Ukrainian Catholic University

Master Thesis

Raspberry quality detection in visual spectrum using neural networks

Author:
Andrii Blagodyr

Supervisor:
Viktor Sakharchuk

*A thesis submitted in fulfillment of the requirements
for the degree of Master of Science*

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2021

Declaration of Authorship

I, Andrii BLAGODYR, declare that this thesis titled, “Raspberry quality detection in visual spectrum using neural networks” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Live as if you were to die tomorrow. Learn as if you were to live forever."

Mahatma Gandhi

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

Raspberry quality detection in visual spectrum using neural networks

by Andrii BLAGODYR

Abstract

The thesis presents the raspberry quality detection approach based on a convolutional neural network with U-net architecture and compared with PSPnet architecture. The limited possibility to use manual labour when growing, sorting, processing vegetables, fruits and berries in the face of increasing risks of new pandemics determines the study's relevance. For the research, a neural network of the U-net architecture has been chosen based on the narrow focus of the task and small repetitive patterns. The neural network of the U-net architecture has proven itself well in solving problems of image segmentation in biomedical researches. Therefore, the author decided to expand the scope of this tool to a new area of investigation. The research is carried out on the data that the researchers have collected for the experiment. The dataset for the experiment has been generated manually based on images of different varieties of raspberries and various states of raspberry fruits. This research is expected to become a part of the complex robotic system for solving the problem of manual berry fruits sorting.

Acknowledgements

I would like to thank Victor Sakharchuk, who supervised and directed my research for this thesis. Special thanks to UCU and Oleksii Molchanovskyi for having a chance to be the student at the master program of Data Science…

Contents

List of Figures

List of Tables

List of Abbreviations

AI	**A**rtificial **I**ntelligence
ANN	**A**rtificial **N**eural **N**etwork
CNN	**C**onvolutional **N**eural **N**etwork
ConvNet	**C**onvolutional **N**etwork
CP	**C**orrect **P**rediction
CV	**C**omputer **V**ision
CVAT	**C**omputer **V**ision **A**nnotation **T**ool
DNN	**D**eep **N**eural **N**etwork
FC	**F**ully **C**onnected
FNN	**F**eedforward **N**eural **N**etwork
GAN	**G**enerative **A**dversarial **N**etwork
IoU	**I**ntersection over **U**nion
mAP	**m**ean **A**verage **P**recision
MV	**M**achine **V**ision
RNN	**R**ecurrent **N**eural **N**etwork
ReLU	**R**ectified **L**inear **U**nit
SaaS	**S**oftware as a **S**ervice

Dedicated to my dear son, who patiently waited for the finish of the research to play computer games in harmony . . .

Chapter 1

Introduction

1.1 Description of the industry/domain

Today digital image analysis is widely used in many areas of human life. This interest results from the remarkable development of modern electronics and computer technology, which in some of their features approaches the technical characteristics of a human being. Advances in biomechanics have come very close to accurately mimicking the motor activity of the human brain.

Computer vision as an artificial intelligence domain is one of the most demanded areas in the modern IT world in its vein. Computer Vision, including Machine Vision, is the automatic fixation and processing of images, both stationary and moving objects, using computer tools [Gartner, 2020].

The first attempts to make a computer "see" date back to the early 60s of the 20th century. However, only in recent years, due to the increase in computing power and speed of processors, memory volumes, increased resolution and other parameters of cameras, the development of communication channels bandwidth, as well as the emergence of technologies such as Machine and Deep Learning, Artificial Intelligence, Computer Vision / Machine Vision technologies find applications in various spheres and people's daily activities.

Two factors that determine progress in computer vision are the development of theory and methods and the development of hardware provision. For a long time, theory and academic research outpaced the possibilities of CV systems‘ practical use.

Computer vision tasks include using photos and videos to analyse and understand the processes and phenomena occurring on the objects under study.

Since there is a certain natural complexity in "understanding images", a visual object often lacks any causal or dynamic description. There is no physical law or mathematical equation to describe the semantic content of such an object. Instead, the variety of bright geometric shapes reflects the information content of the image.

Therefore computer vision tasks are pretty complex and difficult to formalise. For example, a person solves the problem of classifying objects at the subconscious level. However, a wide variety of different objects‘ properties (for example, brightness, geometry) and even little variability of images lead to significant difficulties in developing computer vision algorithms.

For a long time, algorithms that operated on separate images and analysed each as a separate element distinguished the boundaries, identified the depicted object and determined its characteristic features. Now, this method is called "classic". According to [Lin and Davis, 2010, Hashemi et al., 2016], subtasks of CV problems are contour analysis, template matching, search outside templates, matching by key points (feature detection, description matching), and data fusion.

In 2013, there was a real boom in the development of neural networks, which immediately found wide use in computer vision tasks and machine learning algorithms.

The neural network approach has made a real breakthrough since neural networks made it possible to improve analytical algorithms and form expert knowledge bases to train the relevant neural networks for solving specific problems.

At present neural network technologies are among the most promising directions for developing artificial intelligence [Gartner, 2019, AIMultiple, 2021, Campos-Taberner et al., 2020]. Moreover, since they represent a mathematical apparatus that allows the reproduction of fairly complex dependencies, their use is advisable for solving complex to formalise tasks. Significantly in case the input data weakly correlates with the output.

Computer vision encompasses image segmentation, object detection, image classification, tracking moving objects over time, face recognition, optical character recognition, generating images.

In general, CV systems consist of a photo or video camera and a computer which analysis video, streaming, image files using special software. [Bhargava and Bansal, 2018, Park, JeeSook, and Kim, 2018].

CV image processing systems use techniques such as Machine Learning, Deep Learning and Neural Networks. These methods mimic the process of recognition and analysis that takes place in the human brain.

The technology discussed above appears in the agriculture and food processing industry projects, where artificial neural networks solve many applied problems. Previously, people solved such problems in an ineffective and costly manual manner. At the moment, neural networks make it possible to implement precision farming projects, monitor cultivated areas and other natural landscapes, diagnostics of diseases in plants and animals, control various kinds of human activities.

Therefore, any problem to solve in a specific knowledge area, such as in agriculture or agro-processing, has its unique features. Consequently, its solution has something unique and special. Therefore, upon closer examination, one can find problem statements for neural networks in each subject area.

Our research is cross-sectoral since it encompasses agriculture, berry growing, berry processing, and computer vision.

Researchers most often use blueberries and strawberries in their experiments to recognise the qualitative and quantitative characteristics of berries. Unfortunately, raspberry quality detection is not enough presented in the research literature, although this berry ranks second in the world market's sales structure. Moreover, workers sort it by hand in rooms with a temperature of two degrees below zero Celsius.

In Ukraine and almost all Eastern European countries, the primary producers are small private households [Plaza, 2020]. In Western Europe and North America, the share of small producers fell sharply. Large companies that can compete in the market replaced them. For example, in Great Britain and Northern Ireland, many fruit growers became part of the large cooperative K.G. Fruits / Berry Growers. The leading producer of fresh berries in North America is the DSA Driskol Raspberry Growing Partnership (Watsonville, Calif.), which markets up to 80 per cent of raspberry production [Plaza, 2020].

Berry fruit production in the Western world has changed significantly over the past 30 years. If earlier berries were grown mainly for processing, now most of the products are sold fresh. For example, in Scotland, about 80 per cent of the berry crop goes to the fresh market [Intelligence, 2020]. It became possible thanks to the creation of varieties with hard berries, special refrigeration units and storage facilities, small-capacity containers, air transportation, a streamlined logistics system, and sales of products to consumers through a supermarket chain. But growing and sorting berries is a labour-intensive process.

1.2 Motivation

The world is entering an era of pandemics now. In particular, the spread of coronavirus limits the ability to use human workers' work in berry growing and berry processing. In addition, the shelf life of many berries is limited to a few days after harvest. For instance, a raspberry fruit structurally is very fragile when compared to other fruits and berries. Therefore, it restricts sales and increases processing and storage costs for producers and retailers.

Furthermore, raspberries being a very perishable product, succumb to mildew as the fruit bruises easily. Berry fruits of low quality, as well as unripe ones, can provoke a stomach disorder. Mould can spoil a whole lot of raspberries. To be sold, frozen or canned berries should be even, free from spoiled areas and signs of disease and rot. Therefore, the producers and retailers are interested in detecting damaged fruit at an early stage and must inspect the large batches of berries every day. It is a labour-intensive and time-consuming procedure. Humans' ability to recognise defects becomes inconsistent as they get tired or become distracted. That is why the research in detecting the quality of food products using optical non-invasive techniques is relevant.

The applied area of the master's research is related to solving the problems of segmentation, classification and categorisation of raspberries based on the use of convolutional neural networks.

The research aims to develop machine learning methods for image processing in small samples with an artificially enlarged dataset using the example of segmentation, classification and categorisation of raspberry images.

1.3 Goals of the master thesis

1. To provide a literature review on artificial neural networks application in agriculture and agro-processing to raspberry fruits sorting in particular;

2. To apply CNN technique to raspberry fruits' quality detection;

3. To determine during the training experiments the most relevant neural network architecture for assessing the quality of raspberries;

4. To develop a real-time semantic segmentation application for detecting raspberry fruits quality with the RGB camera;

1.4 The master thesis structure

Chapter 2 presents the existing state-of-the-art literature on the artificial neural networks‘ application to the development of computer vision algorithms for agricultural products‘ quality detection. In chapter 3, we provide architecture and approach to the convolutional neural networks‘ building. Chapter 4 describes the dataset of the research: its preparation, prepossessing before training the model. Chapter 5 describes the experiments and their results. Chapter 6 sums the study and points out the prospects for further work in the delineated research domain.

Chapter 2

Literature review: ANN in berry growing and processing (agriculture)

2.1 Methods of berry fruits quality detection

The culture of berry consumption stimulates the demand for quality products and an increase in supply from producers. To maintain the competitiveness and high quality of products, producers need to introduce new technologies in the process of growing berries and post-harvest processing, especially sorting.

During the period from harvest to sale, berries undergo different changes. In the growth period, the berries accumulate valuable substances. The main task when storing berries is to create conditions under which the loss of nutrients would be minimal, and the quality of products would remain the same after harvest. At high temperatures, there is an accelerated metabolism, loss of moisture, vitamins, organic substances. Therefore, the fruits overripe and deteriorate faster. So, immediately after collection, it is crucial to sort berries by quality and cool as quickly as possible the products intended for short-term (from several days to 1-2 months) or long-term (from 2 to 10 months) storage. Under these conditions, the workers carry out a final inspection to sort out the damaged and diseased berries.

The following checks ensure that berries are of appropriate quality and prepared adequately for transportation:

- Quality check (ripeness, firmness, decay, mould)
- Weight and diameter check
- Temperature control
- Defects check (including amount and seriousness of defects)
- Foreign object contamination

Based on high volumes of products, quality levels, and export contracts, manufacturers know that the business is quite complex and requires suitable approaches and constant quality control.

Traditional approaches to quality control of agricultural products, including berries, are sensory analysis methods, measurement methods, and physicochemical methods. Also standard are agronomic methods based on the biological characteristics of different varieties of berries. The conventional methods for the quality assessing are far from perfect since this assessment is carried out in the laboratories. Such control is time-consuming and resource-intensive, while when harvesting berries, a quick detection of the quality on-site is crucial.

The review of literature sources [Shrestha and Mahmood, 2019, Ni et al., 2020, Li et al., 2019, Shen et al., 2018] proves that the conventional approaches to berry fruits detection are developing in the following directions (Figure 2.1)

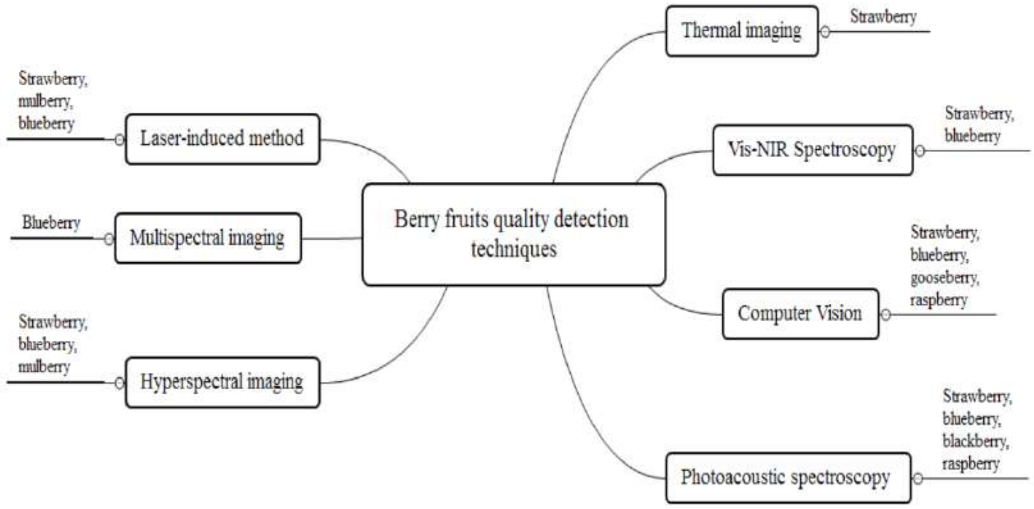

Berry fruits quality detection techniques
Thermal imaging
Strawberry
Vis-NIR Spectroscopy
Strawberry, blueberry
Computer Vision
Strawberry, blueberry, gooseberry, raspberry
Photoacoustic spectroscopy
Strawberry, blueberry, blackberry, raspberry
Laser-induced method
Strawberry, mulberry, blueberry
Multispectral imaging
Blueberry
Hyperspectral imaging
Strawberry, blueberry, mulberry

FIGURE 2.1: Berry fruits detection techniques

Vis-NIR spectroscopy provides high spectral resolution technology with a small amount of input data. At the same time, relatively cheap equipment ensures high analysis efficiency. But lack of spatial resolution is the main disadvantage of the technique [Li et al., 2019, Manjula and Sudha, 2019].

Hyperspectral imaging provides spectral and spatial resolution simultaneously. But it needs expensive equipment and a large amount of data while ensuring low analysis efficiency.

Multispectral imaging gives both spectral and spatial resolution, but the spectral resolution is lower than hyperspectral imaging.

The laser-induced method produces a quick real-time evaluation at a relatively low cost. But its spatial resolution is low.

Thermal imaging enables obtaining thermal features of material and spatial information. But external temperature strongly affects the results, and the technique needs expensive equipment.

Photoacoustic spectroscopy or imaging provides strong penetrating power since it goes deep inside material to get in-depth data. But the complex equipment structure complicates the use of the method.

X-ray techniques have extreme penetrating power, but radiation affects samples and the environment.

Odour imaging allows the differentiation among chemically diverse analyses, but gas sensors used in the technique provide poor performance through high power consumption [Li et al., 2019].

As for the computer vision, it can operate in the spatial dimension unlike spectroscopic techniques [Li et al., 2019, Pathan et al., 2020, Manjula and Sudha, 2019]. It allows the detection of natural objects with high intraclass variability, especially for heterogeneous samples.

2.2 CV application in agricultural products' quality control

At present computer vision as an artificial intelligence domain has found wide use in different spheres. Agriculture is a specific sector for its application since it generates lots of additional data, primarily visual. As it is shown in the studies [Bhargava and Bansal, 2018, Sidehabi et al., 2018, Ni et al., 2020], machine learning algorithms can be widely used in agriculture, in particular, to identify spoiled berries.

As the precision farming market promises to grow, there is a likely need to develop better agricultural data processing methods to help farmers make the best decisions.

Agricultural robotic technology finds its use in planting and harvesting. Video surveillance helps to monitor livestock and crops. The captured information goes to the analytical centre for further processing, and, based on the data obtained, a forecast and recommendations are issued to eliminate the identified potential problems. Assessment of lands and landscapes is carried out through photography, in which modern algorithms process the resulting images for identification and pattern recognition.

Applications of computer vision in agriculture include plant and leaf disease detection [Manjula and Sudha, 2019, Boulent et al., 2019, Park, JeeSook, and Kim, 2018, Park, JeeSook, and Kim, 2018], land cover classification [Campos-Taberner et al., 2020], plant recognition [Khaki et al., 2020, Akiva et al., 2020, Kamilaris and Prenafeta-Boldú, 2018, Abdullahi, Sheriff, and Mahieddine, 2017], fruit counting [Sidehabi et al., 2018, Zabawa et al., 2019, Zabawa et al., 2020] and weed identification [Tellaeche et al., 2011, Sudars et al., 2020], berries quality detection [Ni et al., 2020, Li, Li, and Tang, 2018, Hu, Zhao, and Zhai, 2018].

