensure stable training. Weight initializers are methods that provide initial values for the weights in the network, ensuring that the network starts with suitable values to improve convergence during training. Additionally, the training process conducted visual inspections to identify any anomalies in validation performance. Not only that, but the early stopping method was another factor considered in the system for avoiding under-fitting or over-fitting with monitoring accuracy degradation and inspecting any signs of accuracy loss.

Besides, monitoring and visualization, undoubtedly, was a critical feature that led to add some key details to the classifier like variance scaling, batch normalization, and regularization by monitoring the validation accuracy and learning rate curve. In addition, during the training process and validation process of the DRCAP classifier, cross-validation method was employed to inspect the model performance. In this method, our sub-datasets were divided into several sub-datasets that the model gets validated in one and trained on the rest, depending on the division number, in order to reach as well-performed as possible of the model.

Early stopping is a technique in machine learning that monitors a model's performance on a validation dataset during training and stops the training process when the performance does not show improvement, preventing overfitting. It helps find an optimal balance between

model complexity and generalization by stopping training when further iterations are likely to lead to worse performance on unseen data.

Furthermore, activation functions played an influential role in detecting more complex and deep patterns in the data structure by introducing a non-linearity to the model in terms of controlling the output of the nodes or neurons. In choosing an activation function for DRCAP's classifier, several factors were considered, like gradient anomalies, e.g., disappearing or exploding gradients, output range, etc. A variety of activation functions, such as ELU, SeLU, ReLU, Leaky ReLU, and Sigmoid were implemented and tested alongside with other hyper-parameters in DRCAP. For instance, ELU is a smooth and continuous function with a non-zero gradient for all inputs and the ability to handle vanishing gradients. This can help with smoother convergence during training and may result in a more stable optimization process, especially with variance scaling.

Another technique to improve training quality was batch normalization in the DRCAP classifier. Batch normalization is a technique in deep learning that normalizes the activations of each layer during training by calculating and applying statistics for each mini-batch of data [43]. In other words, batch normalization can improve the stability and performance of each layer while achieving higher learning rates when needed. Additionally, an increase in the robustness and generalization of the model is also achievable.

Indeed, one of the most important aspects of a CNN is the optimization of the system that in DRCAP several optimizers were implemented such as Stochastic Gradient Descent (SGD), Adagrad, Adam, and Adamax. Optimizers in Convolutional Neural Networks (CNNs) are algorithms that update model parameters during the training process to minimize the loss function, such as stochastic gradient descent (SGD) or the Adam optimizer [38]. In addition, different optimizers use different strategies in converging the model efficiency.

Last but not least, the tuning of hyper-parameters is the cornerstone of all the previously mentioned points, as CNNs generally resemble a black box that works as a team. More precisely, a combination of elements must be entangled to reach an optimum point, and all of the hyper-parameters package resembles a gear box that every gear must be in coordination with others.

It can be assumed that the driving force of all hyper-parameters is the mathematical and statistical concepts and equations which would be enough for hyper-parameter selection,

which of course is mostly true, but in most cases, fitting the hyper-parameters requires an advanced understanding of the data structure, data complexity, and software and hardware limitations.

In the case of the DRCAP classifier, hyper-parameter tuning played a prominent role, especially activation function-variance scaling-optimizer triple sets. Some of the most important Hyper-parameters used in the DRCAP classifier, VGG-16, and RESNET-50 are shown in Table 2. The important hyper-parameters that were tested for the VGG-16, DRCAP, and RESNET-50 networks can be seen in Table 2. For VGG-16 and RESNET-50, the following were chosen as the hyper-parameters: Adam as the optimizer, categorical cross entropy as the loss function, and SELU with Softmax as the activation functions for dense layers. Indeed, ELU and Softmax have been considered for activation functions for convolutional layers and dense layers, respectively.

Table 2. DRCAP, VGG-16 (fine-tuned), and RESNET-50 (fine-tuned)tested hyper-parameters for more than 25 times running for each

DRCAP, VGG-16, and RESNET-50 important tested hyper-parameters		
Activation function	**Optimizer**	**Shuffling And Cross-Validation**
Exponential Linear Unit (ELU)	Adamax	
Scaled Exponential Linear Unit (SELU)	Adam	
rectified linear unit (RELU)	Stochastic gradient descent (SGD)	
Leaky Rectified Linear Unit	Nadam	
	Adgard	

Shuffling and cross-validation in Table 2 refer to the dataset from which the test set was chosen by shuffling, randomly reordering the examples in the dataset. It is crucial to note that after shuffling, the data set, including the training set, validation set, and test set, was the same for the custom network, fine-tuned network, and transfer learning-based networks to ensure that testing and training situations were equal to all of them. Indeed, the best performance in cross-validation was considered for VGG-16 and RESNET-50. It is beneficial to remind, 5% and 10% of the dataset were considered test sets. Moreover, the shuffling, training, and testing cycles for each network and technique were approximately 25 cycles. It meant 25 shufflings were applied to the dataset, and the networks were trained and tested on each.

3.3.5 Regression

Regression serves as a powerful tool for predicting continuous outcomes in image analysis, which permit the model to move beyond classification tasks to predict specific numeric value. In the big picture, regression is a statistical method that quantifies the relationship between variables and can be varied based on the application and goal of the regressors.

In DRCAP, we have used two approaches for the regression task. First, as shown in Fig. 3.2, a regression system has been developed with the VGG-16 as the feature extractor using the transfer learning method. Here, the VGG-16 extracts features from the input images, which come from the detection part of the DRCAP, and makes a feature vector to send to the classifiers for training. Additionally, four classifiers have been employed in classification for this purpose: XGBoost, Random Forest, Bayesian [39], and Theil-Sen [40]. A Bayesian regressor is a type of regression model which uses Bayesian statistics to estimate relationships between variables by probability distributions[41]. Theil-Sen is a statistical method that estimates the slope of a linear relationship between variables. Unlike traditional linear regression, Theil-Sen

is less sensitive to outliers, which makes it a useful method in situations where the data may contain outliers [42]

The second regression task was designing a custom regressor for this part, an ANN, which was done by introducing several dense layers after implementing the custom CNN classifier of DRCAP as the feature map extractor for the input image. The custom regressor includes one input layer, which is the feature vector, and three hidden dense layers with ELU and He Uniform as the activation function and kernel initializer, respectively. The output dense layer included an output neuron. Furthermore, Mean Square Error (MSE) and Adam were used as the loss function and optimizer, respectively.

One of the reasons for introducing a regressor part in the analysis part of the DRCAP was the mid-class error. This means that the input image after the detection part went through the classifier and took a label as a class of it, which gave the operator a sense of yield prediction. However, to cover the gaps in the classifier error in terms of classifying the images in mid-class ranges, a regressor was introduced. More specifically, some apples could be in mid-class groups, and that could be both in two classes, especially in ripening time, and add a systematic error to the analysis. Precisely, when an apple grows, it goes from the unripe class to the ripe class, but in transient time, it can be classified in both classes. To address this issue, the regressor can attribute continuous numbers instead of discontinuous class numbers. It must be considered that regression came up as an extra help to the classifier, especially in mid-class cases that start from zero up to around three, indicating the ripeness class. Indeed, the continued numbers in the regression zone gave a sense of an apple label in the growing stage. As an example, if the regressor gave a 1.3 on a scale of zero to three, it was an indication of being an apple in class 1 but at the same time reaching class 2, which means it is a mid-class apple (Fig 3.17).It must be addressed that the image stage belongs to the transient time from class 1 to class 2 based on the dataset split (Table 1).

3.3.6 Image processing

The last but not least step in the DRCAP analysis section belongs to the image processing part, which works based on the image size of the given input. The aim of this method was first to find a suitable average pixel dimension for each class and, second, to find a pixel density of the apple images. Several apples have been tracked since blossom time with specific time temporals: class 0 for unripe apples (up to July 10), class 1 for semi-ripe apples (from July 11 to August 15), and class 2 for ripe apples (from August 16 to September 14), in terms of their weight and images. Those tagged apple weights have been recorded at harvest time.