For years in agro-processing, the berry fruits' sorting has been done manually, which is time-consuming and produces unreliable classification [Sidehabi et al., 2018]. To cope with this problem, researchers [Kumar, 2020, Pachón-Suescúna, Pinzón-Arenasa, and Jiménez-Morenoa, 2020, Sidehabi et al., 2018, Wang, Hu, and Zhai, 2018] proposed different solutions based on computer vision applications.

2.3 CNN as a specific approach to CV problems in argo-processing

In general, the most commonly used algorithms of deep learning are feedforward neural networks, convolutional neural networks, recurrent neural networks, and generative adversarial networks [Bhargava and Bansal, 2018, Zhu et al., 2018, Long, Shelhamer, and Darrell, 2015]. Many other sub-categories of deep learning algorithms are derived from them.

These neural networks differ from each other in their structure and application area [Zhu et al., 2018]. The FNN are suitable for data fitting, pattern recognition, and classification. RNN efficiently deals with the task of time series analysis, emotion analysis, and natural language processing. GAN have found their use for image and video generation. CNN has been applied successfully for image processing, natural language processing, speech signal recognition.

The use of CNN for deep learning has grown in popularity due to some significant determinants in comparison with the rest of deep learning algorithms [Abdullahi, Sheriff, and Mahieddine, 2017, Boulent et al., 2019]:

– CNN eliminates the need for manual feature extraction since it extracts features on its own;

- CNN provides the most up-to-date recognition results;
- CNN enables retraining to perform new recognition tasks, allowing the researcher to use existing networks;
- CNN provide partial resistance to changes in scale, displacement, rotation, change of perspective and other distortions in images;

Convolutional neural networks as a particular sub-set of deep learning models and techniques have been applied successfully in different visual imagery-related tasks in agriculture: precision farming projects, monitoring of cultivated areas and other natural landscapes; diagnostics of diseases in plants and animals; plant recognition, fruit counting and weed identification, plant phenotyping, land classification, quality assessment of agricultural products [Kamilaris and Prenafeta-Boldú, 2018].

To provide a holistic view of CNN application in agriculture, especially in berry cultivation and processing, in this review, we focus on the problems solved with a CNN, a proven approach, data sources, a path to the formation and processing of a dataset, and overall accuracy of the model. Also, some authors under research compared their CNN-based approach with other methods and technologies in terms of performance (Appendix A.1)

Training artificial neural networks to solve most large-scale problems is quite time-consuming. At the same time, CNN learns to solve complex problems quickly due to weight distribution and the use of more complex models that allow massive parallelisation [Kamilaris and Prenafeta-Boldú, 2018]. Convolutional neural networks can improve the likelihood of correct classification when large datasets are available, representing the complexity of the problem under study. Structurally the CNN consists of multiple convolutional layers, pooling layers and fully connected layers. Convolutional layers extract features from input images. In the following stage, the combining layers scale them down. While fully connected layers function as classifiers and at the last level use the learned high-level functions to classify the input images into predefined classes [Long, Shelhamer, and Darrell, 2015, Kamilaris and Prenafeta-Boldú, 2018]. Therefore CNN convert input signals into features and then map the features to some target value.

2.4 NN application to image semantic segmentation

Multilevel structure and high trainability of CNN models allow them to perform classification and predictions exceptionally well in various complex practical problems.

Researchers [Akiva et al., 2020, Boulent et al., 2019, Abdullahi, Sheriff, and Mahieddine, 2017] identify stages of a neural network application to semantic image segmentation and emphasize the importance of a dataset preparation to perform image segmentation, object classification and object recognition.

In computer vision, based on deep learning algorithms, there are many available datasets for various tasks and models pre-built on such datasets. However, they have limited application to solve applied problems.

Open datasets are used to solve and evaluate general scientific problems. Available datasets are generally not suitable for highly specialized applications in various fields, especially for fruit quality recognition tasks. However, according to [Kamilaris and Prenafeta-Boldú, 2018, Bhargava and Bansal, 2018], open datasets help to broaden the size of a specific dataset, if necessary to increase the model's accuracy.

Since high-quality models on small datasets often give poor results. In highly competitive industries, public datasets in terms of completeness and quality are suitable only for solving experimental tasks. The problem with open datasets is insufficient data quality and a large number of errors. In addition, the available datasets maybe so clean that the specificity of the domain under study is lost. If the source data includes errors, they lead to distorted conclusions, even if the algorithm itself is correct. However, the purpose of the study rather than formal criteria determines the dataset quality requirements. The representativeness of the dataset is also essential.

As for the analyzed papers, some authors use datasets from publicly available resources such as the Kaggle dataset website [Kumar, 2020] and Fruit-16 K [Pachón-Suescúna, Pinzón-Arenasa, and Jiménez-Morenoa, 2020], and at the same time, researchers [Turaev, Abd Almisreb, and Saleh, 2020, Akiva et al., 2020, Zabawa et al., 2020, Wang, Hu, and Zhai, 2018, Prakash, 2018] prepared datasets on their own.

Data preprocessing is a crucial stage in building a neural network, since initial dataset may have different deteriorations. Hence, data preprocessing aims to bring data to such a state that algorithms can easily interpret their attributes means cleaning and organising the raw data set for the ANN building and training since it contains noisy data and missing data.

Image segmentation involves dividing a digital image into different subgroups of pixels to reduce image complexity and simplify image analysis [PRASAD, 2021, Sultana, Sufian, and Dutta, 2020]. There are two main types of image segmentation: semantic segmentation and instance segmentation. A unified version of the two basic segmentation processes constitutes the third type and is called panoptic segmentation. Semantic segmentation involves associating each pixel of an image with a class label. Instead, instance segmentation masks each instance of the object contained in the image autonomously.

Feature extraction aims at dimensionality reduction. In doing so, the researcher divides and shrinks the original raw dataset into more manageable groups. Such large datasets contain a large number of variables. Their processing is very resource-intensive in terms of computational power. Thus, feature extraction helps the researcher get the best features from large datasets by selecting and combining variables into elements, effectively reducing the original dataset. These enlarged features are easy to handle, but they can still accurately describe the original dataset.

Object recognition is a process that assigns a label to an object based on its descriptor [Long, Shelhamer, and Darrell, 2015, Sultana, Sufian, and Dutta, 2020].

To train convolutional neural networks the researcher divides dataset into a training dataset and a validation dataset. The training dataset is supposed to train the neural network, and a validation dataset is used to check its operation. The classification problem states: there is a set of objects; for some of them, we know to which classes they belong; for other objects, their class is undefined and requires their distribution. Alternative approaches to local feature extraction include classical computer vision methods that do not use training models. These methods search the image for distinct areas, meeting explicitly specified conditions [Shrestha and Mahmood, 2019, Bhargava and Bansal, 2018, Zhu et al., 2018].

A convolutional neural network at the input can work with different data types, for example, audio, video, images, natural language, and quantitative data, since convolution is a versatile operation, which can be applied to any signal type. As [Kamilaris and Prenafeta-Boldú, 2018] points out, this leads to the successful application of convolutional neural networks in various fields, such as the Internet (for example, personalisation systems, online chat robots), healthcare, disaster management (i.e. identifying natural disasters using images remote sensing), postal services (such as automatic address reading), the automotive industry (such as autonomous vehicles).

2.5 Tools to facilitate the CNN development

A vide variety of special libraries and tools that make it easier for programmers to work with neural networks have contributed to the explosive growth of deep learning based on neural networks and have reached the agriculture, berry cultivation and processing.

There are popular architectures for researchers to use when building their models instead of starting from scratch [Kamilaris and Prenafeta-Boldú, 2018, Zhu et al., 2018]. These include VGG, Inceptions, ResNets, MobileNets and others. Every architecture has different advantages and the most appropriate applications [Kamilaris and Prenafeta-Boldú, 2018]. All of the above architectures have already been trained with some dataset and can accurately recognise a specific problem area [Sultana, Sufian, and Dutta, 2020]. Deep learning frameworks allow researchers to save time and efforts significantly when building a CNN model.

Also, there are frameworks and platforms for experimenting with DL such as Tensorflow, PyTorch [Kamilaris and Prenafeta-Boldú, 2018].

Some implementations of popular CNN architectures build on transfer learning [Zhu et al., 2018, Sultana, Sufian, and Dutta, 2020]. Transfer learning uses preexisting knowledge of some related problems to improve the learning curve of the problem under study by tuning pre-trained models. If the researcher has a small set of training data or needs to solve a complex problem, then it is impossible to train the network from scratch. Therefore, it is useful to initialise the network with weights from another pre-trained model. Pre-trained CNNs are models that have already been trained on some relevant dataset with varying numbers of classes. These models are then adapted to suit the specific task and the dataset under study [Sudars et al., 2020, Li, Li, and Tang, 2018].

2.6 Performance metrics and overall accuracy

Performance evaluation of the recognition method is a topical issue in image recognition[Boulent et al., 2019, Akiva et al., 2020]. To obtain the numerical value of the estimate, scientists use general methods of mathematical statistics and specific indicators used to evaluate machine learning algorithms. One of the most straightforward metrics for assessing performance is the percentage of correctly recognized images [Kamilaris and Prenafeta-Boldú, 2018]. Correct recognition means obtaining at the output of an algorithm a class corresponding to a predetermined class. For the assessment, a sample is designed similarly to the training sample but contains images that the algorithm could not access during training. As a rule, the initial total sample is divided into two unequal parts (the size of the test sample may differ and make up 5-30 per cent of the full sample size).

The authors used various performance indicators in the works describing the application of convolutional neural networks in agriculture and agricultural products processing. The most popular is the percentage of correct predictions on a validation or testing dataset.

Other performance indicators included RMSE, F1 Score, QM, RFC and LC. Most of the related work used CPs, which tend to be high (i.e., above 90 per cent), indicating the successful application of CNN to a variety of agricultural problems.

The literature review has revealed that apart from image classification, convolutional neural networks can be applied for image segmentation and object detection, which are more advanced problems. Segmentation helps to identify where objects of different classes are located in the image. A group of scientists [Sultana, Sufian, and Dutta, 2020] analysed the comparative performance of various architectures of convolutional neural networks on some datasets. Table 2.1 compares the performance metric of mean average precision as Intersection over Union threshold.

Model	Year	Used Dataset	mAP as IoU
FCN-VGG16	2014	Pascal VOC 2012	62.20%
DeepLab	2014	Pascal VOC 2012	71.60%
Deconvnet	2015	Pascal VOC 2012	72.50%
U-Net	2015	ISBI cell tracking 2015	92% on PhC-U373 7.5% on DIC-HeLa dataset
DialatedNet	2016	Pascal VOC 2012	73.90%
ParseNet	2016	ShiftFlow	40.40%
		PASCAL- Context	36.64%
		Pascal VOC 2012	69.80%
SegNet	2016	CamVid road scene segmentation	60.10%
		SUN RGB-D indoor scene segmentation	31.84%
GCN	2017	PASCAL VOC 2012	82.20%
		Cityscapes	76.90%
PSPNet	2017	PASCAL VOC 2012	85.40%
		Cityscapes	80.20%
FC-DenseNet103	2017	CamVid road scene segmentation	66.90%
		Gatech	79.40%
EncNet	2018	Pascal VOC 2012	85.90%
		Pascal Context	51.70%
Gated-SCNN	2019	Cityscapes	82.80%

TABLE 2.1: Comparative accuracy of different semantic segmentation models in terms of mean average precision as Intersection over Union [Sultana, Sufian, and Dutta, 2020]

As table shows, U-net and PSPNet architectures demonstrate one of the highest accuracy levels.

The U-net is the architecture for fast and precise segmentation of images [Azimi, Eslamlou, and Pekcan, 2020, Sultana, Sufian, and Dutta, 2020, Ronneberger, Fischer, and Brox, 2015]. It provides higher accuracy for image recognition and has not been applied before for raspberry quality detection the same as PSPnet model.

Summing up, we can note that there are currently successfully implemented computer vision projects based on ANN technologies in berry growing and processing. They have shown significant performance for different purposes (fruit counting, berries quality detection, plant recognition, plant and leaf disease detection). Moreover, the considered examples in Chapter 2 are narrow in profile but can be scaled to other identical research objects by expanding a posteriori knowledge base of the developed neural network. Convolutional neural networks are just one of the variants of ANN topologies developed by scientists, primarily intended for processing images but can be used with different architectures. It significantly expands the functionality of this tool and enables researchers to implement tasks indirectly related to images (forecasting dynamics, classification, optimization).

Chapter 3

Approach to Solution

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing [Gurucharan, 2020]. CNNs were first introduced in the 1980s by Yann LeCun, a postdoctoral computer science researcher. LeCun had built on the work done by Kunihiko Fukushima, a Japanese scientist who, a few years earlier, had invented the neocognitron, a very basic image recognition neural network. But despite their ingenuity, they remained on the sidelines of computer vision and artificial intelligence because they faced a serious problem: they could not scale. CNNs needed a lot of data and compute resources to work efficiently for large images. At that time, the technique was only applicable to images with low resolutions [Dickson, 2020].

3.1 CNN Architecture

CNN application involves a convolution procedure in at least one of its layers. The neural network architecture consists of multiple parts. In turn, each of these parts consists of four elements:

- a filter bank called kernels;
- a convolution layer;
- a non-linearity activation function;
- a pooling or subsampling layer;

Each part of neural architecture represents functions as sets of arrays called function maps. Figure 3.1 gives a typical CNN architecture. It comprises several convolutional stages and fully connected layers, which gives the final output as a classification module. The main components of a typical CNN design are presented below.

Filter bank or kernels: each filter or kernel is aimed at detecting a specific characteristic at each entry point, so the spatial transformation of the entry from the characteristic detection layer will be passed to the output unchanged. According to LeCune's definition, in each convolutional layer there is a bank of $m1$ filters, and the output signal $Y_i^{(l)}$ of the l^{th} layer consists of $m_1^{(l)}$ characteristic maps of size $m_2^{(l)} \times m_3^{(l)}$. The map of the i^{th} object is calculated as follows:

$$Y_i^{(l)} = B_i^{(l)} + \sum_{j=1}^{m_1^{(l-1)}} K_{ij}^{(l)} * Y_j^{(l-1)} \tag{3.1}$$

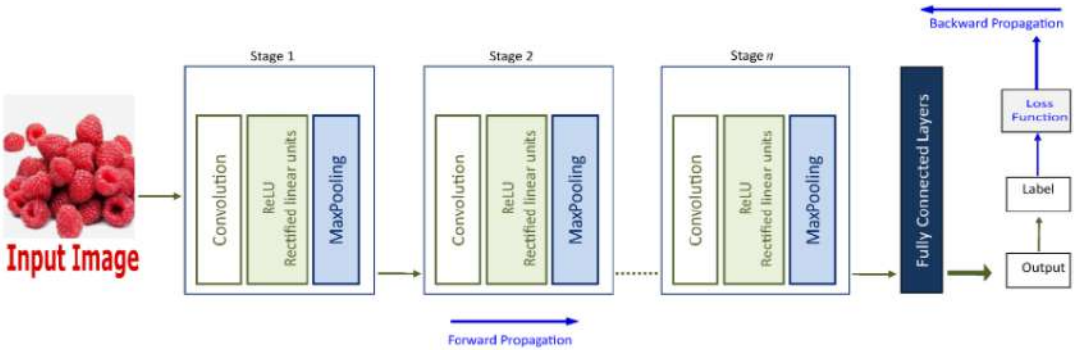


FIGURE 3.1: General architecture of a convolutional neural network [Naranjo-Torres et al., 2020].

where $B_i^{(l)}$ denotes the trainable bias parameters matrix, K_{ij}^l is the filter with dimensions $(2h_1^{(l)+1} \times 2h_2^{(l)+1})$ that connect the j^{th} feature map of $(l-1)$ layer with i^{th} feature map of (l) layer, and $(*)$ is the $2D$ discrete convolution operator.

Convolution layer: the convolution operation is widely used in digital image processing where the $2D$ matrix representing the image (I) is convolved with the smaller $2D$ kernel matrix (K), then the mathematical formulation with zero padding is given by:

$$S_{i,j} = (I * K)_{i,j} = \sum_m \sum_n I_{i,j} * K_{i-m,j-n} \tag{3.2}$$

A small sliding filter moves from left to right through the image from top to bottom during the convolution procedure. The sum of the products between each core element and the corresponding input element at each location is calculated. This is a repeatable process and can be replicated using different kernels to generate many output function maps.

The dimensions of the output feature maps are reduced more than the input ones. As an alternative, a padding method being applied will keep the same size in the plane by adding zeros around the entrance and placing the center of the kernel on the outermost elements [Dash et al., 2020]. In addition, the strides denotes the size of the passage between two consecutive core positions. Usually a a stride with value one is chosen, but sometimes a step greater than 1 is used to reduce the resolution of the feature maps that make up the downsampling.

First, the filter bank issues an output signal, and then a nonlinear activation function is applied to create activation maps, which determines the behaviour of the neuron output. Only the activated characteristics are carried to the next level. The action of the activation function is represented as:

$$\phi\left(Y_i^{(l)}\right) = f\left(B_i^{(l)} + \sum_{j=1}^{m_1^{(l-1)}} K_{ij}^{(l)} * Y_j^{(l-1)}\right) \tag{3.3}$$

The most commonly used types of activation functions for convolutional neural networks are as follows:

- Rectified Linear Unit function: it is the most used activation function for convolution layers. The ReLU activation function returns its argument x for positive values, and returns 0 for negative ones. Its derivative equal to 1 when x is positive and 0 otherwise [El Jaafari, Ellahyani, and Charfi, 2021]. It is mathematically defined as:

$$f(x) = \max(0, x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases} \tag{3.4}$$

– Sigmoid function: it's curve looks like an S-shape, as shown in Figure 3.2. The function varies between [0,1]. Therefore it is used to predict a probability as an output. Mathematically it has the form:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.5}$$

– Hyperbolic Tangent function: this function has a similar form to the Sigmoid function, but the range is [1,1]. The advantage is that it will map a zero value near zero, and negative values will be mapped strongly negative. Its mathematical definition is:

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3.6}$$

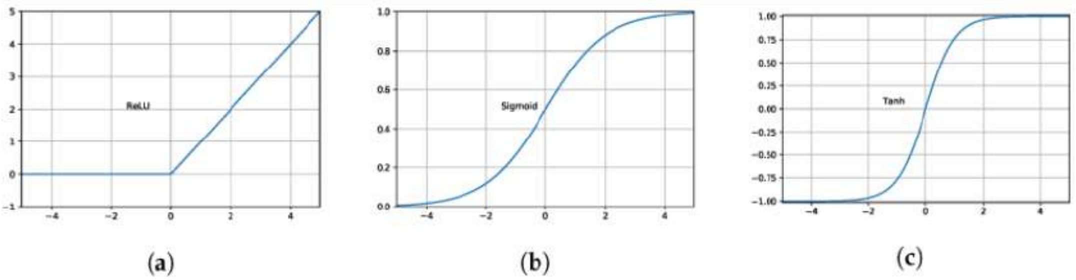


FIGURE 3.2: Curve representations of most used activation functions: (a) ReLU, (b) Sigmoid, and (c) Hyperbolic Tangent [Naranjo-Torres et al., 2020]

Pooling layer: it decreases the network's number of parameters by reducing convolutional outputs' spatial size. Additionally, pooling operations contribute to obtaining an invariant representation of the input's small translations. The pooling function can be max or average. Max polling is used more often [Dash et al., 2020].

Max pooling: it determines the maximum value for each patch of the input. The max-pooling layer saves the total value of each patch by sliding the filter over the feature map. Mathematically it has the form:

$$f_{\max}(A) = \max_{n \times m}(A_{n \times m}) \tag{3.7}$$

Usually, in a max-pooling layer, a 2×2 filters are applied with a stride of 2. It downsamples the input by 2 along its dimensions and discards the 75 per cent of the convolutional outputs.

Average pooling: it calculates the medium value for each patch of the input. The standard pooling layer downsamples the convolutional activation by dividing the input into pooling regions and computing their average values. It is mathematically defined as follows:

$$f_{\text{ave}}(A) = \frac{1}{n + m} \sum_{i=1}^{n} \sum_{k=1}^{m} (A_{i,k}) \tag{3.8}$$

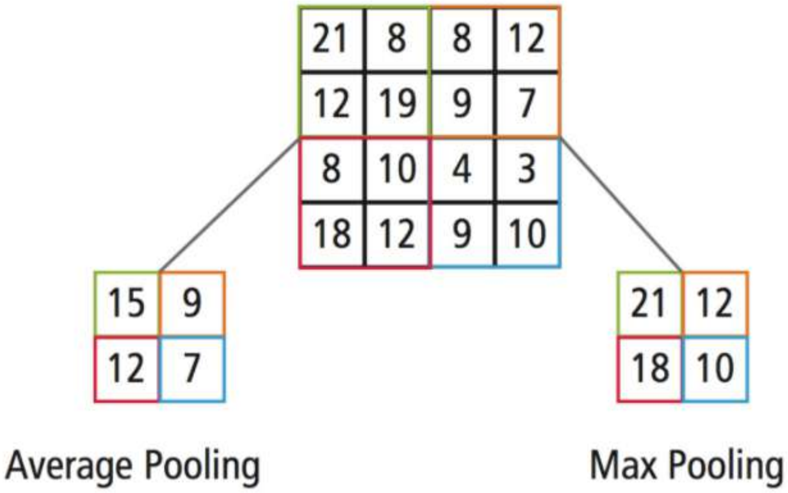


FIGURE 3.3: Examples of pooling operations by using a 2×2 filters applied with a stride of 2 [University, 2019]

Examples of max and average polling are presented in Figure 3.3

Fully connected layer: The result of convolutional cascades is reduced to a one-dimensional array and connected to a fully connected layer. The FC layers take the convolution/pooling process results and use them to classify the image by label (i.e., class), as in a traditional neural network. Thus, the activation function of the last layer (i.e. the output layer) calculates the final probabilities for each class, and it is chosen according to the problem. Typically, classifying several types uses the Softmax function, where the probability value of each class is in the range [0,1], and their total sum is 1. Finally, each output neuron selects each of the labels, and the highest output value corresponds to the classification solution.

3.2 Training Process of CNN

The CNN training aims to optimise layer parameters of a neural network to minimise variations between training dataset labels and the output predicted results. Usually, the backpropagation algorithm is most often used to train a neural network. The stages of training a neural network using the backpropagation algorithm are as follows:

1. Select a training set of images, usually obtained in batches with smaller sizes.
2. Send each batch over the network and get the result.
3. Calculate the deviation between the given labels and the predicted output applying the loss function L.
4. Propagate the error over the network using a backpropagation algorithm.
5. Update the weights K to minimise the error.
6. Redo until converge or reach iterations limit.

To complete the previous steps in CNN training, the researcher must take into account the following aspects:

- define the CNN architecture: it consists of defining the number of layers for each respective type and the size and number of filters for each layer. Architecture design always depends on the purpose of CNN;
- Loss function: measures the difference between the given ground masks and the network outputs. For example, the root mean square error function is used, which is determined by the formula:

$$L = \sum(\text{ target } - \text{ output })^2 \tag{3.9}$$

  Therefore, L must be minimised in order to find the contribution of each weight and optimise them. The gradient descent algorithm is widely applied for the minimisation procedure. Mathematically it can be expressed as the partial derivative of the loss function. Then the process of updating the parameters is expressed as follows:

$$W_k = W_{k-1} - \alpha * \frac{\partial L}{\partial W} \tag{3.10}$$

  where α indicates the learning rate. The rate of learning as a crucial parameter should be established before starting the training process. Also, a lower learning rate may provide more accurate results, but the network may take longer to train;
- Training dataset: The researcher must divide the available data into three subsets: a training set for training the network, a validation set for evaluating the model during training, and a test set for evaluating the final trained model. The lion's share of CNN structures requires all training data to have the same shape (that is, dimensions). Hence, data preprocessing is the first step before the training process to normalise the data.

Another critical point is that the dataset should be balanced, which means the same number of images for each class. Data Augmentation is a technique which can be helpful in such case. It involves increasing the amount of training data by performing a series of transformations, such as rotations, translations, and mirroring [Naranjo-Torres et al., 2020]. It is used for improving the model accuracy and is often used when there is not sufficient data for training process.

3.3 Metrics and their description

When we evaluate generally a standard ML model, we classify our predicted results into four classes: true positives, false positives, true negatives, and false negatives. Though, for the image semantic segmentation predictions, it is not contiguously clear what can be counted as a "true positive" and how we can estimate our predictions. The most widely used are the following metrics:

- The Intersection over Union metric, also known as Jaccard index, is the most popular metric in object detections, where a trained model outputs a box that fits perfectly around an object. IoU is defined as a ratio of the number of pixels common between the ground truth and prediction masks divided by the total number of pixels present across both masks. If we evaluate results for a few classes then IOU of each class is calculated and their mean is taken;

$$IoU = \frac{\text{target } \cap \text{ prediction}}{\text{target } \cup \text{ prediction}} \tag{3.11}$$

The intersection (A∩B) introduces pixels found for both the predicted mask and the original mask, whereas the union (A∪B) is simply comprised of all pixels found for both masks [Rezatofighi et al., 2019].

- Another alternative for estimating image segmentation is the Pixel Accuracy metric. It is calculated as the ratio between the correctly classified pixels, without respect of class, and the total number of pixels.

$$\text{accuracy} \ = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.12}$$

  where true positive characterize a pixel that is correctly predicted to the certain class according to the ground truth and a true negative depicts a pixel that is correctly identified as not belonging to the given class. The main drawback of such metric is that result might look good if one class overpowers the other. Say for example the background class covers 90 per cent of the input image we can get an accuracy of 90 per cent by just classifying every pixel as background [JORDAN, 2018];

- The metric which is often used in classification F1-score is used in image segmentation. The formula for the standard F1-score is the harmonic mean of the precision and recall. A perfect model has an F-score of 1.

$$F_1 = 2 \cdot \frac{\text{precision} \ \cdot \ \text{recall}}{\text{precision} \ + \ \text{recall}} \tag{3.13}$$

  Precision is the fraction of true positive examples among the examples that the model classified as positive. In other words, the number of true positives divided by the number of false positives plus true positives. Recall, also known as sensitivity, is the fraction of examples classified as positive, among the total number of positive examples. In other words, the number of true positives divided by the number of true positives plus false negatives [Wood, 2020];

- Sparse Categorical accuracy calculates how often predictions match integer labels. This metric creates two local variables, *total* and *count* that are used to compute the frequency with which *y_pred* matches *y_true*. This frequency is ultimately returned as sparse categorical accuracy: an idempotent operation that simply divides total by count.

3.4 Overview of U-net Convolutional Network

The U-Net is an architecture that Olaf Ronneberger developed for Biomedical Image Segmentation [Ronneberger, Fischer, and Brox, 2015]. It is considered as one of the best network for fast and precise segmentation of images [Sultana, Sufian, and Dutta, 2020]. Its architecture is presented in Figure 3.4

The network consists of the convolution operation, max pooling, ReLU activation, concatenation, upsampling layers operations and three sections: contraction, bottleneck, and expansion section. The contraction path gets an input, applies two 3X3 convolutions, ReLu layers and 2X2 max pooling. The number of feature maps increases in two at each pooling layer. The bottleneck layer uses two 3X3 Conv layers and a 2X2 up convolution layer. The expansion section consists of several expansion blocks, with each block passing the input to two 3X3 Conv layers and a 2X2 upsampling layer that halves the number of feature channels. In the end, the 1X1

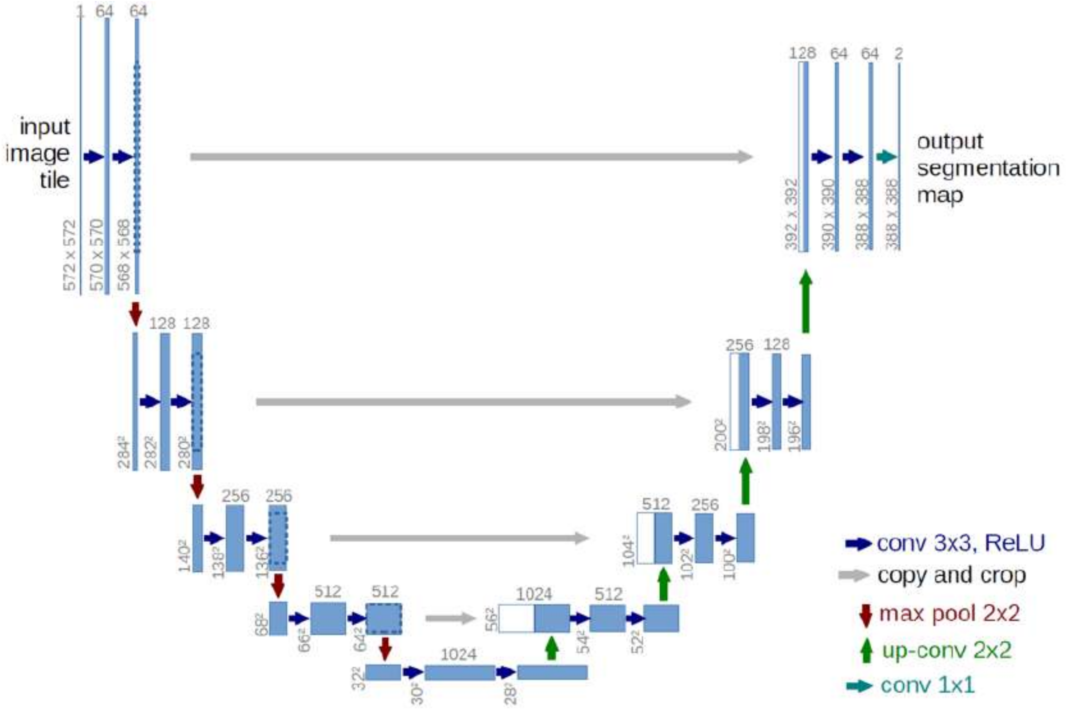


FIGURE 3.4: U-net architecture. Blue boxes represent multi-channel feature maps, while boxes represent copied feature maps. The arrows of different colours represent different operations. [Ronneberger, Fischer, and Brox, 2015]

Conv layer is used to make the number of feature maps the same as the number of desired segments in the output. U-net uses a loss function for each pixel of the image. This helps in the easy identification of individual cells within the segmentation map. Softmax is applied to each pixel followed by a loss function. This converts the segmentation problem into a classification problem where we need to classify each pixel to one of the classes. The advantage of the architecture is that it combines the location information from the downsampling path with the contextual information in the upsampling path to finally obtain general information combining localization and context, which is necessary to predict a good segmentation map. Also, it does not have dense layers that allow using of images of different sizes as input [Kızrak, 2019].

3.5 Overview of PSPnet architecture

PSPNet, or Pyramid Scene Parsing Network, is another semantic segmentation model along with the Unet, which can be trained to classify pixels in a raster. PSPNet model utilises a pyramid parsing module that exploits global context information by different region-based context aggregation. The local and global clues together make the final prediction more reliable [Zhao et al., 2017].

PSPNet introduces more context information based on the FCN algorithm. The global average pooling and feature fusion are implemented, so the feature is pyramidal, which is why the paper is called pyramid. The PSPNet algorithm is one of the most widely used semantic segmentation algorithms. It won the championship of the scene parsing task in the 2016 ImageNet competition, got the first place on PASCAL VOC 2012 semantic segmentation benchmark, and the 1st place on urban scene Cityscapes data. The mean Intersection Over Union of the algorithm on the PASCAL VOC2012 test set is 82.6 per cent [*PSPNet* 2021]. The architecture of PSPnet can be viewed in in Figure 3.5

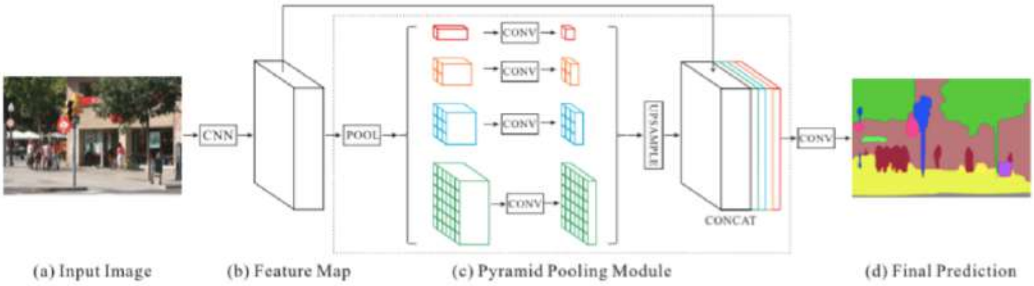


FIGURE 3.5: Overview of PSPNet. Input image (a), CNN to get the feature map of the last convolutional layer (b), pyramid parsing module is applied to harvest different sub-region representations, followed by upsampling and concatenation layers to form the final feature representation, which carries both local and global context information in (c). The representation is fed into a convolution layer to get the final per-pixel prediction (d). [Zhao et al., 2017]

Firstly, the input image passes through the feature extraction network. The most important is the next Pyramid Pooling Module which consists of total 4 branches (red, orange, blue green). In each of them, the feature map is divided into 1x1, 2x2, 3x3, 6x6 sub-regions correspondingly and the average pooling for each sub-region is performed. Pyramid pooling module allows the network to have an information for both the global scene and local objects and allows to pay attention to various scales of objects in a scene. Then 1×1 convolution is performed for each pooled feature map to reduce the context representation to 1/N of the original one if the level size of pyramid is N. In the last part of Pooling module the previous pyramid feature map is concatenated with the original feature map and performed a convolution to generate the final prediction [Zhao et al., 2017].