Precisely, seven apples were tracked from their first fruiting time up to harvesting time which one of them was shown in Fig. 3.18. To monitor the growth correlation of apples, a chart of pixels growing has been plotted, with the X-axis representing the length of the picture in pixels and the Y-axis representing the width of the image in pixels as shown in Fig. 3.19. Sample 1 and sample 2 refer to the two different apple variants from two different locations in the orchard where the pictures were taken. Four of the seven tracked apples belong to the sample 1 and three belongs to the sample 2. These apples have been tracked manually from the pictures of main dataset and cropped as accurately as possible during the growing process. Then, an average of the length and width of the pixels of cropped apple images in each stage or class has been calculated as shown in Table 3. Table 3 shows the detailed average pixel thresholds for each class of both samples.

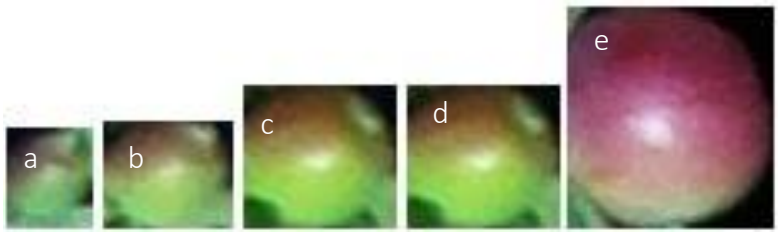


Figure 3.18: Growth monitoring of a specific apple in time.

a: 2021-06-25, b: 2021-07-04, c: 2021-07-28, d: 2021-08-21, and e: 2021-09-14

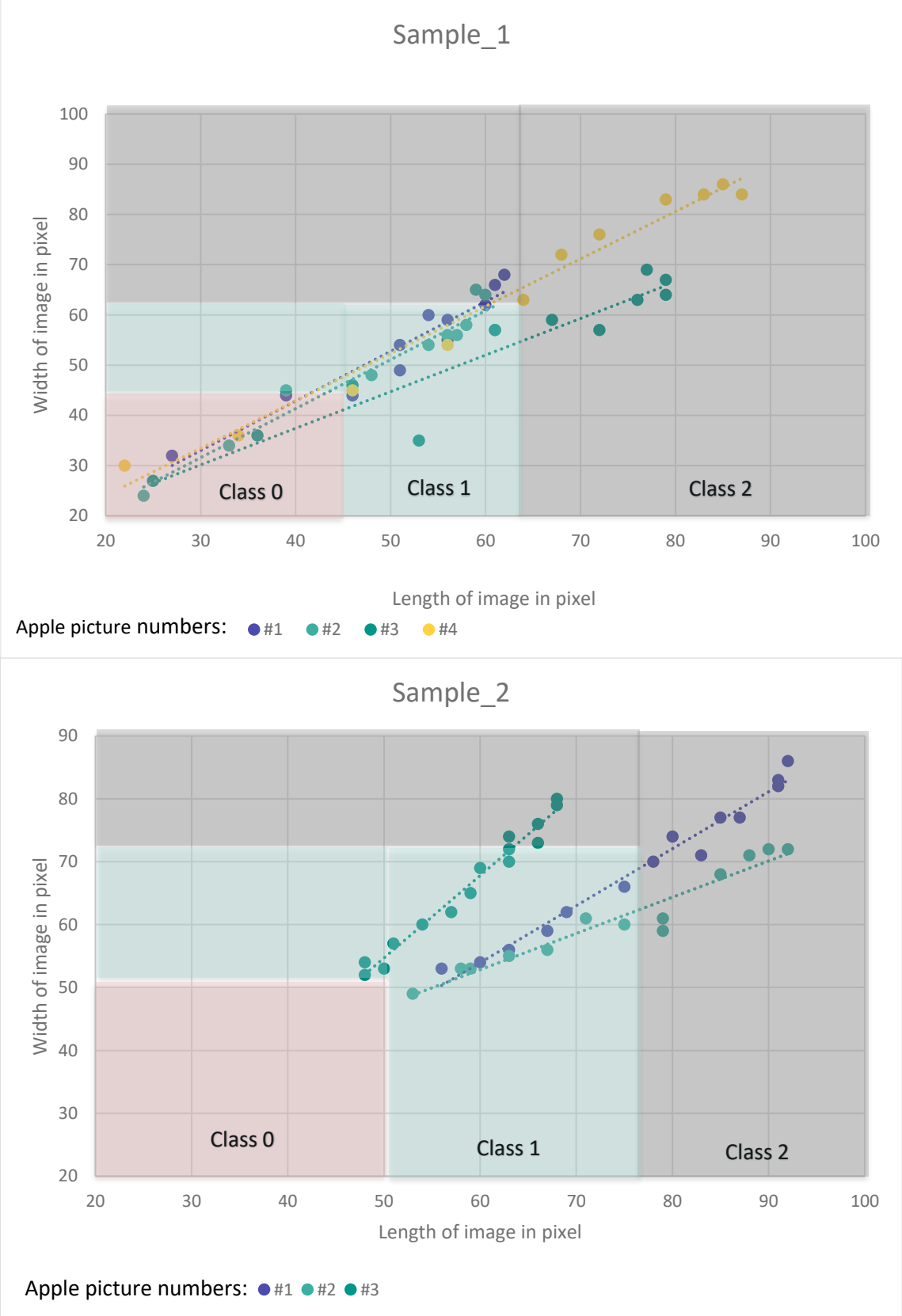


Figure 3.19: Four apples class-change chart in the growing period for Sample 1 and three apples class-change chart in the growing period for Sample 2

Table 3: Pixel thresholds for classes of samples

sample 1 class pixels			sample 2 class pixels		
0	1	2	0	1	2
45*45	46*46	67*63	52*51	53*52	78*73

For the average pixel density, a more reliable approximation of the number of pixels of an apple in an apple image was calculated in the GNU Image Manipulation[43] (Fig. 3.18). Therefore, for all the tagged apples, the same process has been applied. It has been tried at this stage to capture the pixels of the apple as accurately as possible to find a density for it. For instance, in Fig. 3.21, the apple pixel was 6058 pixels, and since we have its weight, by dividing its weight by the number of pixels, a pixel density can be extracted. The same method was used on the tagged apples, yielding both a class pixel density and an average pixel density (Fig. 3.19).

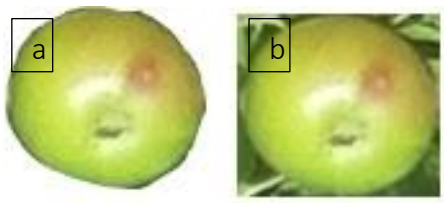


Figure 3.20: fine cropping of apple pixels a: after fine cropping, b: before fine cropping

Figure 3.21: pixel growing of a tagged apple, the numbers below each picture represents the pixels of the cropped picture

Chapter 4

Results

This chapter includes two sub-chapters, detection and analysis, presenting the results of the study for the detection and analysis parts. In the detection sub-chapter, the results of apple detection were covered, and the analysis sub-chapter covers the results in three micro-chapters for classification, regression, and image processing.

4.1 Detection

Aa shown in Table 4 different versions of YOLO (YOLOV5, YOLOV7, and YOLOV8) have been trained on three sub-datasets in 108 training cycles as mentioned in Fig 3.8 and section 3.2. It is crucial to note that the test set for different versions of YOLO on a particular sub-dataset was the same to have an equal evaluation situation for all versions. For the 108-training cycle, the YOLOV7 performed with best accuracy of 82% compared with 76% best accuracy for YOLOV8 and 78% best accuracy for YOLOV5 (Fig. 4.1).