Chapter 4

Dataset description

4.1 Dataset collection

Raspberries are harvested mainly by hand as the fruit is very delicate and must be handled with great care.

At the same time, mechanized harvesting is widespread in some countries, particularly in the United States. For example, about 85 per cent of red raspberries farmers harvest with a harvester in Oregon and Washington[WRRC, 2021]

With a combined harvester, the fruits are removed by shaking off or using vertical vibrating drums with teeth. Vertical drum collectors pick up fruit twigs by vibrating teeth mounted on a rotating drum. In shakers, the raspberry fruits are separated from the raspberry receptacles, peduncles and sepals by shaking. The fruits are collected in the pallet of the machine and then fed to the sorting table Figure 4.1. It takes 5 to 7 people to make a preliminary quality inspection of collected berries, depending on the purpose of the product (processing or using whole fresh fruits).

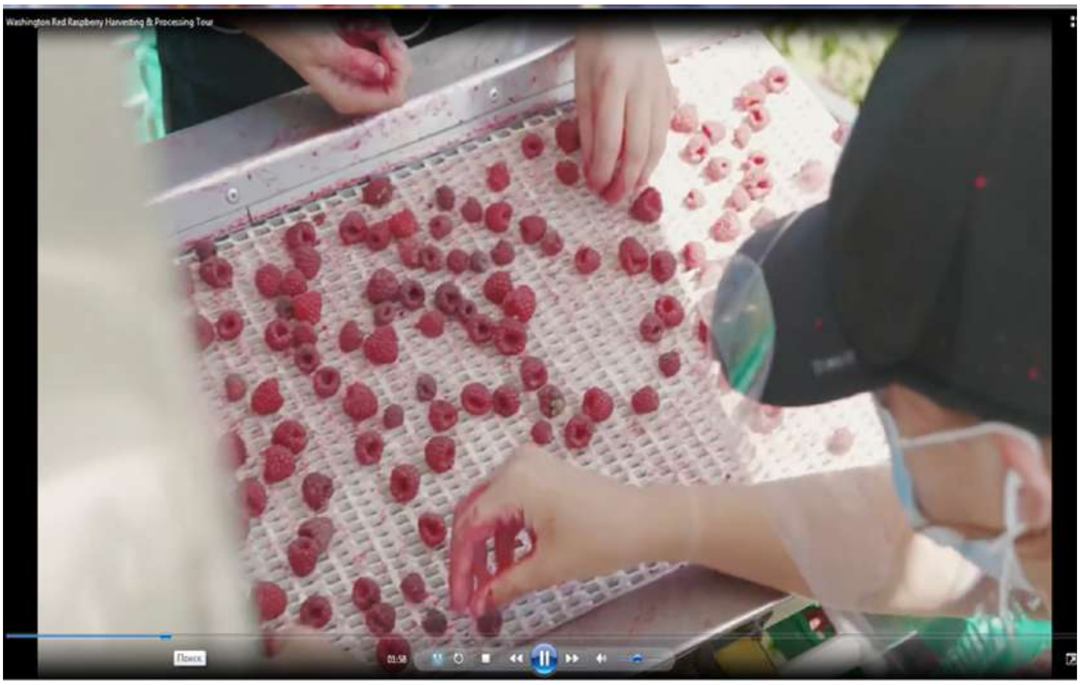


FIGURE 4.1: Manual sorting of raspberries on a harvester conveyor [WRRC, 2021]

For pre-sale berries preparation or preparation for berries processing, sorting and inspection tables or conveyors are used Figure 4.2.

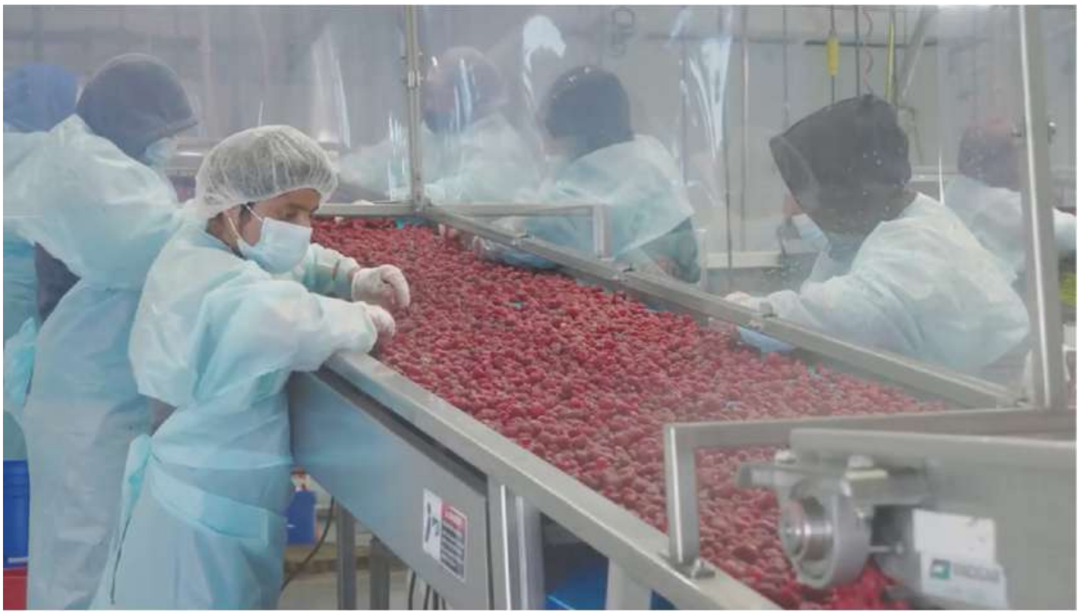


FIGURE 4.2: Manual sorting and final quality inspection of raspberries in a freezing tunnel at a temperature of -2 Celsius [WRRC, 2021]

Some inspection tables are equipped with fluorescent lamps and support wheels for easy movement of the equipment. Thus, it provides the possibility of a smooth change of the table height. The speed of the rollers, which directly affects productivity, is controlled by a frequency converter.

To replicate the industrial raspberry sorting environment as closely as possible, we used a light background and daylight to prepare the dataset.

The raspberry dataset for our experiment did not exist and was prepared and composed by ourselves. When forming the dataset, we considered the quality requirements for the raspberry fruits defined in international standards. In particular, we focused on the Codex Alimentarius [Standards, 2021], the United Nations Economic Commission for Europe (UNECE) [UNECE, 2019], United States Standards for Grades of Raspberries. We also attracted experts from the Agrana Fruit Ukraine company to visually assess the condition of the berries selected for the experiment. They helped to identify the following conditions of raspberry fruit in the photo: molded, fresh, damaged, rotten.

A significant part of the training set was created by capturing images using two smartphone cameras of different vendors. Twenty-five per cent of dataset images were downloaded from the internet resources. The camera used for image capturing has a resolution of 12MP and allows the manually adjusting the ISO value from 50 to 2000. The output is the JPG image of size 4160x3120 and 4032x3024 pixels for both cameras, respectively. Sizes of JPG images from the internet varies from 240x240 to 600x800 pixels. The colour image can be transformed into three matrices where each number is a pixel value of each RGB (Red, Green, Blue) channels. For a 600x800 RGB image we get a 600x800x3 array. Each number can be from 0 to 255.

To get the variety of the images, we collected five different sorts of raspberry from shops and markets. All raspberries were fresh and without any signs of damage or rotten on the first stage of the experiment. To get different states of spoilage, we kept the berries in different temperature and moisture conditions. The appearance and condition of the berries gradually changed under the influence of the natural conditions of the external environment. The dataset collecting process for each

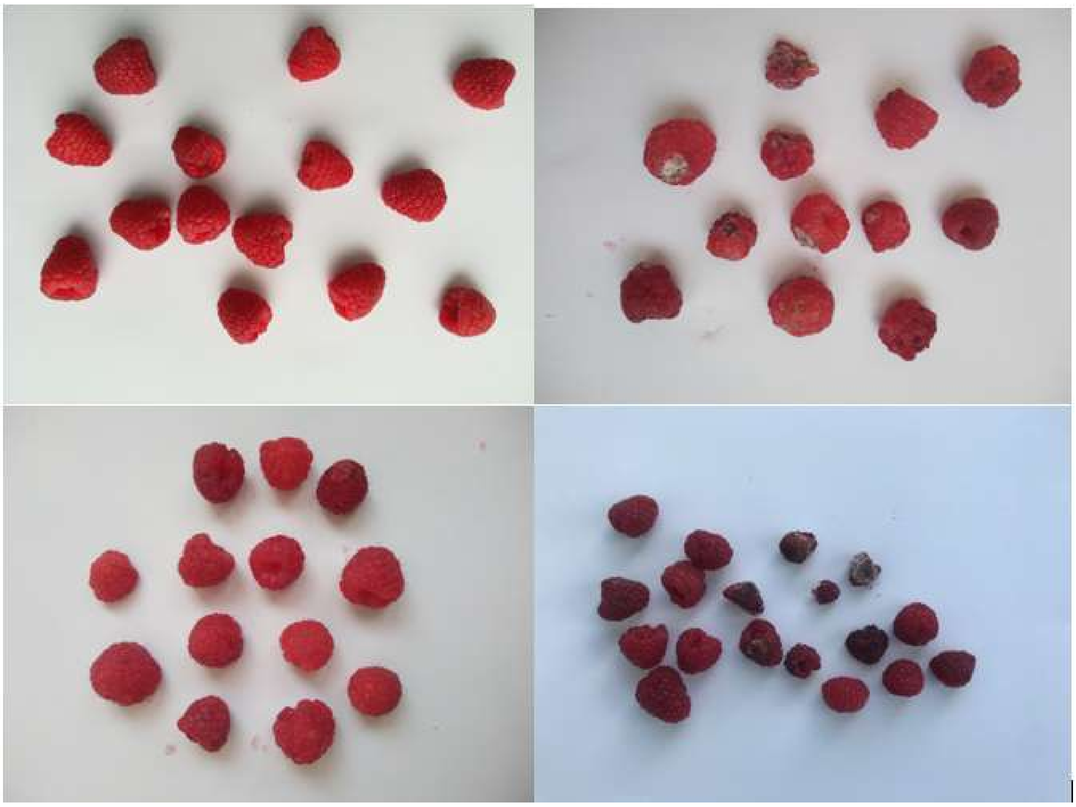


FIGURE 4.3: Images of spoiled and unspoiled raspberry fruits.

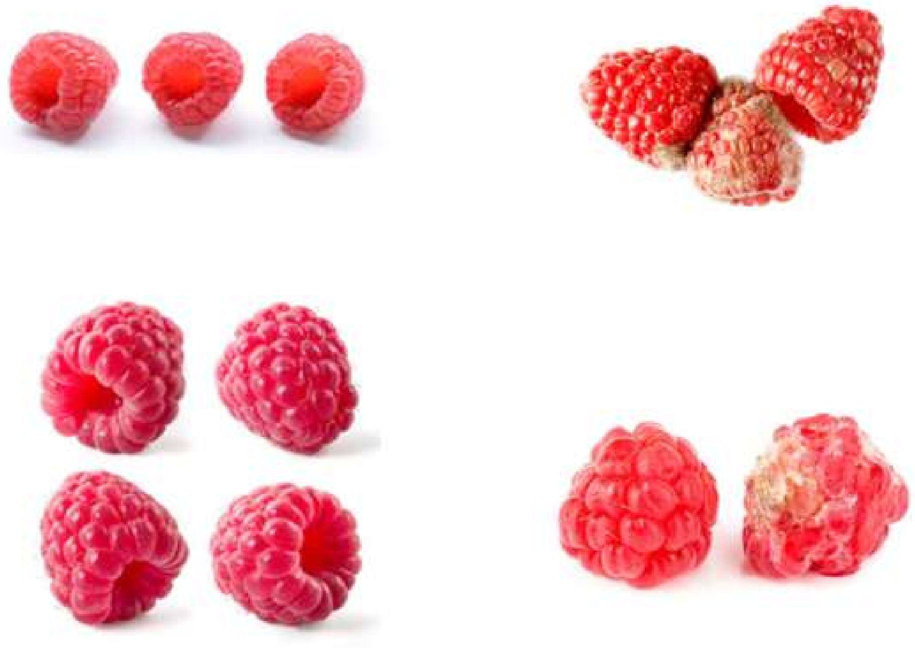


FIGURE 4.4: Sample of images from Internet.

subset took around twelve – fourteen days. Some sets were kept in the room at the average temperature, while others in the refrigerator. The signs of rotten started to appear in five-six days from the beginning of the experiment. Berries were scattered in one layer on a flat area with white background to make it similar to the sorting machine line. Once per two days, we captured images rotating the berries to view the berries from different angles. The camera was held perpendicular from the top to the berries, and the distance between the camera and surface was about the same for all images. Lighting conditions are different for some images as the sunlight was not the same every day while images were captured.

Looking for images from the Internet resources, we used searching images results from google.com. The keywords for the searches were spoiled raspberry, rotten raspberry, fresh raspberry etc. Keyword in different languages like English, Ukrainian etc., found more variety of different resources. We had to filter a significant part of images as they were with watermarks, had inappropriate format or background.

Once images from different sources were captured and a sufficient dataset has been generated, they were transferred from the camera to the local system and grouped together. As a result, we collected 400 images, including 100 images from the Internet resources. Please, find examples of captured images in Figure 4.3 and images from Internet in Figure 4.4

4.2 Image annotation

As it is mentioned [*Image Annotation 101* 2019], image annotation deals with the task of annotating an image with labels, typically involving human-powered work and, in some cases, computer-assisted help. Labels are predetermined by a machine learning engineer and are chosen to give the computer vision model information about what is shown in the image [*Image Annotation 101* 2019]. The process of labelling images also helps machine learning engineers to determine the overall precision and accuracy of their model. With image segmentation, the goal is to recognize and understand what is in the image at the pixel level. Every pixel in an image belongs to at least one class.

The most common annotation technique is the bounding box, which is the process of fitting a tight rectangle around the target object. Bounding boxes are relatively straightforward. [Petrosyan, 2019]. However, this technique has many drawbacks:

– it does not allow reaching high human detection accuracy, no matter how much data you have. Additional noise around the objects included in the segmented box causes this problem;

– the detection is very complicated for occluded objects as the target object covers less than twenty per cent of the bounding box;

– the dataset should be very large to get high accuracy.

We used Computer Vision Annotation Tool to solve the problems described above and to get an accurate pixel annotation [CVAT, 2021]. This is a free tool for image segmentation which has online or offline versions. The online program version requires uploading the set of images on the first stage. In the second stage, the researcher has to define the number of classes which will be used for annotation. We chose "Spoiled", "Unspoiled" and "Other" classes. "Unspoiled" class includes fresh

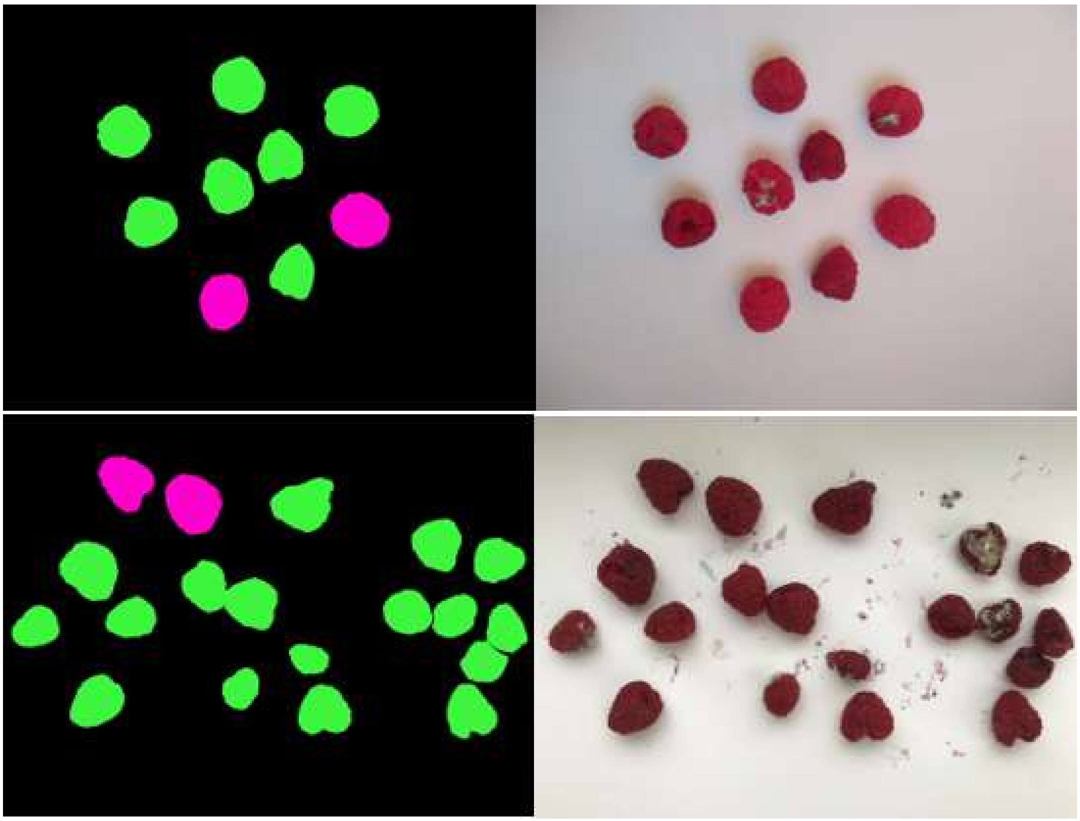


FIGURE 4.5: Images and corresponding annotation outputs from CVAT.

berry fruits without any sign of damaging, rotten or mold. "Spoiled" class includes berries with any small sign of spoilage, rotten and mold. "Other" class includes everything that is not included in "Unspoiled" and "Spoiled" classes.

Polygon annotation allows annotators to plot points on each vertex of the target object. This annotation method allows all the object's exact edges to be annotated, regardless of its shape. This is a slow and time-consuming process where the annotator has to go through object edges and carefully select each point manually, but it allows to avoid the drawbacks with bounding boxes [Petrosyan, 2019]. Downloading annotations from the CVAT allows choosing a couple of methods of data in different formats. CVAT enables the researcher to set the value of each pixel belonging to each class.

Downloading the segmented images, the researcher gets the XML file that contains some metadata and the edge coordinates for each polygon on the image, which can be used in data processing and calculations. Metadata allowed us to calculate the share of "Spoiled" and "Unspoiled" objects in our final dataset (please see Figure 4.6 for the reference). Samples of images and their corresponding annotations are available in Figure 4.5 where the green colour depicts the "Spoiled" class and the pink colour depicts "Unspoiled" class.

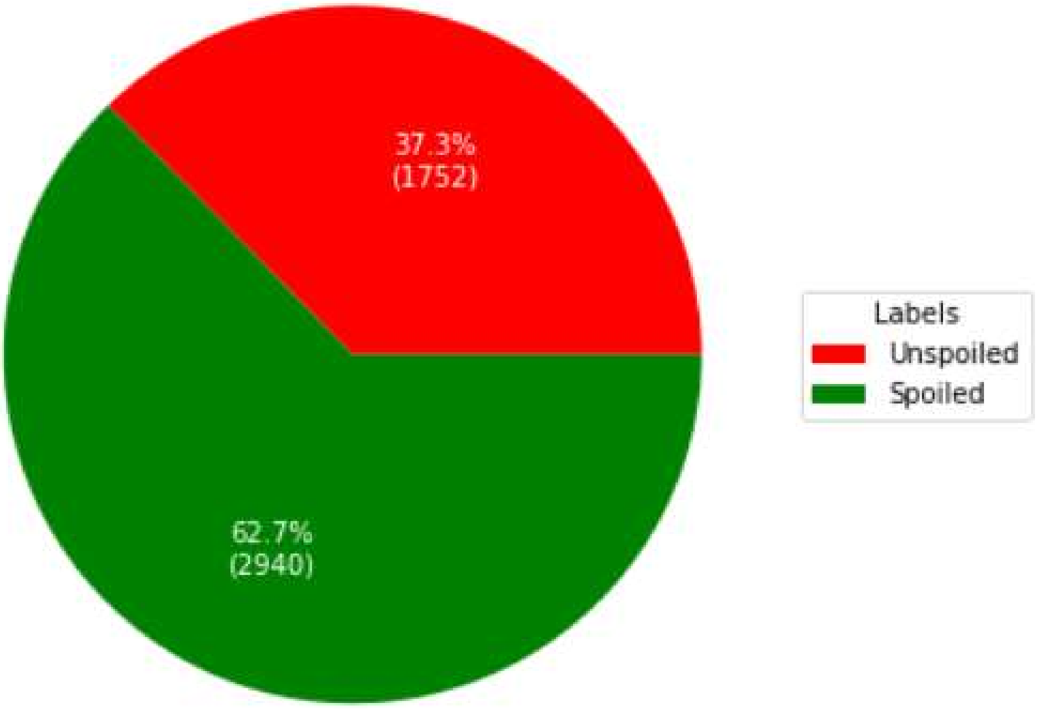


FIGURE 4.6: Share of spoiled and unspoiled berries in dataset.

4.3 Object size normalization

4.3.1 Objects distribution

The raspberry fruits‘ sizes of different varieties differ considerably in nature. Therefore the dataset displays objects of natural origin with a solid intraclass visual variability.

We do not consider recognizing raspberries under unpredictable conditions, as we prepare a sample for use in industrial situations. Where the dimensions and geometry of the sorting stand (table, conveyer) are standardized. Accordingly, we experimentally found that the distribution of raspberries in the dataset and reality will be the same, so we resize. We collect images for our data set from different sources: the Internet and our smartphone camera shooting. For the neural network not to learn signs that will not be a reality, normalization of data to the range that we will use in practice was applied.

Berry fruits‘ images differ in our data sample because of raspberry sorts, spoiled berries size decreased day after day, and the height to the camera was not the same for all images during the capturing process. To make the size of segmented objects the same for all images, we investigated them building their distribution, taking the coordinates from the annotation XML file and calculating the average size of each bounding box's sum of width and height size. Then, we resized the images to make sizes of bounding boxes in a specific range. The target size was chosen randomly for each image from 100 to 200 pixels. After resizing images, we received a new distribution of bounding boxes sizes for images objects. Further distribution is a normal one, and now the significant part of berries bounding boxes ranges from 100 to 200 pixels. Increasing the images, we increased the coordinates of each image object from the XML annotation file to get coordinates related to the resized images. Please find the distributions of object bounding boxes in Figure 4.7.

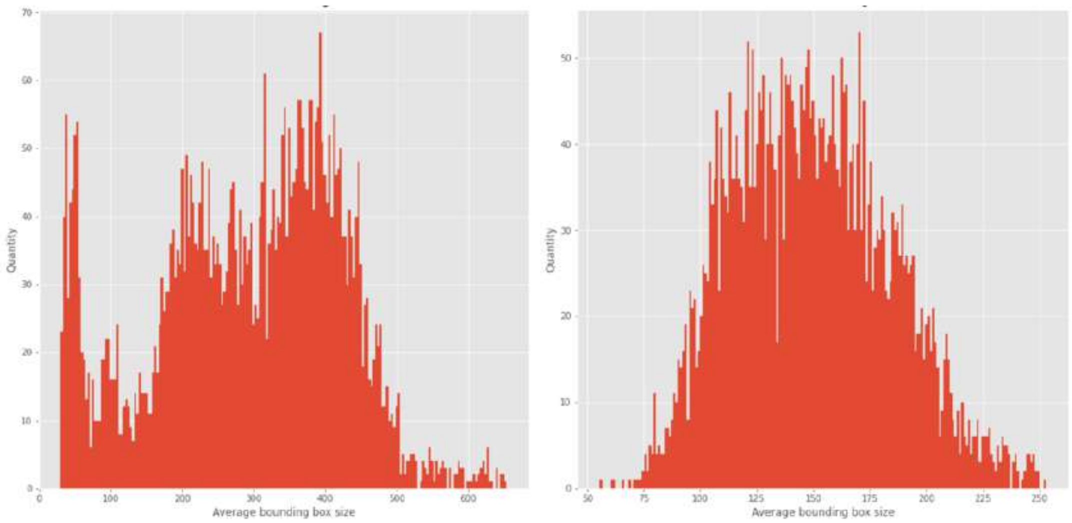


FIGURE 4.7: Distribution of Bounding box sizes before and after images resizing.

4.3.2 Bounding boxes

The bounding box is a rectangular box determined by the x and y-axis coordinates in the upper-left corner and the x and y-axis coordinates in the lower-right corner of the rectangle. After resizing, we defined the bounding boxes of the berries in the image based on the coordinate information. This process was required to verify the correctness of our calculations on image increasing and its corresponding annotation image. OpenCV 4.4.0 library and its methods were used for this purposes [Project, 2021b]. Results of plotted bounding boxes around the image objects are in Figure 4.8.

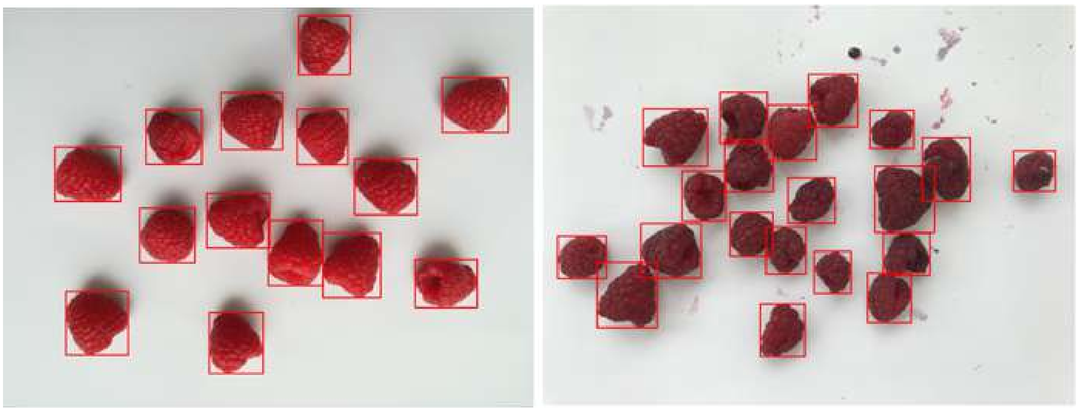


FIGURE 4.8: Bounding boxes samples.

4.4 Image segmentation

4.4.1 Images smart cropping

Once we resize images and their annotation, the next step of the experiment was to create smaller parts of large images. Image segmentation, which is quite essential for computer vision, is introduced as partitioning an image into its regions based on some criteria where the regions are meaningful and disjoint. This process is an intermediate step of some recognition objects applications [Zhou, 2015]. Image cropping in our case is required to achieve the following goals:

- to increase the number of images;
- to improve the model's learning process by highlighting the region of interest instead of resizing and losing essential features;
- to apply affine transformation to make objects positioned in different corners of the patch.

Our research combined cropping of images and centring of objects to avoid patches in our dataset which does not have many required features, for example, patches near the edges of the images where a significant part of it is a background. U-net architecture allows training model with different image sizes. However, the complexity of calculations significantly increases for the images with the dimension higher than 400x400 pixels. We defined the size of patches to be 385x385 pixels, taking into account the restriction that at least one berry on the image should not be cut in this patch by its edges. We randomly coordinated ten objects or all objects (if their value was less than ten) on the resized images, masks, and corresponding annotation coordinates. For each object, we calculated the position of its bounding boxes coordinates and shifted them randomly to the top, right, bottom, left side from the centre but with the restrictions of the patch size and image edges. Python and OpenCV library were used for programmatical calculations, cropping and saving results. The names of the images plus iterative number were used for both image's and mask's patches to correspond to each other. As a result of 400 images, we received 3100 patches of the same size and their corresponding masks. Figure 4.9 illustrates samples of patches which were used for training our model.

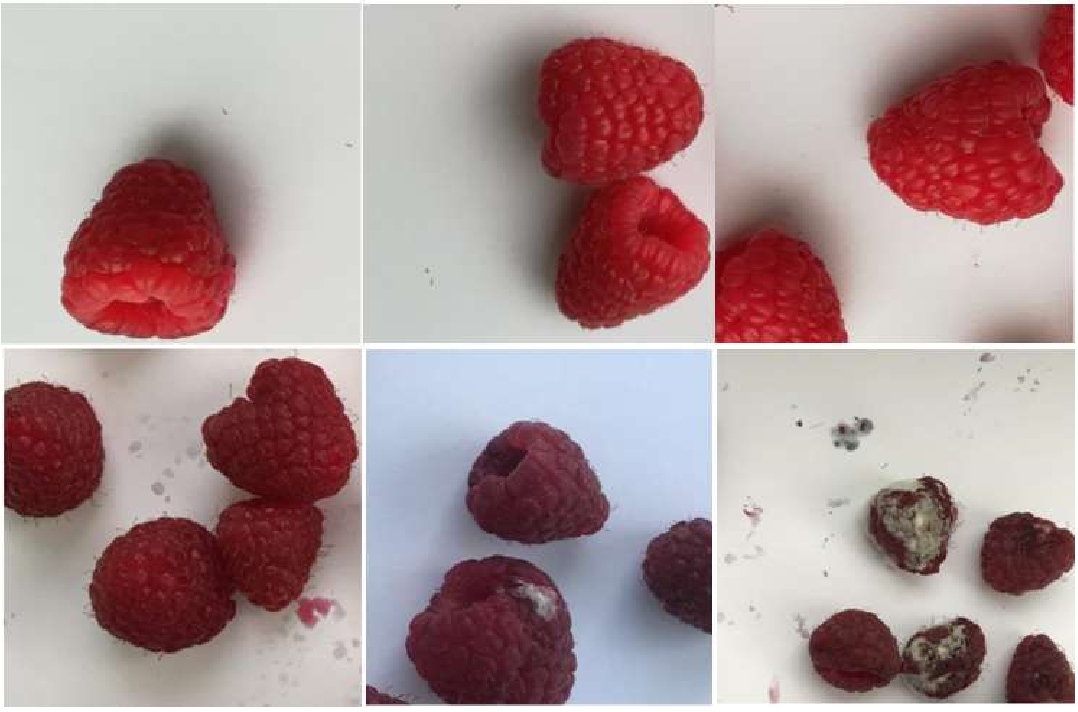


FIGURE 4.9: Samples of cropped patches

4.4.2 Images blending

After all preprocessing methods, cropping, renaming and saving results, we have to be sure that masks correspond to all images. It is a rather complicated task to make it visually, comparing file names, their view, thousands of patches and its annotations. We applied the blending method for each patch and its mask to ensure all the data is correct for model training for our task. Blending is the composition process of two or more images and outputting one image with features taken from the input ones which can be calculated by formula 4.1.

$$g(x) = (1 - \alpha) f_0(x) + \alpha f_1(x) \tag{4.1}$$

Where two source images are f0(x) and f1(x) and is from 0 to 1. this operator can be used to perform a temporal cross-dissolve between two images. The alpha value for images was taken 0.3 and 0.7 for masks corresponding [Project, 2021a].