Table 4: YOLO version performance in detection part in different sub datasets

	YOLOv5			**YOLOv7**			**YOLOv8**		
	Sub-set1	*Sub-set2*	*Sub-set3*	*Sub-set1*	*Sub-set2*	*Sub-set3*	*Sub-set1*	*Sub-set2*	*Sub-set3*
Best Accuracy	72%	72%	78%	72%	76%	82%	70%	74%	76%
Average Accuracy	76%			79%			73%		

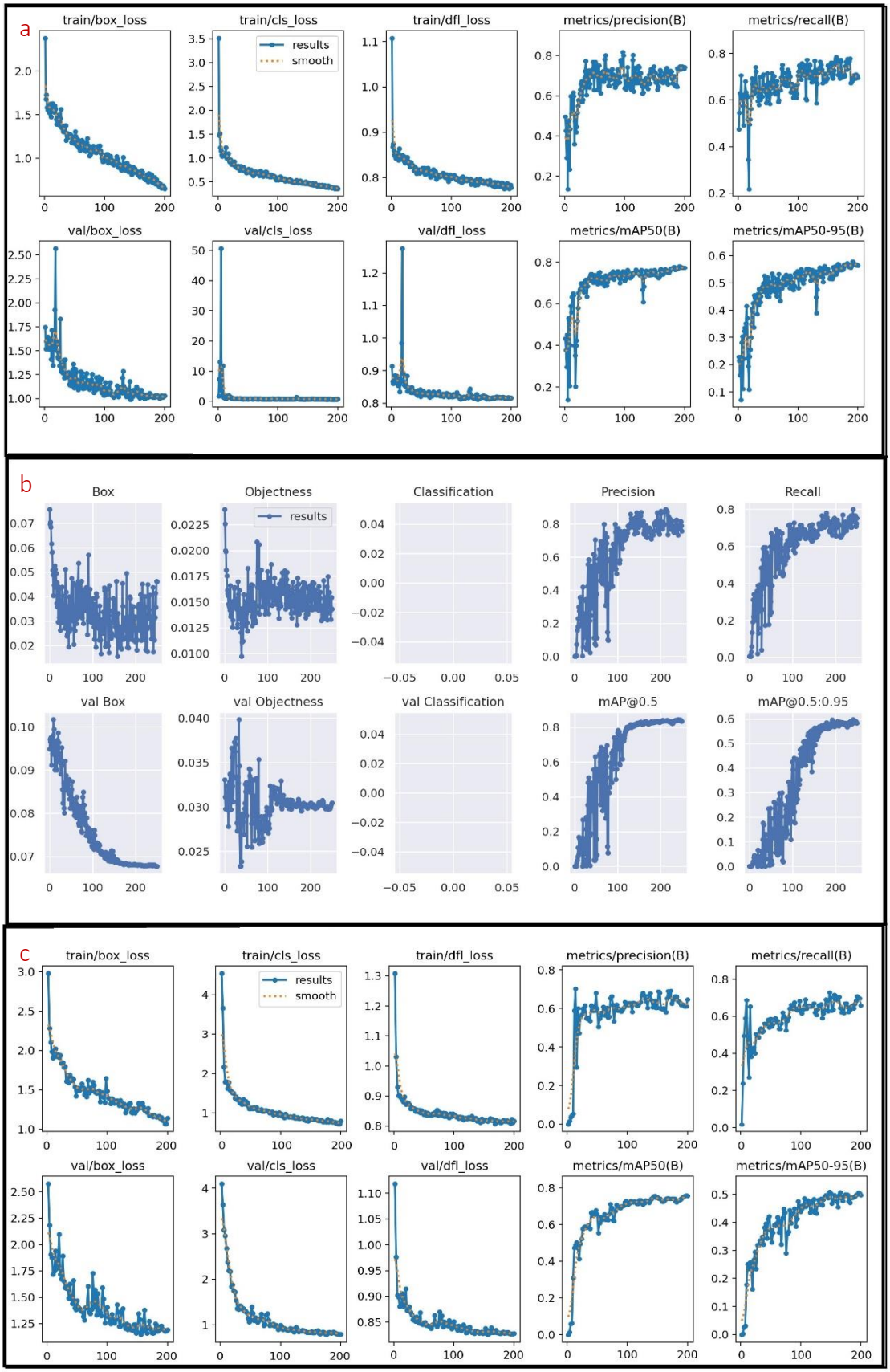


Figure 4.1: YOLOv5 results: first double-row (box a); YOLOv7 results: second double-row (box b), YOLOv8 results: third double-row (box c).

Fig. 4.1 and metrics are essential for evaluating the performance of YOLO versions during training, validation, and testing phases. They give insights into how well the model is learning to detect objects, classify them, and generate accurate bounding boxes. Therefore, it would be beneficial if we went through each chart and what it showed us.

The train/box_loss box presents the box loss during the training process which is a measure of how well the predicted bounding box coordinates align with the ground-truth bounding box coordinates. The val/box_loss box presents the same information, but for the validation process. Comparing these two boxes for all versions gives us information about the over- or under-fitting of the training process. Since there is no divergence in the trend of them, an overfitting issue, at least with this factor consideration, is not probable. Also, as shown in the results for the YOLOv7, the training loss drastically drops to 0.02 around 50 epochs, which is significant in comparison to the YOLOv5 and v8.

Moreover, the train/cls_loss figure represents the classification loss during training. It indicates how well the model is performing in terms of correctly classifying objects in the images. The val/cls_loss box presents the same information but for the validation process. The train/dfl_loss stands for "Direction-Focus Loss," is a type of loss function that improves object detection performance, especially in multi-class object detections. However, since our case only involves one class, the apple class, this chart would not be as important as others. But, the use of focal loss can still be beneficial in terms of optimizing the YOLO algorithm to learn and improve its ability to detect objects in the single-class dataset. The val/dfl_loss box presents the same information but for the validation process.

Furthermore, metrics/precision (B) is a metric that measures the accuracy of positive predictions made by the model. The "(B)" indicates that it is calculated on a per-batch basis. More precisely, by changing the batch size of the process, this factor will change; therefore, finding an optimum batch size could be a crucial point to achieve a high-accuracy weight function. The metrics/recall (B) can be interpreted as how well the YOLO model is capturing relevant instances of the object in each batch of the test set. This metric can help us understand the performance of the model in terms of sensitivity and how well it is identifying positive instances, or objects, out of all the actual positive instances in each batch. Comparing the results on the recall boxes of the three versions shows that the YOLO v5 and v8 recalls reach a higher number than YOLO v7 in fewer epochs. However, it drops significantly soon and struggles to reach the same level in the rest of the process, and after 200 epochs, it ends in

the same number with a saturated and flat chart. But Yolo v7 reaches its peak with a slighter sloop and remains there with a less fluctuated trend.

The metrics/mAP50 (B) refers to the mean average precision (mAP) at an intersection over union (IoU) threshold of 0.5, and the calculation is performed on a per-batch basis in the context of your YOLO model evaluation. The IoU (Intersection over Union) is a measure of the overlap between the predicted bounding box and the ground truth bounding box, or actual bounding box. The last but not least, metrics/mAP50-95(B) box refers to IoU thresholds from 0.5 to 0.95 on the test set, calculated on a per-batch basis.

Moreover, the F1 score of each version has been shown in Fig. 4.2. This score is a single metric that provides a balanced assessment of a model's ability to correctly classify positive instances while minimizing both false positives and false negatives. In Fig. 4.2 the F1 score was plotted in relation with the confidence threshold. This confidence score represents the model's confidence that the predicted bounding box contains an apple.