The images and corresponding masks were taken from different folders and blended according to their file names. As a result of the experiment, we received images to check whether mask contours belong to the objects on the image or not. Results of blended images and corresponding masks are in Figure 4.10

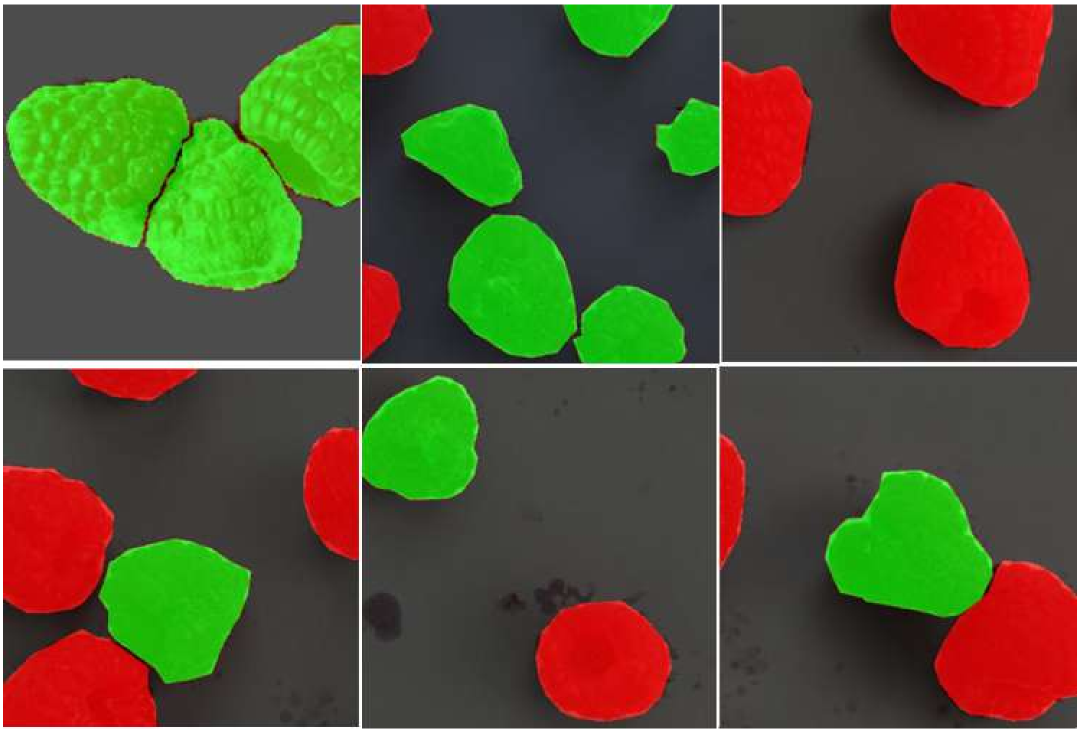


FIGURE 4.10: Samples of blended patches and corresponding annotations

4.5 Data augmentation

Image augmentation is an effective technique for increasing the size of the dataset and improving the accuracy of the model. More data is generated based on the existing dataset. Data augmentation is also relevant to use to prevent overfitting. The initial image can be modified using geometric transformations, kernel filters, colour transformation, rotation, resizing, and flipping. To increase our dataset of

3100 patches, we used the python library ’imgaug’, which provides a large variety of augmentation techniques [Jung, 2020]. To every patch of our set, we randomly applied one of the following mechanisms:

- gaussian blur with random sigma between 0 and 0.5 but only blurring 50 per cent of the image;
- strengthen or weaken the contrast in each image;
- adding Gaussian noise. For 50 per cent of images, we sample the noise once per pixel, and for the rest 50 per cent we sample the noise per pixel and channel to change the colour (not only brightness) of the pixels;
- make some images brighter and some darker.

As a result, we received additional 3100 augmented patches, and our dataset counts 6200 image patches and 6200 corresponding annotated files.

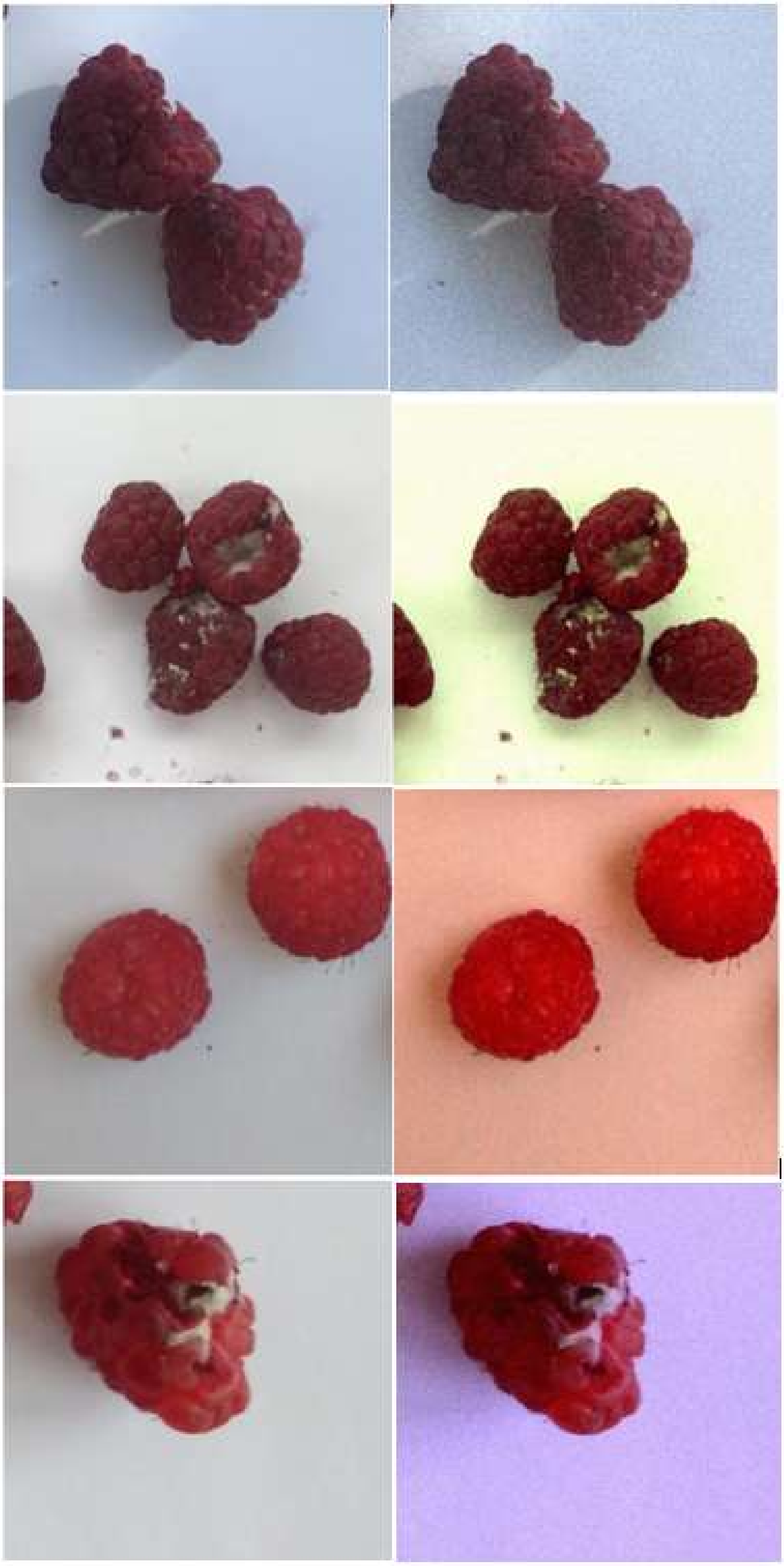


FIGURE 4.11: Original and corresponding augmented patches samples

Chapter 5

Experiments and results

5.1 Training experiments and results

We performed our experiments on Windows OS with NVIDIA GeForce RTX 2070 Super 8GB Graphics card and Cuda platform. The card allows us to decrease the training time of one epoch from 45 minutes to around 100-110 seconds. Tenserflow 2.2.1 and Keras 4.4 were used for training models. For our experiments of a building model for detection of "Unspoiled" and "Spoiled" raspberries, we chose U-net and PSP- net models as these models suit for solving our task. They have skip connections and are able to aggregate features on objects with different scaling from the largest, like raspberry fruits, to the smallest, like mold damaging.

The dataset for our training model consists of 6200 image patches, including 3200 augmented patches. Images were shuffled before training with the same seed for two models to compare the same predictions. Images were divided into training, testing, validation sets in proportion 80%, 10%, 10%. The annotation files are grayscaled images with three values (0, 1, 2) of png format where each pixel belonged to a specific class like 0 - Other, 1 - Good, 2 - Spoiled and encoded to 1-hot representation.

U-net model was compiled with Root Mean Square Propagation optimizer. The following Keras callbacks were used: early stopping if the validation loss does not improve, save the weights only if there is improvement in validation loss, write accuracy metrics to the history after each epoch.

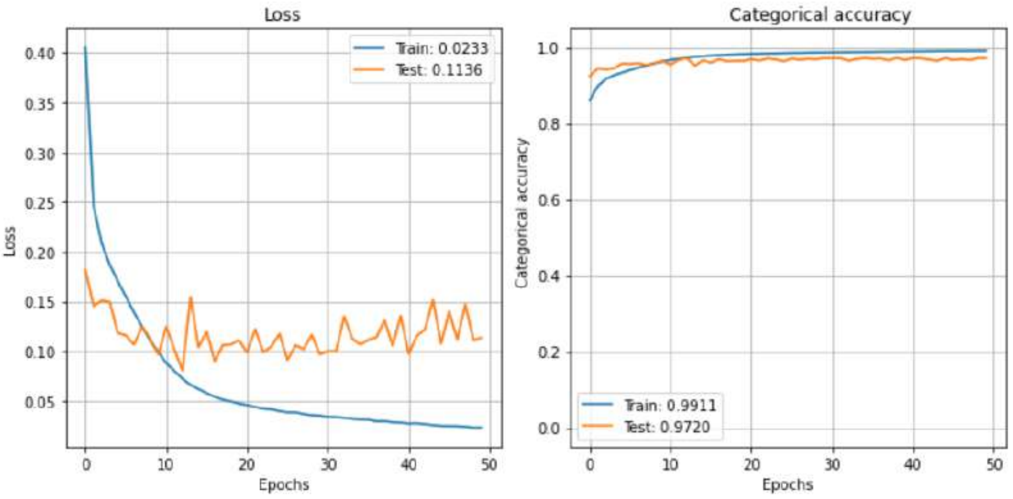


FIGURE 5.1: Training, validation loss and Categorical accuracy of U-net NN

Training the model on the first stage with 50 images without all preprocessing methods for our dataset in Chapter 4, we received an overfitted model with the accuracy on the validation set 0.66. Training the model with dataset of 6200 patches, the accuracy increased to 0.97 on the validation set. Visualization of the U-Net training history with training and validation loss and Categorical accuracy you can see in Figure 5.1.

We used a batch size of 8. The input size of patches for the model was 320x320 pixels. The predict function of the trained model outputs a (320,320,3) mask with probability inside it. Then we used argmax function with numpy to convert it to (320,320,3) true image and visualize results. It takes around one and half hours to train the model for 50 epochs.

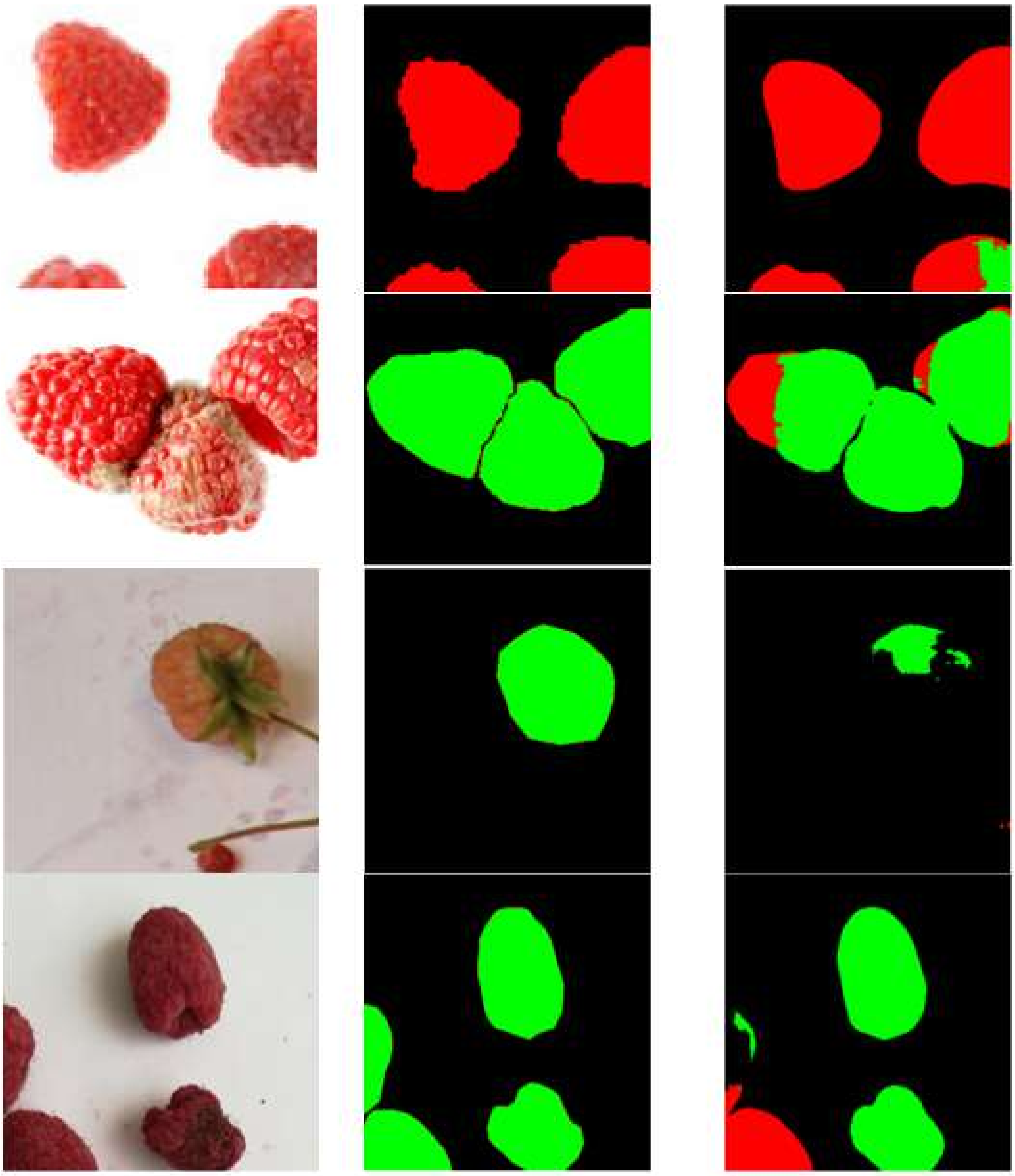


FIGURE 5.2: Wrong predictions of U-net model. a) Input input image b) Ground truth mask c) U-net prediction

Wrong predictions of U-net model are available in Figure 5.2. For the first sample bottom right corner was predicted as a "Spoiled" class. The resolution of the base image was rather small, and increasing it to the size of other samples caused image distortion and this effect. For the second sample, the model left some parts as "Unspoiled" class as these regions were too far from regions with the mold. In the third sample, we had just a few samples in the training set of unripe berries. That is why it was badly recognized. One more example of incorrect recognition is the fourth case. We annotated berries as a "Spoiled" class even if they had small damages from any side. In this case, the image was cropped and the damaged region was missed while the mask of "Spoiled" class was left.

PSPnet model was compiled with Adam optimizer with learning rate equal to 0.001 and Mean Squared Error loss function. The batch size for training was size 4.

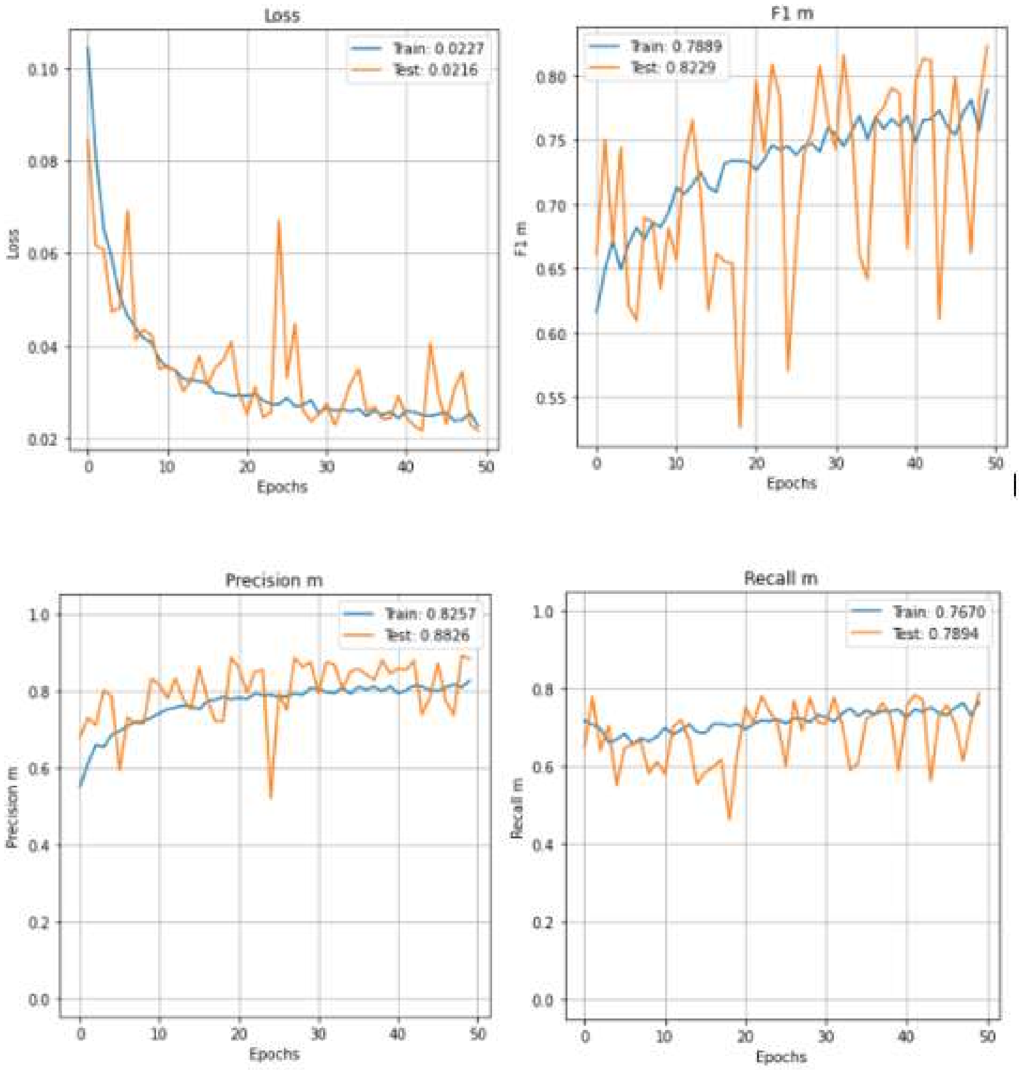


FIGURE 5.3: Precision, Recall, F1 metrics and training loss history of PSPnet NN training

Visualisation of training history and Precision, Recall, F1 metrics are in the Figure 5.3. Training time for 50 epochs was around 1 hour. After 50 epochs, the Precision values were 0.82 for the training set and 0.88 for the test set. Recall values were 0.76 and 0.78 for train and test sets. F1 score was 0.78 and 0.82 for train and test sets correspondingly.

Wrong predictions of PSPnet model are available in Figure 5.4. Some regions of "Unspoiled" class on the first sample is recognized as background. The model is very

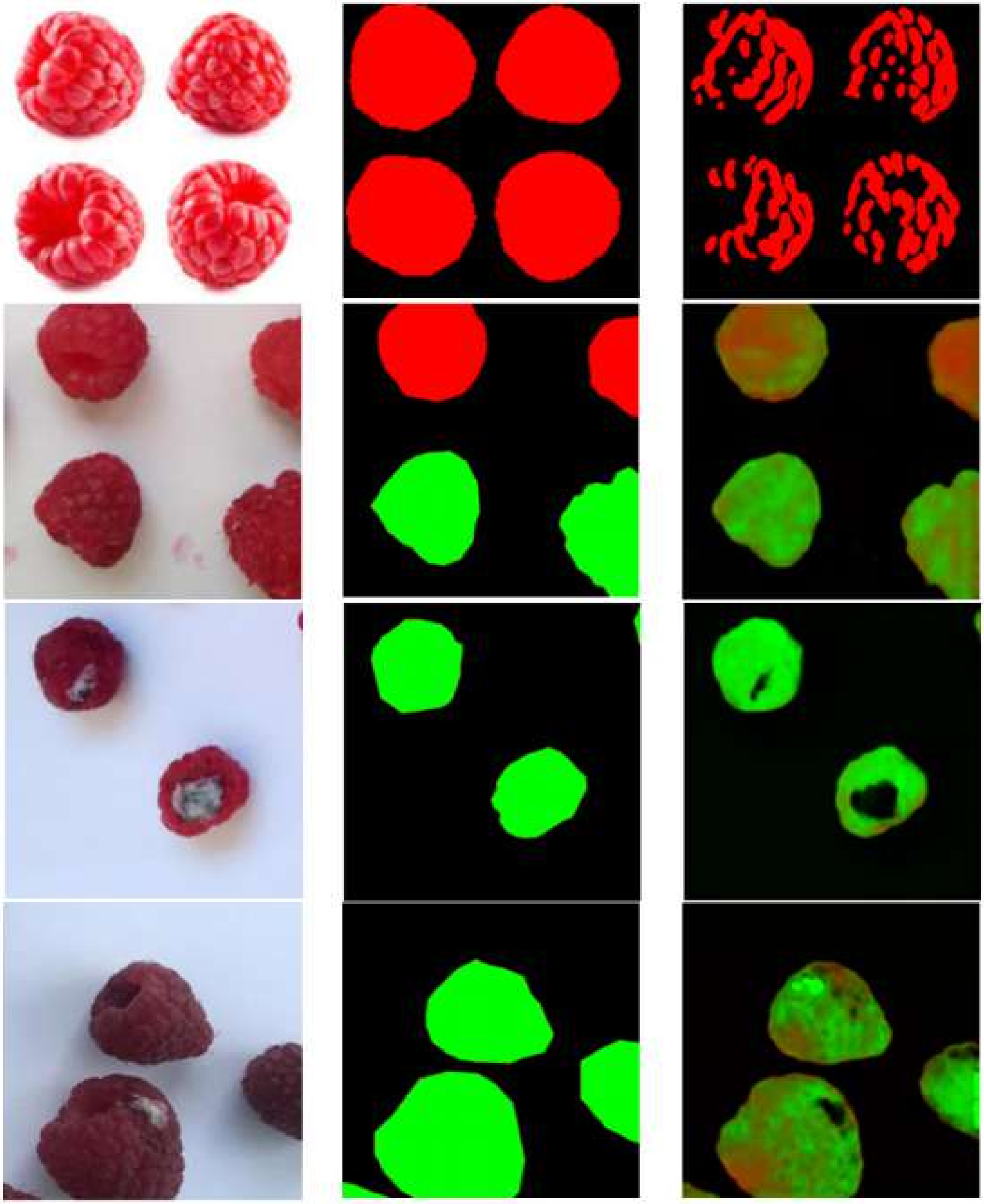


FIGURE 5.4: Wrong predictions of PSPnet model. a) Input input image b) Ground truth mask c) PSPnet prediction

sensitive to light. Shiny regions on the raspberry are recognized as a white colour similar to white background. The same thing is on the third and fourth samples. White-coloured mold on the berries is recognized as an "Other" class of background. On the second sample, we can see many green pixels on the "Unspoiled" and some red pixels on the "Spoiled" berries. The berries on the sample are sluggish and look rather similar for the model.

Visual comparison of predicted results of both models can be viewed in Figure 5.5 Both models showed a high accuracy and good prediction results for all classes, but U-net results are better. This model showed much better results with samples that are not fresh enough but still have no damaged visual areas. It distinguish better berries with white mold and shining regions while PSPnet, in these cases, predicts some wrong classification of pixels.

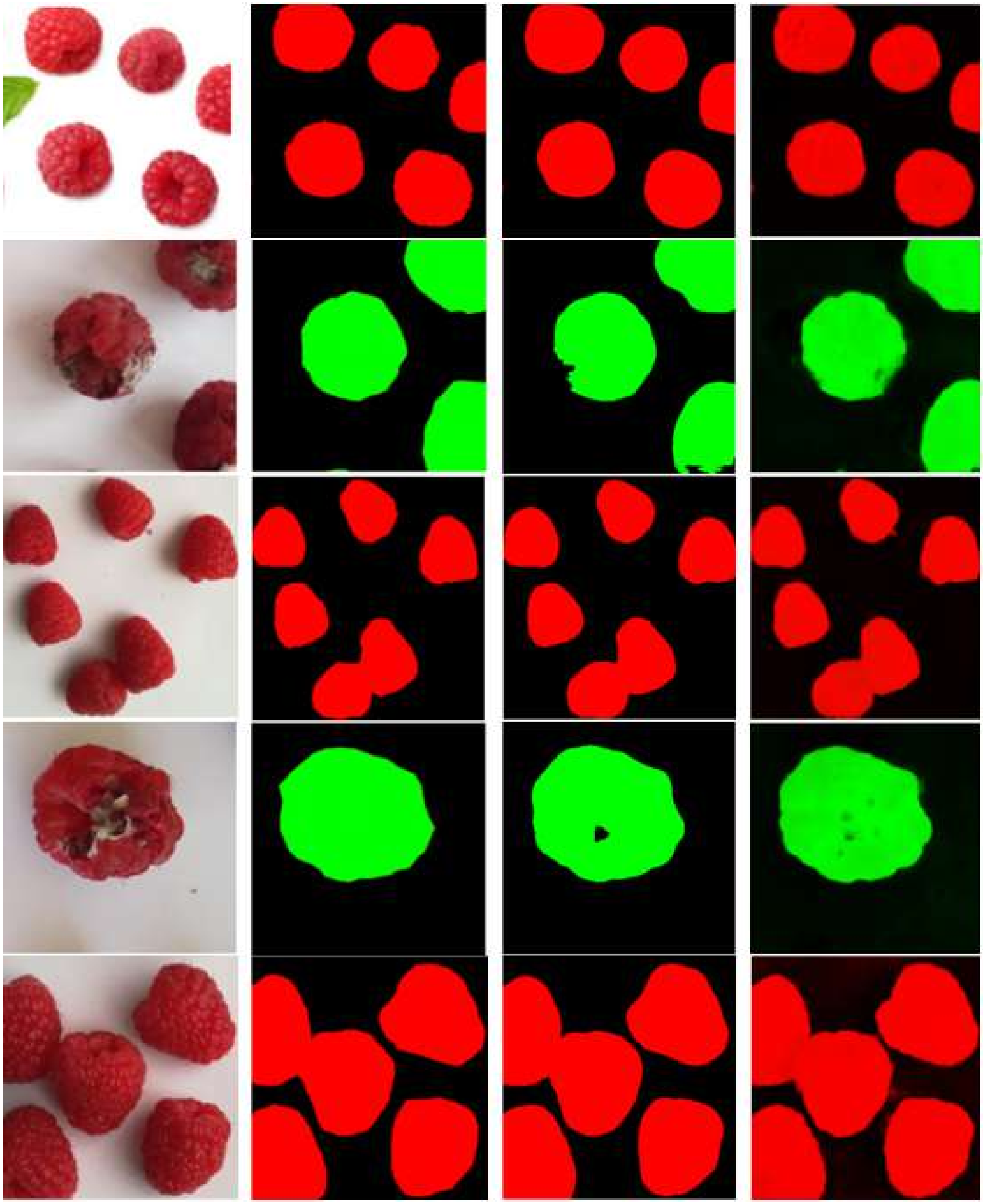


FIGURE 5.5: Comparison of U-net nad PSPnet model predictions a) Input input image b) Ground truth mask c) U-net Prediction d) PSPnet prediction.

5.2 Real-time semantic segmentation

Semantic segmentation is crucial for our research since we are going to apply the model in the complex system with the camera for live detection of berry fruits quality. We received perfect results in our Keras models and defined better results of U-net architecture. We wanted to check its performance in a more complex system like a live web application with a video camera. The benefits of running the model in the browser from the user's perspective are that there is no need to install libraries or packages. It is ready to run with GPU acceleration.

Our web application was created with the modern, widely used framework React and uploaded to the GitHub repository. The application was decided to be hosted on the AWS service. To make the process automatic and to be sure that our code works well, we used CircleCI. CircleCI is a modern continuous integration and continuous delivery (CI/CD) platform. It automates the build, test, and deployment of software. After a software repository in GitHub is added as a project to the CircleCI Enterprise application, every new commitment triggers a build and notification of success or failure through webhooks with integrations for Slack, Flowdock, or IRC notifications. We may also configure CircleCI to deploy code to various environments, including the following [CircleCI, 2021]:

- AWS CodeDeploy;
- AWS EC2 Container Service;
- AWS S3;
- Google Container Engine;
- WS CloudFront.

We chose AWS S3 and AWS CloudFront services as they suit the best for our task. Amazon Simple Storage Service (Amazon S3) is storage for the Internet. It has a simple web services interface that we can use to store and retrieve any data from anywhere on the web. Amazon CloudFront is a web service that speeds up distributing our static and dynamic web content, such as .html, .css, .js, to users. CloudFront delivers content through a worldwide network of data centres called edge locations with the best possible performance. To configure the Cloudfront to deliver your content, you specify origin servers, like an Amazon S3 bucket, from which CloudFront gets your files which then is distributed all over the world [AWS, 2021].

There are other tests/checks which can be added to the pipeline like unit test, integration test, functional tests, code style. For our application, we created the pipeline, given in Appendix B.1. Processes in CicrcleCI can be configured to run in parallel or dependently on the success pass of other tasks. In the first stage, we check if we can build the application successfully. Secondly, we check in parallel linting and prettier rules. These processes check if the stylistic rules of the project code syntaxes are correct. Developers who work with one project can set their own rules in their code editors. However, it is vital to have the same rules in the project for all contributors to see the correct difference in committing branches with the master branch. In the next stage, we checked project unit tests to ensure that the logic of required processes is not broken and the code with failed tests does not appear in the production. Deploying to the AWS S3 and Cloudfront services, publishing to the docker-hub image can be performed if all previous steps passed successfully. Otherwise, the pipeline stops running on the failed process. The last step is the notification via slack. We should not track manually how steps are passing in the pipeline. When all steps are passed, or one of the tasks failed—an appropriate message appears in the Slack channel. At the current stage of the project, it takes around three and half minutes to pass all checks and deploy the changes to the production after pushing a commitment to the git. After setting up the project and deployment pipeline, we moved to the implementation of our main task. Tenserflow.js and Keras are modern and popular technologies, and they can be combined. TenserflowJS has broad functionality, which ensures compatibility with other similar tools. In the future, it seems that we can do machine learning models across various platforms, languages and devices and optimise them to the situations where it suits most of all. Keras models are typically saved as HDF5 format file, and this format is not web-friendly. TensorFlow.js converter is an open-source library that provides a tool which is called a `tenserflowjs_converter`. We can use it to convert our Keras model in the form that TenserflowJS can utilise.

```
tensorflowjs_converter \
- -input format=keras \
/tmp/my_keras_model.h5 \
/tmp/my_tfjs_model
```

where *tmp* is a folder containing Keras model and `/tmp/my_tfjs_model` is a path to the folder where a converted model appears.

Except of *model.JSON* file, `my_tfjs_model` folder contains a set of shared weights files in a binary format. These additional files are to enable faster repeat loading by the browser. In addition, the files are below the typical cash size loading, so they are likely to be cached for sub calls when we are serving them. After converting our model, its size decreased from 24 Mb to 11.8 Mb. Tensorflow.js models have to be served through a javascript load URL. It can be loaded using the command $tf.loadModel()$, setting up the URL's parameters where the model is hosted, and TensorFlow does the rest. Since the model operates on the client-side using JSON files, it is not an exceptional case to secure the model. Setting up the camera and streaming the video of the project, we perform the following steps:

- check if data is available;
- get video properties in the project;
- set video width and height according to the required model input data;
- make a model prediction;
- draw prediction in the canvas adding specific colours to corresponding classes.

Since the raspberry season has not started yet and there is no sample, we experimented on pictures from a laptop. Then, we swiped and filmed it with a video camera on the application of another computer. The camera used in the experiment is A4Tech full HD 1080p. Computer RAM is 32 GB. Video of the experiment is available via [Blagodyr, 2021]

From our experiment, we can conclude that it works relatively fast on the machine with 32GB RAM, and there is a slight delay in less than half of a second, which is required for the image reconstruction in the browser. However, the model is pretty fast and recognises correctly "Unspoiled" and "Spoiled" raspberries. Nevertheless, running the project on the laptop with RAM 8 GB took a longer time, around one and half seconds, to display the predicted result. Another important thing is that the model in web application scope is susceptible to the surrounding light. Lighting fluctuations add more noise, and the model predicts more often "Unspoiled" berries, like "Spoiled" ones, but it still detects well the raspberry objects.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The thesis investigates theoretical and practical issues related to quality control of agricultural products using computer vision and machine learning algorithms on the example of raspberries. The focus of our study was the problem of image objects' classification using semantic segmentation. The main conclusions and suggestions are as follows:

1. Each case of CNN application has its peculiar features, which require a specific data collection. Our composing dataset has been preprocessed with the following techniques: getting image annotations, object size normalization, images smart cropping, images blending, data augmentation. The mentioned approaches allowed us to get high accuracy and visual predictions of trained U-net and PSPnet models, while both neural networks have some limitations. Output results of U-net model turned out better than PSPnet ones;

2. Combining Keras and Tenserflow.js technologies allowed us to create a fast working live video segmentation application, which provides new opportunities for semantic segmentation in the fields where it can be used. The application successfully detects all three classes of raspberry fruits. It may be used in production conditions as a detector of a spoiled raspberry just opening the browser and setting up the camera.