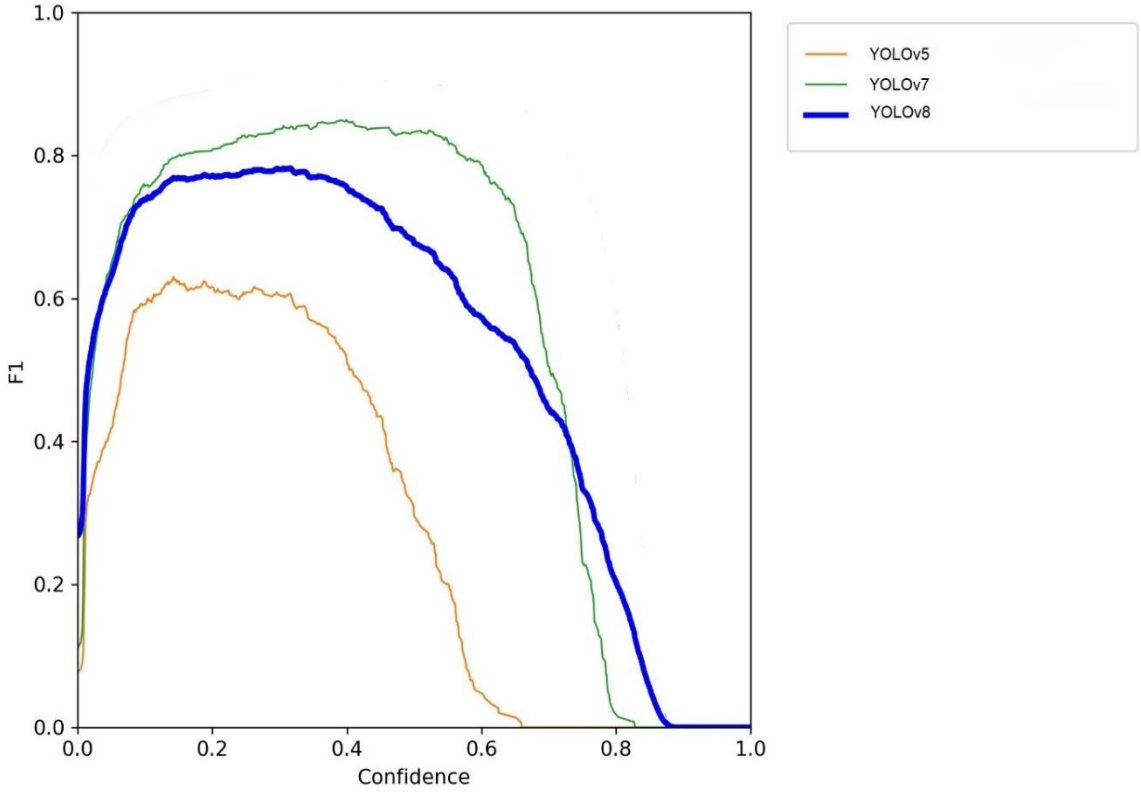


Figure 4.2: F1 score comparison chart of YOLO versions

Fig. 4.2 suggests that, for our specific dataset, YOLOv7 performs better in terms of F1 score, and the optimal trade-off between precision and recall occurs at a confidence threshold of 0.45. It must be included that the choice of the best model and confidence threshold should align with the objectives and constraints of case. For instance, if we prioritize minimizing false positives (higher precision), we might choose a higher confidence threshold. If we prioritize capturing as many true positives as possible (higher recall), we might choose a lower confidence threshold. Therefore, a trade-off must be considered and the F1 score can be an option to find a balance between these two goals.

In addition to the test set, several pictures from different apple tree variants from different locations in Norway under less desirable illumination and contrast have been taken to test how well the final weight of YOLOv7 acts. Before that, it would be beneficial to provide examples of improper augmentation, such as saturated noise, extreme zoom in/out, improper cropping, and extreme blur, that affected the results in sub-dataset1 and sub-dataset2, as shown in Fig. 4.3. Having a look to the results of the best weight models of YOLOv5, v8 and some instances before going to the YOLOv7 results would make it more transparent. The Fig. 4.4 shows the output of best weight models of YOLOv5 and v8 applied on an image from our 2022 main image set which was totally intact for all models and methods. Indeed, Fig. 4.5 shows YOLOv5 and v8 outputs on the images from different apple variants for the best F1 score.

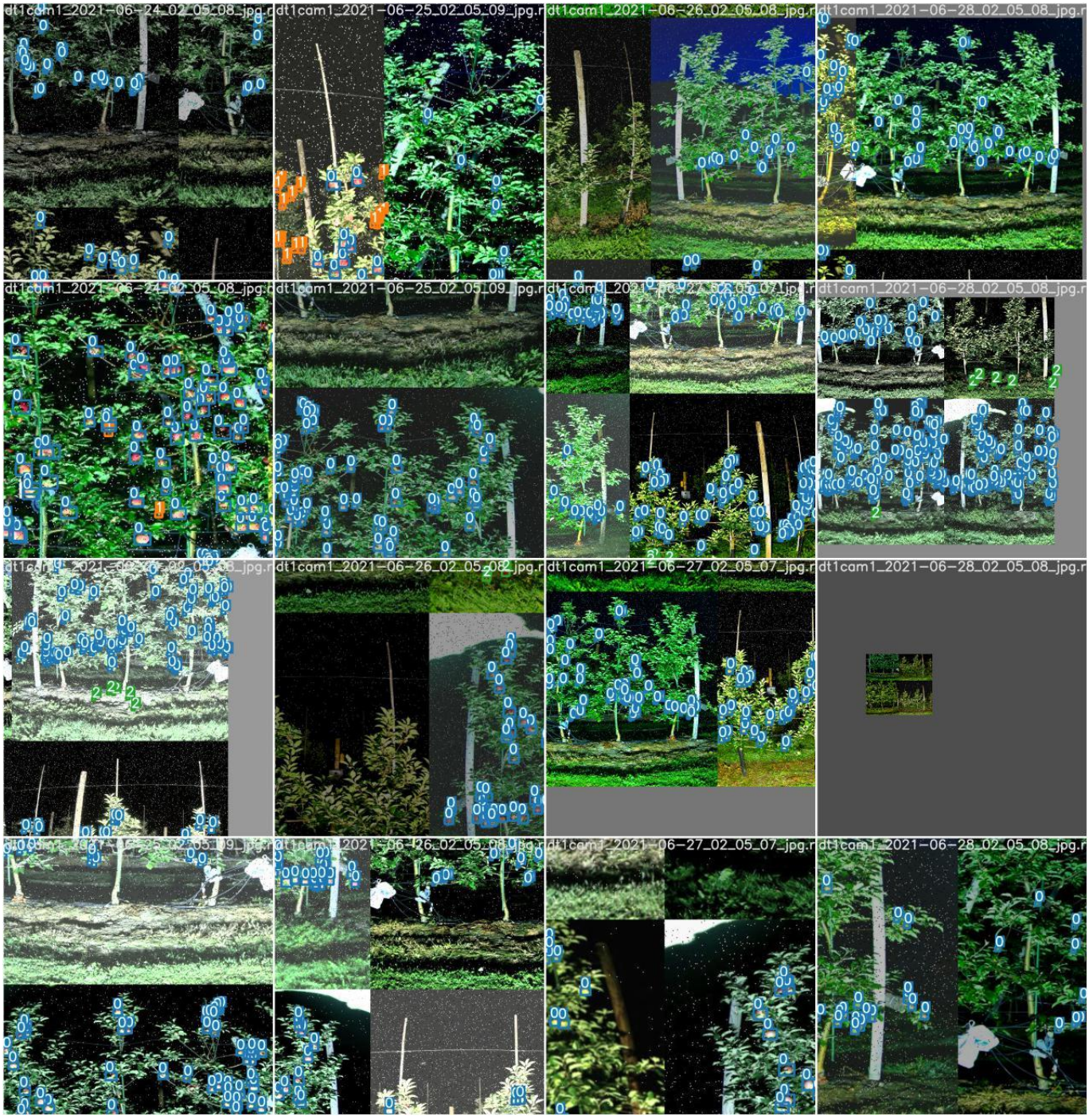


Figure 4.3: Random augmentation examples on sub-dataset 1 and sub-dataset 2.

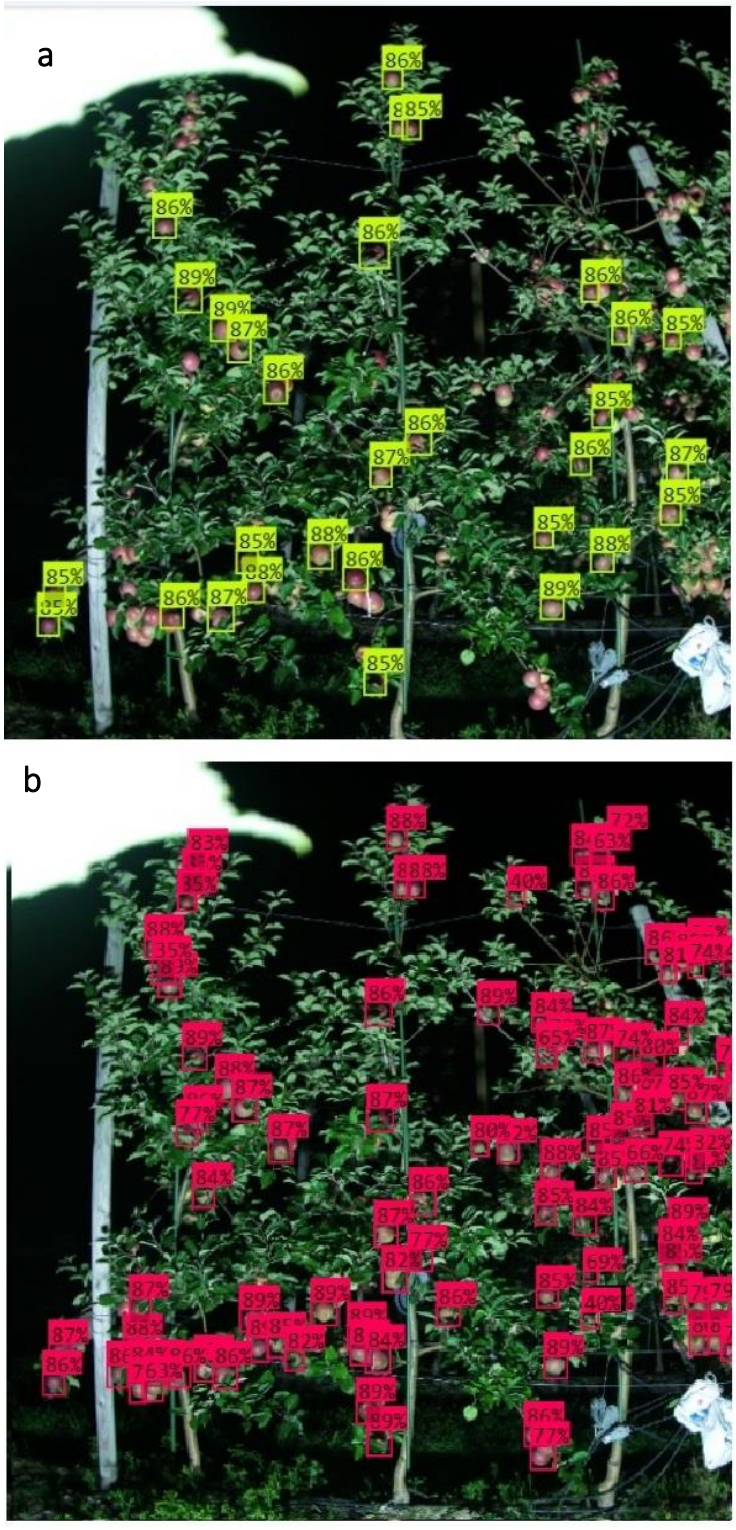


Figure 4.4 a: YOLOv5 best weight output, b: YOLOv8 best weight output for best F1 score

Click for better resolution

Figure 4.5: a: YOLOv5 best weight output; b: YOLOv8 best weight output for best F1 score

The Fig. 4.6, 4.7, and 4.8 shows theYOLOv7 best weight output for the best F1 score in an image from 2022 main image set from class 0, class1, and class2 respectively. Indeed, the Fig. 4.9 shows theYOLOv7 best weight output on different apple variant images.

Figure 4.6: the YOLOv7 best weight output on a class0 image for best F1 score

Click for better resolution

Figure 4.7: the YOLOv7 best weight output on a class1 image for best F1 score

Click for better resolution

Figure 4.8: the YOLOv7 best weight output on a class2 image for best F1 score

Click for better resolution

Figure 4.9: YOLOv7 best weight output on different apple variant images for best F1 score

[Click for better resolution]

4.2 Analysis

The analysis section includes three sub-chapters including classification, regression, and image processing in which the results will be presented separately.

4.2.1 Classification

Table 5 shows the classifiers used in the classification part of the DRCAP in custom CNN, Fine-tuning, and Transfer learning concepts. They have been evaluated based on their accuracy and loss. This table shows that the Custom Classifier of DRCAP performed better than other classifiers with a 98.3% accuracy and 0.5 loss. Based on these results, several outputs of the DRCAP custom CNN classifiers were presented in Fig. 4.10 with the only false class prediction in the Fig. 4.10(g). Moreover, the training curve of the model have been shown in Fig. 4.11. Furthermore, the heatmap of the confusion matrix has been provided in Fig. 4.12.

Table 5: DRCAP classifiers results

	DRCAP Classifiers results					
	Custom CNN	Fine-tuning		Transfer Learning		
		VGG-16	RESNET-50	XGBoost	R-Forest	SVC
Accuracy (%)	98.3	90	83	87	89	91
loss	0.05	0.58	0.86			