3. Our model demonstrates poor results on samples not sufficient in the training set (like unripe berry), images with low resolution which were significantly resized, images with shadows and pieces related to both classes, like Spoiled and Unspoiled. But, both architectures accurately detected raspberry berries with obvious signs of spoilage or fresh berry fruits without any signs of spoilage.

4. The following aspects comprehensively determine the complexity of this research. First, the data used for model training have high intraclass variability. In addition, the original dataset was small in size. And, the primary study phase was when the raspberry growing and harvesting period is over in our climate zone. Secondly, infrastructure development and deployment are the resource-intensive and time-consuming process, but it is significantly important for the production version of the application. Such projects should be implemented in a cloud environment with appropriate support. It can be built into embeded devices, but there must be an independent third-party quality control.

6.2 Future work

Prospects for further research are determined, among other things, by the limitations of our current study. In our further research, we are going to use the following improvements:

- usage of the multispectral camera for dataset preparation and detection to add additional channels, which will include additional feature field for training the model and will not influence the complexity of the process significantly;
- illumination is another essential element in the system of computer vision. To adjust and limit the influence of shadows, we want to set up stationary lighting and production conditions for the sorting process;
- data augmentation with all possible affine transformations for the outputted patches, including rotations that have not been applied in our dataset previously;
- training of the robot with the use of our model to detect and remove berries from the "Spoiled" class;
- application of the developed approach to other berry fruits.

Appendix A

Application of DL algorithms in berry cultivation and processing

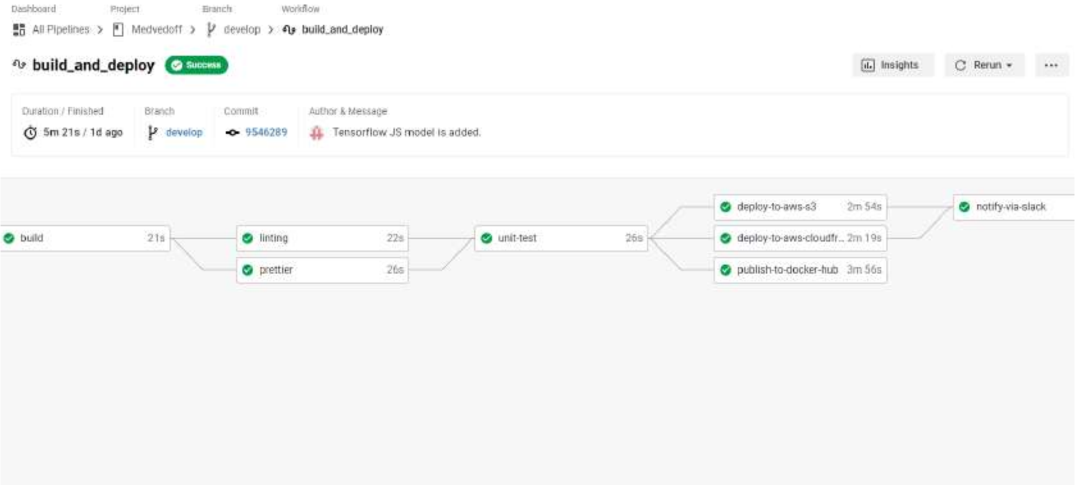
Application area	Problem description	Dataset	Precision / Metrics	DL model used
Berry fruits quality assessment	The concept of transfer learning in fruits and vegetable quality assessment, based on the pre-trained CNNs	Authors` dataset of images from 12 fruits and vegetable samples. The overall number of classes is 60.	Vgg19 model achieved the highest validation accuracy with 91.50% accuracy and the ResNet18 model scored the highest validation accuracy based on the augmented dataset with 91.37% accuracy.	Pre-trained DL models: AlexNet, GoogleNet, ResNet18, ResNet50, ResNet101, Vgg16, Vgg19, and NasNetMobile
Precision berry cultivation	Simultaneous segmentation and counting of cranberries to aid in yield estimation and sun exposure predictions	Authors` CRanberry Aerial Imagery Dataset	Mean Absolute Error for countingMean Intersection over Union for segmentation.	Author defined Triple-S Network based on U-net
Fruits quality detection	Separation of the fruits into good, moderate and rotten one	The sample dataset (11000 images) downloaded from Kaggle dataset website.	94.12% % is accomplished with help of Gradient Descent Algorithm	Author defined

Single berry fruit detection	Grapes Yield estimation and forecasting	Authors` dataset (60 independent and dot annotated images) acquired with the Phenoliner, a fieldphenotyping platform	Up to 87% of berries in the leaf-covered areas of the SMPH identified.For the VSP we are able to detect up to 94% of berries correctly.	An hourglass encoder-decoder architecturebased on the inverted residual concept
Fruit Identification and Quality Detection	Evaluation of a DAG-CNN to classify 8 different types offruit and detect if they are fresh for consumption or not.	Database FRUIT-16K 16,000 images acquired	94.43% of the accuracy ofthe test images' classification in each of the categories	NN with Directed Acyclic Graph-structure
Berry fruits quality assessment	Detection of internal mechanical damage of bluberries	Authors` dataset consisting of 557 blueberry samples, including 304 sound samples and 253 damaged samples.	Recall, Presision and F1-score	Res-Net and Res-NetXt
Berry fruits quality detection	Image Classification	Author`s dataset	Accuracy of 86% with the F1 score of 0.82	U-NET

TABLE A.1: Application of deep learning algorithms in berry cultivation and processing [Turaev, Abd Almisreb, and Saleh, 2020, Akiva et al., 2020, Kumar, 2020, Zabawa et al., 2020, Pachón-Suescúna, Pinzón-Arenasa, and Jiménez-Morenoa, 2020, Wang, Hu, and Zhai, 2018, Prakash, 2018]

Appendix B

CircleCI Pipeline



Duration / Finished: 5m 21s / 1d ago

Branch: develop

Commit: 9546289

Author & Message: Tensorflow JS model is added.

FIGURE B.1: CircleCI Pipeline

Bibliography

Abdullahi, Halimatu Sadiyah, Ray E Sheriff, and Fatima Mahieddine (2017). "Convolution neural network in precision agriculture for plant image recognition and classification". In: *2017 Seventh International Conference on Innovative Computing Technology (INTECH)*. Vol. 10. Ieee.

AIMultiple (2021). *Ultimate Guide to the State of AI Technology in 2021*. URL: https://research.aimultiple.com/ai-technology/.

Akiva, Peri et al. (2020). "Finding berries: Segmentation and counting of cranberries using point supervision and shape priors". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 50–51.

AWS (2021). *AWS Documentation*. URL: https://docs.aws.amazon.com/.

Azimi, Mohsen, Armin Dadras Eslamlou, and Gokhan Pekcan (2020). "Data-driven structural health monitoring and damage detection through deep learning: State-of-the-art review". In: *Sensors* 20.10, p. 2778.

Bhargava, Anuja and Atul Bansal (2018). "Fruits and vegetables quality evaluation using computer vision: A review". In: *Journal of King Saud University-Computer and Information Sciences*.

Blagodyr, Andrii (2021). "Live video segmentation for raspberry quality detection". In: URL: https://youtu.be/b6Ed-JzBCDo.

Boulent, Justine et al. (2019). "Convolutional neural networks for the automatic identification of plant diseases". In: *Frontiers in plant science* 10, p. 941.

Campos-Taberner, Manuel et al. (2020). "Understanding deep learning in land use classification based on Sentinel-2 time series". In: *Scientific reports* 10.1, pp. 1–12.

CircleCI (2021). *CircleCI Docs*. URL: https://circleci.com/docs/2.0/dynamic-config/.

CVAT (2021). *CVAT Project*. URL: https://github.com/openvinotoolkit/cvat.

Dash, Sujata et al. (2020). *Deep learning techniques for biomedical and health informatics*. Springer.

Dickson, Ben (2020). *What are convolutional neural networks (CNN)?* URL: https://bdtechtalks.com/2020/01/06/convolutional-neural-networks-cnn-convnets.

El Jaafari, Ilyas, Ayoub Ellahyani, and Said Charfi (2021). "Rectified non-linear unit for convolution neural network". In: *Journal of Physics: Conference Series*. Vol. 1743. 1. IOP Publishing, p. 012014.

Gartner (2019). *Artificial Intelligence Trends: Computer Vision (2019)*. URL: https://www.gartner.com/en/documents/3937025/artificial-intelligence-trends-computer-vision-2019.

— (2020). *Emerging Technologies: Tech Innovators for Computer Vision*. URL: https://www.gartner.com/en/documents/3994180/emerging-technologies-tech-innovators-for-computer-visio.

Gurucharan, MK (2020). *Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network*. URL: https://www.upgrad.com/blog/basic-cnn-architecture.

Hashemi, Nazanin Sadat et al. (2016). "Template matching advances and applications in image analysis". In: *arXiv preprint arXiv:1610.07231*.

Hu, Meng-Han, Yu Zhao, and Guang-Tao Zhai (2018). "Active learning algorithm can establish classifier of blueberry damage with very small training dataset using hyperspectral transmittance data". In: *Chemometrics and Intelligent Laboratory Systems* 172, pp. 52–57.

Image Annotation 101 (2019). URL: https://labelbox.com/image-annotation-overview.

Intelligence, Mordor (2020). *FRESH BERRIES MARKET - GROWTH, TRENDS, COVID-19 IMPACT, AND FORECASTS (2021 - 2026)*. URL: https://www.mordorintelligence.com/industry-reports/fresh-berries-market.

JORDAN, JEREMY (2018). *Evaluating image segmentation models.* URL: https://www.jeremyjordan.me/evaluating-image-segmentation-models/.

Jung, Alexander (2020). *Image Augmentation library documentation*. URL: https://imgaug.readthedocs.io/en/latest/.

Kamilaris, Andreas and Francesc X Prenafeta-Boldú (2018). "A review of the use of convolutional neural networks in agriculture". In: *The Journal of Agricultural Science* 156.3, pp. 312–322.

Khaki, Saeed et al. (2020). "Convolutional neural networks for image-based corn kernel detection and counting". In: *Sensors* 20.9, p. 2721.

Kumar S. K., Kaviya J. Prakash G. D. Srinivasan K (2020). "Fruit quality detection using machine vision techniques". In: *International Journal of Advance Research, Ideas and Innovations in Technology, 6(2)*, pp. 17–22.

Kızrak, Ayyüce (2019). *Knuth: Computers and Typesetting*. URL: https://heartbeat.fritz.ai/deep-learning-for-image-segmentation-u-net-architecture-ff17f6e4c1cf.

Li, Shuping et al. (2019). "Optical non-destructive techniques for small berry fruits: A review". In: *Artificial Intelligence in Agriculture* 2, pp. 85–98.

Li, Xin, Jun Li, and Jing Tang (2018). "A deep learning method for recognizing elevated mature strawberries". In: *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, pp. 1072–1077.

Lin, Zhe and Larry S Davis (2010). "Shape-based human detection and segmentation via hierarchical part-template matching". In: *IEEE transactions on pattern analysis and machine intelligence* 32.4, pp. 604–618.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Manjula, T and T Sudha (2019). "Cognitive Agriculture—Novel Approach for Sustainability". In: *International Conference On Computational And Bio Engineering*. Springer, pp. 183–187.

Naranjo-Torres, José et al. (2020). "A review of convolutional neural network applied to fruit image processing". In: *Applied Sciences* 10.10, p. 3443.

Ni, Xueping et al. (2020). "Deep learning image segmentation and extraction of blueberry fruit traits associated with harvestability and yield". In: *Horticulture Research* 7.1, pp. 1–14.

Pachón-Suescúna, Cesar G, Javier O Pinzón-Arenasa, and Robinson Jiménez-Morenoa (2020). "Fruit Identification and Quality Detection by Means of DAG-CNN". In: *International Journal on Advanced Science, Engineering and Information Technology, Vol. 10 (2020) No. 5*, pp. 2183–2188.

Park, Hyeon, Eun JeeSook, and Se-Han Kim (2018). "Crops disease diagnosing using image-based deep learning mechanism". In: *2018 International Conference on Computing and Network Communications (CoCoNet)*. IEEE, pp. 23–26.

Pathan, Misbah et al. (2020). “Artificial cognition for applications in smart agriculture: A comprehensive review”. In: *Artificial Intelligence in Agriculture*.

Petrosyan, Vahan (2019). *Why pixel precision is the future of the image annotation*. URL: https://superannotate.medium.com/why-pixel-precision-is-the-future-of-the-image-annotation-12a891367f7b.

Plaza, Fresh (2020). *OVERVIEW GLOBAL RASPBERRY, BLACKBERRY AND RED-CURRANT MARKET*. URL: https://www.freshplaza.com/article/9222741/overview-global-raspberry-blackberry-and-redcurrant-market/.

Prakash, Karthik Kuchangi Jothi (2018). “Spoilage Detection in Raspberry Fruit Based on Spectral Imaging Using Convolutional Neural Networks”. In:

PRASAD, SUNIT (2021). *What is Image Segmentation?* URL: https://www.analytixlabs.co.in/blog/what-is-image-segmentation/.

Project, Open Source (2021a). *Adding (blending) two images using OpenCV*. URL: https://docs.opencv.org/master/d5/dc4/tutorial_adding_images.html.

— (2021b). *OpenCV-Python Tutorials*. URL: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html.

PSPNet (2021). URL: https://www.programmersought.com/article/5332125053/.

Rezatofighi, Hamid et al. (2019). “Generalized intersection over union: A metric and a loss for bounding box regression”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 658–666.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241.

Shen, Fei et al. (2018). “On-line discrimination of storage shelf-life and prediction of post-harvest quality for strawberry fruit by visible and near infrared spectroscopy”. In: *Journal of Food Process Engineering* 41.7, e12866.

Shrestha, Ajay and Ausif Mahmood (2019). “Review of deep learning algorithms and architectures”. In: *IEEE Access* 7, pp. 53040–53065.

Sidehabi, Sitti Wetenriajeng et al. (2018). “The Development of Machine Vision System for Sorting Passion Fruit using Multi-Class Support Vector Machine.” In: *Journal of Engineering Science & Technology Review* 11.5.

Standards (2021). *International Food Standards*. URL: http://www.fao.org/fao-who-codexalimentarius/codex-texts/list-standards/en/.

Sudars, Kaspars et al. (2020). “Dataset of annotated food crops and weed images for robotic computer vision control”. In: *Data in Brief* 31, p. 105833.

Sultana, Farhana, Abu Sufian, and Paramartha Dutta (2020). “Evolution of image segmentation using deep convolutional neural network: a survey”. In: *Knowledge-Based Systems* 201, p. 106062.

Tellaeche, Alberto et al. (2011). “A computer vision approach for weeds identification through Support Vector Machines”. In: *Applied Soft Computing* 11.1, pp. 908–915.

Turaev, Sherzod, Ali Abd Almisreb, and Mohammed A Saleh (2020). “Application of Transfer Learning for Fruits and Vegetable Quality Assessment”. In: *2020 14th International Conference on Innovations in Information Technology (IIT)*. IEEE, pp. 7–12.

UNECE (2019). *UNECE STANDARD FFV-57 concerning the marketing and commercial quality control of BERRY FRUITS*. URL: https://unece.org/fileadmin/DAM/trade/agr/standard/fresh/FFV-Std/English/57_BerryFruits.pdf.

University, Stanford (2019). *Analyses of Deep Learning (STATS 385)*. URL: https://stats385.github.io/poolinglayers.

Wang, Zhaodi, Menghan Hu, and Guangtao Zhai (2018). "Application of deep learning architectures for accurate and rapid detection of internal mechanical damage of blueberry using hyperspectral transmittance data". In: *Sensors* 18.4, p. 1126.

Wood, Thomas (2020). *ML glossary*. URL: `https://deepai.org/machine-learning-glossary-and-terms/f-score`.

WRRC (2021). *The Washington Red Raspberry Commission (WRRC)*. URL: `https://redrazz.org/our-story/waredraspberries/`.

Zabawa, Laura et al. (2019). "Detection of Single Grapevine Berries in Images Using Fully Convolutional Neural Networks". In: arXiv: `1905.00458 [cs.CV]`.

Zabawa, Laura et al. (2020). "Counting of grapevine berries in images via semantic segmentation using convolutional neural networks". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 164, pp. 73–83.

Zhao, Hengshuang et al. (2017). "Pyramid scene parsing network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890.

Zhou, Kevin (2015). *Medical Image Recognition, Segmentation and Parsing*. ISBN: 9780128026762. URL: `https://www.elsevier.com/books/T/A/9780128025819`.

Zhu, Nanyang et al. (2018). "Deep learning for smart agriculture: Concepts, tools, applications, and opportunities". In: *International Journal of Agricultural and Biological Engineering* 11.4, pp. 32–44.