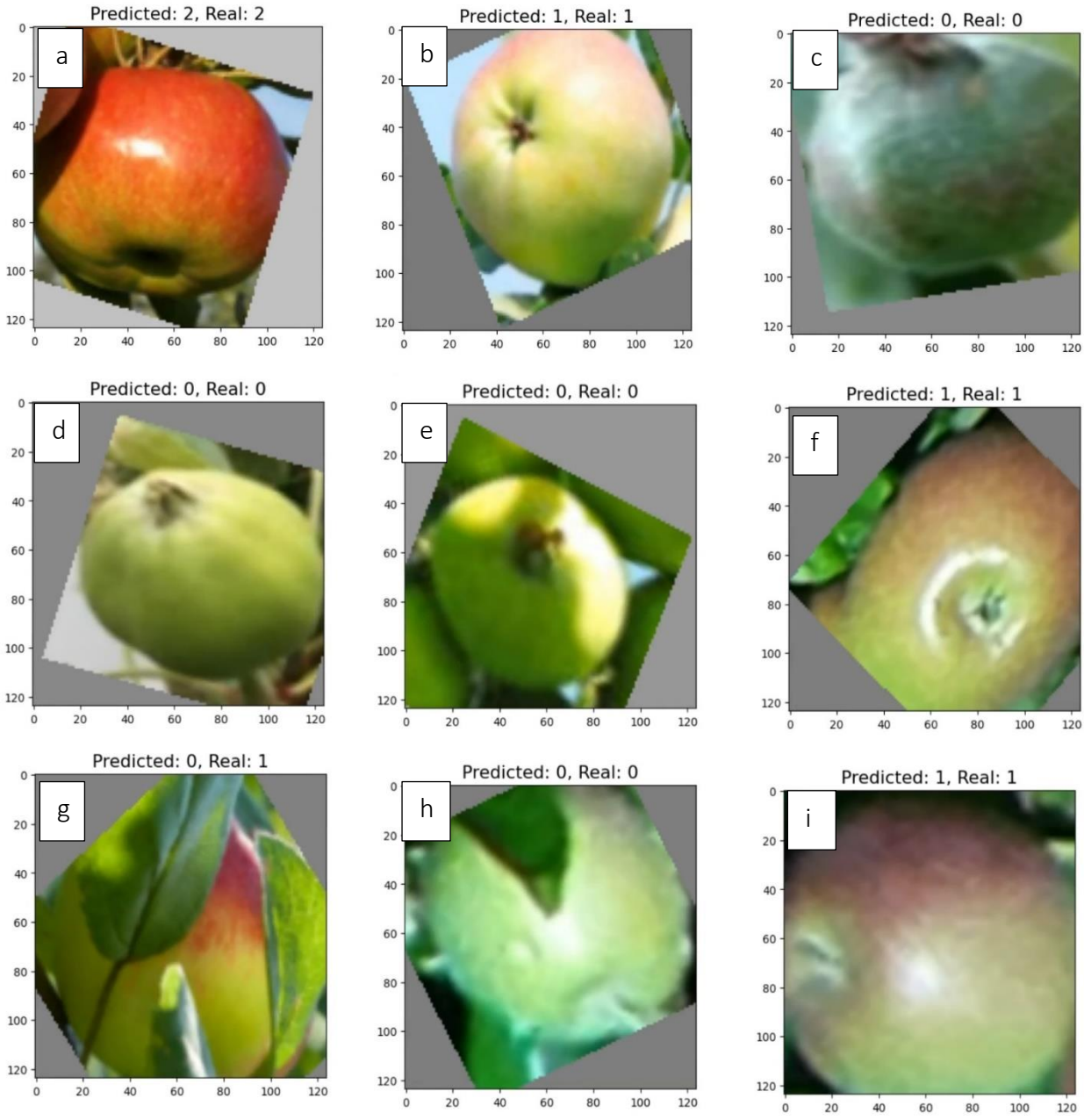


Figure 4.10: predicted and real classes of the output of the DRCAP custom classifier model

Click for more pictures (video of all test images)

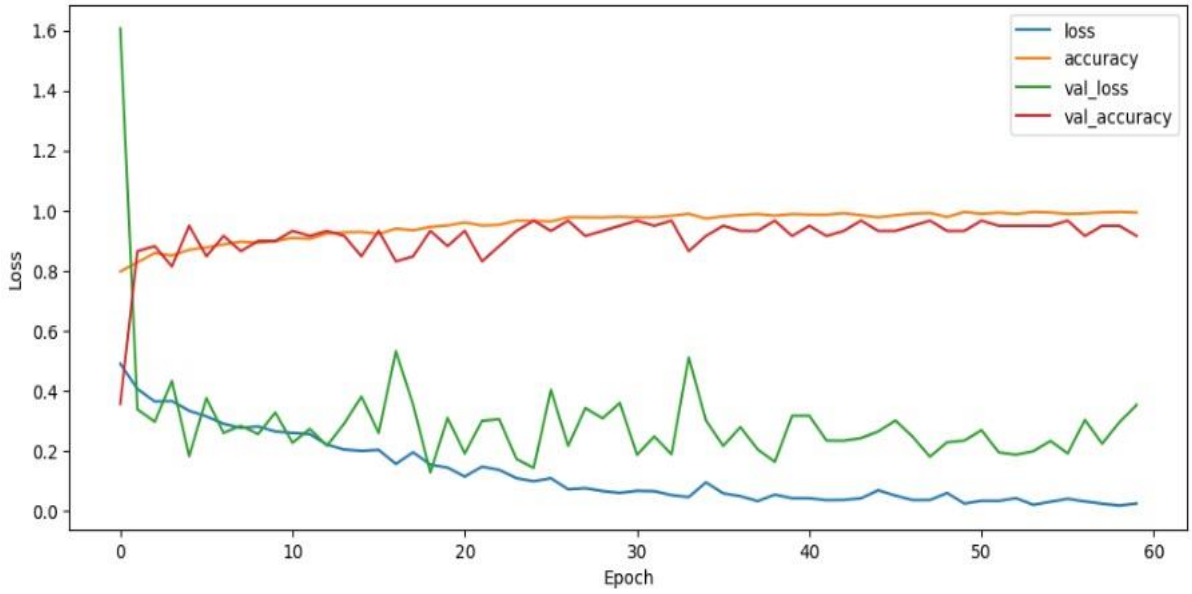


Figure 4.11: Training curve of the DRCAP custom classifier model

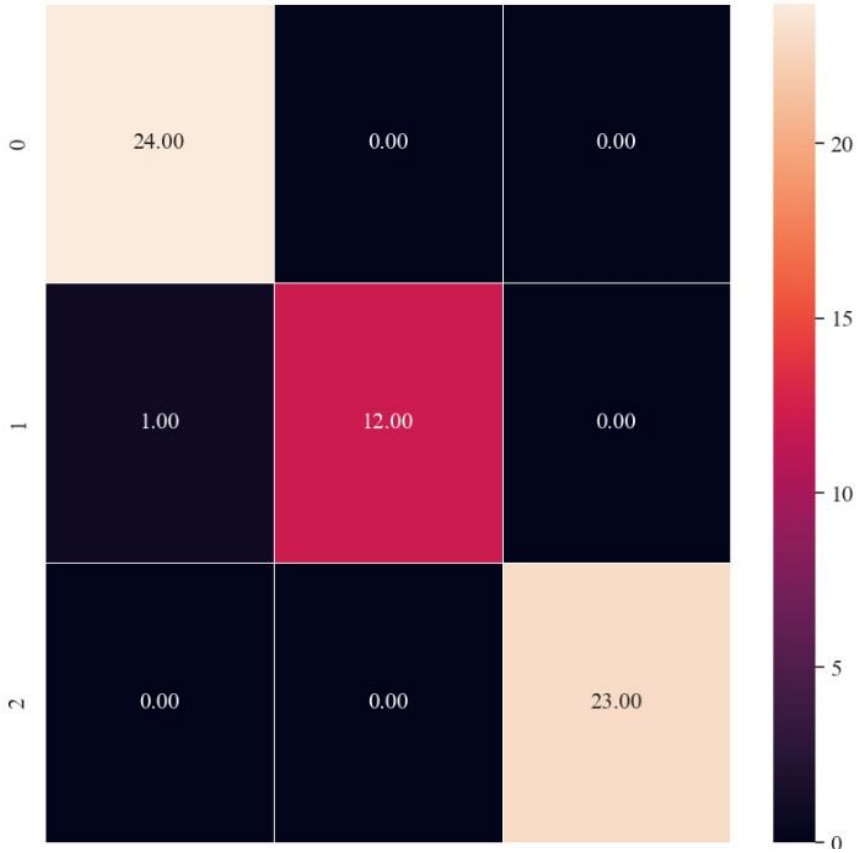


Figure 4.12: Heat map of confusion matrix of the output of DRCAP classifier

4.2.2 Regression

According to Fig. 3.1, the regression part of the DRCAP included two sections. Table 6 compares them in terms of R-squared, MAE, and running time spent on test set which was around 80 images. A relatively higher MAE and R-squared results have been demonstrated by the Custom ANN regressor model with a less than a minute computation time, followed by Theil-Sen as the second-best performer.

Table 6. Regression task detailed results

	Regression				
Feature extractor	VGG-16				Custom CNN model
Regression Model	XGBoost	Random Forest	Theil-Sen	Bayesian	Custom Regression Model
R-Squared	86	81	91	87	94
MAE (Mean average Error)	0.56	0.18	0.11	0.22	0.13
Running Time(minutes) on test set	17	15	18	21	<1

4.2.3 Image Processing

In the image processing part of the analysis section of DRCAP, the pixel dimension of each class has been calculated, and an average pixel density of the apple pictures has been proposed. Fig. 4.13 shows the measurements of the seven tagged apples on 14 September with an extra apple in their class 2 stages. Table 8 presents the average gram of eight apples, seven tagged apples with one more, shown in Fig. 4.13 and the average pixel amounts of the tagged apples for samples 1 and 2 on 14 September 2021, which have been captured by the GIMP software. It shows that the average of eight apples is 171.6 grams and reports at the same time that the average pixels of two samples were 5617 pixels. More precisely, the 30.6 m g/pixel is the gram of an apple image pixel, which is derived by dividing the weight of the apple by average pixel measured with GIMP software. The 30.6 m g/pixel tells us that a pixel of an apple image is 30 milligrams.

Besides, this method can be used as an auxiliary method for classification and yield prediction. According to Table 3, an average pixel dimension for each class of two samples has been calculated, and a total average for all three classes can be shown in Table 7, which was calculated just by averaging each class in both samples. According to Table 7, if the image comes out of the detection part as a cropped apple image and faces a discrepancy between the classifier and regression in attributing a class number to it, the pixel processor gives the operator another view to help in determine what class the image belongs to. It must be known that discrepancy comes mostly in the transient time of an apple's growing period.

Table 7: Average class pixels

Average class pixels		
0	1	2
48*48	49*49 to 74*69	75*70

fruit number	g/frukt
1	190
2	157
3	171
4	170
5	136
6	184
7	164
8	201
Sum	1373
Average	171.625

Figure 4.13: tagged apples for weight measuring on September 15, 2021

[Click for more details]

Table 8: Pixel density details

date		24.07.21	02.08.21	10.08.21	20.08.21	14.09.21
Gram Average						171.6
Pixel Average	sample 1	2551	3040	3673	4186	4823
	sample 2	3460	4300	5051	5940	6411
Samples pixel average		3005	3670	4362	5063	5617
density(mg/pix)						30.6
Total density per pixel			30.6 m G/Pixl			

Chapter 5

Discussion

5.1 Detection

Comparing the YOLO versions results in the detection part of the DRCAP, which clarifies several advantages and disadvantages of each model. First of all, YOLOv5, mostly known for its speed, showed 78% accuracy but came across as lacking in precision, especially for smaller objects and unseen datasets as same as the 76% accuracy of YOLOv8 (Fig. 4.5). on this dataset the YOLOv8 showed some potential improvements in comparison with YOLOv5 but faced some challenges in terms of generalization capabilities. On the other hand, the 82% accuracy of YOLOv7 showed better accuracy performance not only on the test set but also on a new and unseen dataset but its training time was more than YOLOv8. Some of the observed differences in accuracy can be attributed to several factors, including the model complexity, the training data quality, and the hyperparameter tuning which depends the data complexity. For instances, YOLOv7's Path Aggregation Network (PANet) [53] and head network contribute to its higher feature extraction and fusion capabilities compared to YOLOv5 and YOLOv8. Additionally, YOLOv7's DIoU-v5 loss function improves bounding box regression accuracy. After all these technical considerations in the architecture of each model and their benefits, the greatest jump in accuracy happened after preparing a proper dataset and training process to the network. As seen in Table 4 the achieved accuracy gap between the sub-dataset 1 and sub-dataset 3 was ten percent which is considerable. Therefore, providing a pre-cleaning and pre-processing method and dataset to the network would play a crucial role and even in some cases it can compensate the lacks in model improvements techniques [54].

5.3 Analysis

As shown in Table 5, in the transfer learning methods part of the classification section, SVC as a linear classifier, which finds hyperplanes for best class separation, performed better than R-Forest and XGBoost as ensemble models. Since the VGG-16 was the feature extractor in this part, which produces a high-dimensional feature space due to its deep architecture, SVC's upper-hand accuracy indicates there was a linearly separable class in the feature space. Therefore, lines can separate the classes in this high-dimensional space. Also, SVC is a robust classifier for outliers in comparison to ensemble models like XGBoost and Random Forest; so, the existence of outliers in feature space might be handled more by SVC than by two other models.

Furthermore, in the fine-tuning part of Table 5, the VGG-16 and RESNET-50 have been considered as classifiers instead of traditional machine learning models and algorithms like SVC, XGBoost, and Random Forest. The gap between the result of VGG-16 and the RESNET50 shows that the architecture complexity of a model, such as using more layers, cannot necessitate better results. As a result, there are other crucial factors beyond model complexity, such as hyper-parameter selections, tuning, and keeping a balanced trade-off between complexity and simplicity.

Additionally, DRCAP custom CNN classifier performance excelled others in terms of accuracy and loss. There were multiple factors to address this accuracy, First and foremost, scaling the variance of neurons played a crucial role, which was not the case in RESNET50 and VGG-16, since it was important to have a stabilized learning process that directly influences the model training convergency. The stabilization matters more, especially in deep models, since all the neurons can be considered as decision-makers for the input data, which here is the input feature. The fan average mode in variance scaling controls the overall decision for each neuron. This mode is beneficial when the input and output are significantly different. Also, employing ELU as an activation function in architecture was another factor since it was important to control the output at the same time, preventing neurons from being completely inactive and leading them to a balanced response to both positive and negative inputs.

Additionally, in the regression task in the analysis part of the DRCAP in Table 6, Theil-Sen showed that a statistical model can proceed with machine learning algorithms like Bayesian, Random Forest, and XGBoost in terms of R-squared and MAE. Theil-Sen's robustness and resistance to outliers, coupled with its ability to handle non-linearity, led it to a better position in results. However, the computation time is still the same for them, with a slight difference. But the custom regressor ANN model showed even better results not only in R-squared but also in computation time, bringing the time from more than 15 minutes for all classical models to less than a minute, which makes it a proper choice for real-time applications. The ANN's performance can be improved for several reasons. the ANN takes advantage of a wide range of customizations in the architecture and activation function, which have contributed to the ANN's ability to better fit the regression task.

In the image processing part of the DRCAP, we tracked and observed seven tagged apples from two different trees and places in the orchard. The tracked apples were chosen from different parts of the tree since, during the ripening process, some factors, like apples exposure to sunlight, would be changed. We chose apples from the top, middle, and lower parts of the trees in both samples. Based on Fig. 3.17, an obvious correlation among the apples, even from different samples and locations, is visible, which indicates that the tracked apples were growing in correlation with each other.

5.4 Future work

Several steps are still available in DRCAP to make it more specialized than before and can be considered as future steps. First of all, improving one of the YOLO versions or even designing another custom detection network will be an important step in localizing the DRCAP.

Owing to the fact that this study is part of a larger project called Digital Orchard, having a big picture of the orchard is a crucial point. Expanding the image analysis part to help the operator with real-time monitoring for crop management, as mentioned in Chapter 1, would be possible by introducing real-time-based systems. The real-time-based systems in the DRCAP

pipeline have been considered in designing shallow networks and ANNs to make them as optimized as possible. Also, monitoring the orchard for early pest detections and anomalies is possible in the image analysis section by tracking more apples from several places in the orchard. Additionally, not only fertilizer effects on trees and soil can be investigated, but also precipitation patterns and their correlation with fruit quality and volume can be understood.

Chapter 6

Conclusion

A pipeline, DRCAP, was introduced in this study for apple yield prediction. In the pipeline, firstly, the apple images were detected and cropped by YOLO (v5, v7, and v8); YOLOv7, with an accuracy of 82%, showed better performance than 78% for YOLOv5 and 76% for YOLOv8. A custom CNN classifier with an accuracy of 98.3% assigned a class to the cropped and detected images in the analysis part, outperforming VGG-16 at 90% and RESNET-50 at 83%. Indeed, an ANN regression model with an R-squared of 94% and less than a minute of running time for tests gave a comprehensive understanding of image classes, especially for the inner-class images. Other regressors like Random Forest, XGBoost, Theil-Sen, and Bayesian showed 81%, 86%, 91%, and 87%, respectively. Moreover, using an image processing technique, an average pixel dimension for three classes of apples has been calculated based on the tracking of seven tagged apples during their growing period and showed a correlation between apple growth rate in different parts of the tree.

References

[1] FoodCollection. "Unripe Apples On The Tree." fine art america. https://fineartamerica.com/featured/unripe-apples-on-the-tree-foodcollection.html (accessed 2023).

[2] M. Mandal. "Introduction to Convolutional Neural Networks (CNN)." analyticsvidhya. https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/ (accessed 11 November 2023.

[3] T. Aggarwal. " Artificial Neural Networks are to Deep Learning, What Atoms are to Matter." https://acsicorp.com/blogs/fundamentals-artificial-neural-networks-are-to-deep-learning-what-atoms-are-to-matter/ (accessed November 27,2023.

[4] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 7464-7475, 2022.

[5] A. Gupta *et al.*, "Deep Learning in Image Cytometry: A Review," *Cytometry Part A,* vol. 95, 12/01 2018, doi: 10.1002/cyto.a.23701.

[6] 123RF. "The unripe apples on apple-tree branches." https://www.123rf.com/photo_14054602_the-unripe-apples-on-apple-tree-branches.html (accessed 2023.

[7] starkbros, "Stages of Apple Tree Growth: What to Expect After Planting," 2023. [Online]. Available: https://www.starkbros.com/growing-guide/article/stages-of-apple-tree-growth.

[8] J. Gené-Mola *et al.*, "Fruit detection in an apple orchard using a mobile terrestrial laser scanner," *Biosystems Engineering,* vol. 187, pp. 171-184, 2019/11/01/ 2019, doi: https://doi.org/10.1016/j.biosystemseng.2019.08.017.

[9] T. Yu, C. Hu, Y. Xie, J. Liu, and P. Li, "Mature pomegranate fruit detection and location combining improved F-PointNet with 3D point cloud clustering in orchard," *Computers and Electronics in Agriculture,* vol. 200, p. 107233, 2022/09/01/ 2022, doi: https://doi.org/10.1016/j.compag.2022.107233.

[10] D. Mhlanga, "Artificial Intelligence in the Industry 4.0, and Its Impact on Poverty, Innovation, Infrastructure Development, and the Sustainable Development Goals: Lessons from Emerging Economies?," *Sustainability,* vol. 13, no. 11, p. 5788, 2021. [Online]. Available: https://www.mdpi.com/2071-1050/13/11/5788.

[11] R. Kler *et al.*, "Machine Learning and Artificial Intelligence in the Food Industry: A Sustainable Approach," *Journal of Food Quality,* vol. 2022, p. 8521236, 2022/05/12 2022, doi: 10.1155/2022/8521236.

[12] W. E. Forum. "The Fourth Industrial Revolution: what it means, how to respond." World Economic Forum. https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/ (accessed 10 November 2023.

[13] L. Lipper *et al.*, "Climate-smart agriculture for food security," *Nature Climate Change,* vol. 4, pp. 1068–1072, 11/26 2014, doi: 10.1038/nclimate2437.
[14] Y. Rouphael and G. Colla, "Modelling the transpiration of a greenhouse zucchini crop grown under a Mediterranean climate using the Penman-Monteith equation and its simplified version," *Australian Journal of Agricultural Research - AUST J AGR RES,* vol. 55, 01/01 2004, doi: 10.1071/AR03247.
[15] D. Wulfsohn, F. Zamora, C. Tellez, I. Lagos, and M. García-Fiñana, "Multilevel systematic sampling to estimate total fruit number for yield forecasts," *Precision Agriculture,* vol. 13, 04/01 2011, doi: 10.1007/s11119-011-9245-2.
[16] R. Gebbers and V. Adamchuk, "Precision Agriculture and Food Security. Science327(5967), 828-831," *Science (New York, N.Y.),* vol. 327, pp. 828-31, 02/01 2010, doi: 10.1126/science.1183899.
[17] D. B. Lobell and C. B. Field, "Global scale climate–crop yield relationships and the impacts of recent warming," *Environmental Research Letters,* vol. 2, no. 1, p. 014002, 2007/03/16 2007, doi: 10.1088/1748-9326/2/1/014002.
[18] J. W. Jones *et al.*, "The DSSAT cropping system model," *European Journal of Agronomy,* vol. 18, no. 3, pp. 235-265, 2003/01/01/ 2003, doi: https://doi.org/10.1016/S1161-0301(02)00107-7.
[19] C.-h. Qu, X.-x. Li, H. Ju, and Q. Liu, "The impacts of climate change on wheat yield in the Huang-Huai-Hai Plain of China using DSSAT-CERES-Wheat model under different climate scenarios," *Journal of Integrative Agriculture,* vol. 18, no. 6, pp. 1379-1391, 2019/06/01/ 2019, doi: https://doi.org/10.1016/S2095-3119(19)62585-2.
[20] Y. Zhang, J. Walker, and V. Pauwels, "Assimilation of wheat and soil states for improved yield prediction: The APSIM-EnKF framework," *Agricultural Systems,* vol. 201, p. 103456, 08/01 2022, doi: 10.1016/j.agsy.2022.103456.
[21] R. Linker, "Machine learning based analysis of night-time images for yield prediction in apple orchard," *Biosystems Engineering,* vol. 167, pp. 114-125, 03/01 2018, doi: 10.1016/j.biosystemseng.2018.01.003.
[22] S. Bargoti and J. Underwood, "Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards," *Journal of Field Robotics,* vol. 34, 10/25 2016, doi: 10.1002/rob.21699.
[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, 1 ed. (Adaptive computation and machine learning). London, England
Cambridge, Mass: London, England: The MIT Press, 2016.
[24] G. Hinton *et al.*, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *Signal Processing Magazine, IEEE,* vol. 29, pp. 82-97, 11/01 2012, doi: 10.1109/MSP.2012.2205597.
[25] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent Trends in Deep Learning Based Natural Language Processing [Review Article]," *IEEE Computational Intelligence Magazine,* vol. 13, pp. 55-75, 08/01 2018, doi: 10.1109/MCI.2018.2840738.
[26] L. Biffi, E. Mitishita, V. Liesenberg, J. Centeno, M. Schimalski, and L. Rufato, "Evaluating the performance of a semi-automatic apple fruit detection in a high-density orchard system using low-cost digital rgb imaging sensor," *Boletim de Ciências Geodésicas,* vol. 27, 01/01 2021, doi: 10.1590/s1982-21702021000200014.
[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You Only Look Once: Unified, Real-Time Object Detection*. 2016, pp. 779-788.

[28] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-Time Flying Object Detection with YOLOv8," *arXiv preprint arXiv:2305.09972,* 2023.
[29] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767,* 2018.
[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.
[31] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785-794.
[32] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001/10/01 2001, doi: 10.1023/A:1010933404324.
[33] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning,* vol. 20, no. 3, pp. 273-297, 1995/09/01 1995, doi: 10.1007/BF00994018.
[34] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed.: O'Reilly Media, Inc., 2022.
[35] SVM. "SVM." https://scikit-learn.org/stable/modules/svm.html (accessed.
[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 27-30 June 2016 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
[37] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A Comprehensive Survey of Loss Functions in Machine Learning," *Annals of Data Science,* vol. 9, 04/01 2022, doi: 10.1007/s40745-020-00253-5.
[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980,* 2014.
[39] naive_bayes, "naive_bayes." [Online]. Available: https://scikit-learn.org/stable/modules/naive_bayes.html.
[40] TheilSenRegressor, "TheilSenRegressor." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.TheilSenRegressor.html.
[41] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*, 3,Third edition. ed. (Chapman & Hall/CRC texts in statistical science). Philadelphia, PA: Philadelphia, PA: Chapman and Hall/CRC, 2013.
[42] D. S. Moore and G. P. McCabe, *Introduction to the practice of statistics*, 5th ed. New York: Freeman, 2006.
[43] GIMP. "GIMP." https://www.gimp.org/ (accessed.