Ben-Gurion University of the Negev
Faculty of Engineering Sciences

Department of Industrial Engineering and Management

# Melon yield estimation using UAV images and deep learning

Aharon Kalantar

September, 2019

Supervisors:

Prof. Yael Edan, Ben-Gurion University of the Negev

Dr. Iftach Klapp, Agricultural Research Organization, The Volcani Center

**Aharon Kalantar**
*Melon yield estimation using UAV images and deep learning*
September, 2019

Supervisors: Prof. Yael Edan and Dr. Iftach Klapp

**Ben-Gurion University of the Negev**
**Faculty of Engineering Sciences**
Department of Industrial Engineering and Management
Marcus Family Campus, Ben-Gurion University of the Negev, Israel
P.O.B. 653 8410501 Be'er Sheva

Ben-Gurion University of the Negev
Faculty of Engineering Sciences

Department of Industrial Engineering and Management

# Melon yield estimation using UAV images and deep learning

Name: Aharon Kalantar

Academic Supervisors: Prof. Yael Edan
Dr. iftach klapp

September, 2019

AUTHOR: ......[illegible]...... DATE: .....24.09.19.......

SUPERVISOR: ......[illegible]...... DATE: .....24.09.19.......

SUPERVISOR: ......[illegible]...... DATE: .....24.09.19.......

CHAIRMAN OF GRADUATE STUDIES COMMITTEE: ......[illegible]...... DATE: .....25.09.19.......

# Abstract

This thesis focuses on developing an algorithm for detection and yield estimation of melons in the field using color images acquired from a digital camera mounted on an unmanned aerial vehicle. The system receives as input an aerial RGB image of a melon field, and the output is a report that includes each melon's location and estimated weight.

One of the motivations for implementing such a system is related to the fact that estimating yield production before harvesting is considered as a labor intensive task, since it requires a detailed account of accumulated yield and general yield distribution, in addition to detailed measurements of melon size and location.

The problem of object detection and yield estimation in agricultural environments using computer vision methods has been studied intensively in the past decades. Promising advancements in deep neural network algorithms have shown impressive performance in solving many problems of object detection. However, despite the wide research in this field, only a few commercialized agricultural applications use computer vision based models. This is because detecting objects in a real world scene is considered a difficult task and further complicated in the complex and unstructured agricultural environment. In addition, much of the published work focused on improving the accuracy of algorithms to accurately predict the number of fruits within images. However, in this work, we try establish a system which will not only predict the number of the fruits but will also estimate the actual weight of each fruit.

Two different systems were developed in this present work using an algorithm pipeline. Both systems include three sequential main stages: 1) melon recognition\detection, 2) feature extraction, 3) yield estimation. The implementation of the first two stages in each system was different. The yield estimation stage was identical in both systems and was based on a linear regression. The first system is composed of a classical and light CNN algorithm that requires less computational effort. The second system is based on deep learning methods and solves the drawbacks that rose from the first system. A dedicated experiment was performed in order to acquire data for both systems.

Results on 4 images including 2,264 melons acquired in 3 different seasons show high detection and promising level of yield estimation. The first system achieved average precision of 0.82 and F1 score of 0.85 for detection and more than 4% yield estimation error. The second system outperformed the first system with an overall average precision score of 0.92 and a F1-score of 0.9 for detection and only 3% yield estimation error.

The main contribution of this thesis is development of algorithms suitable for yield detection from UAV images that are characterized by small sized objects. Yield estimation includes the weight of each individual melon in additional to the traditional melon count. The research provides empirical evidence that the automation of this agriculture task can be obtained using an engineered system. With further fine-tuning of the algorithm pipeline, these systems could potentially become productive.

**Keywords:** Computer vision, Precision agriculture, Fruit detection, deep convolutional neural networks, Yield estimating/prediction, Weight estimating/prediction, Melon

# Publications

This thesis is in partially based on the following publications:

**Journal papers under review**

1. Dashuta, A, Kalantar, A., Edan Y., Dafna, A., Gur, A, Klapp, I.
   Estimating open-field melon yield by machine-vision processing of UAV images
   (Chapter 4).

2. Kalantar A., Edan Y., Gur A., Klapp I.
   A deep learning system for yield estimation of melons using UAV images
   (Chapter 5).

**Conference papers**

1. Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A., Klapp, I. 2019.
   Estimating melon yield for breeding processes by machine-vision processing of UAV images. In *Precision Agriculture'19* (pp. 1386-1393).
   Wageningen Academic Publishers. (Appendix A).

## Contents

## List of Figures

## List of Tables

# 1. Introduction

## 1.1 Problem description

Spatial information of crop growth can be provided by accurate yield estimation leading to improved resource management and marketing agronomic decisions (Lobell, Cassman, & Field, 2009). Yield estimation is especially important for providing an accurate assessment of crop yield to retailers.

Estimating the yield production before harvesting is a labor intensive almost impossible task, since it requires a detailed account of accumulated yield and general yield distribution, in addition to detailed measurements of melon size and location (Bergerman, Billingsley, Reid, & van Henten, 2016). Due to lack of human resources and the high cost of human labor (van't Ooster, Bontsema, van Henten, & Hemming, 2014) many efforts have been made to automate this process (Sarig, 1993), (Bac, van Henten, Hemming, & Edan, 2014), (Bergerman et al., 2016).

Many technologies have been used for yield estimation including satellite imagery, aerial remote sensing, geographic information systems, and ground robots/sensors (Farjon, Krikeb, Hillel, & Alchanatis, 2019).

An important tool for field monitoring and precision farming is the Unmanned Aerial Vehicle (UAV). The UAV allows to explore the crop field in a short time and provides an overview of the crop yield (Zhang & Kovacs, 2012). A prerequisite for observing and analyzing the yield is the ability to identify crops from image data (Cheng & Han, 2016), (Milioto, Lottes, & Stachniss, 2017). The fruit location in the field, its size, shape and maturity can be identified using machine vision techniques (Kapach, Barnea, Mairon, Edan, & Ben-Shahar, 2012), (Cheng & Han, 2016), (Koirala, Walsh, Wang, & McCarthy, 2019a), (Lobell et al., 2009).

The complexity of applying computer vision in the natural field environment is due to adverse weather conditions, luminance variability and the presence of obstructing leaves and branches dust, insects and other unavoidable image noises (Pereira, Morais, & Reis, 2017). Advanced image analysis achieved impressive results overcoming some of the challenges (Koirala, Walsh, Wang, & McCarthy, 2019b), (Guo et al., 2016), (Pereira et al., 2017), (Liakos, Busato, Moshou, Pearson, & Bochtis, 2018), (Gongal, Amatya, Karkee, Zhang, & Lewis, 2015).

Recently, new image recognition methods based on machine learning algorithms, such as the Convolution Neural Network, are creating new opportunities to understand complex processes in agricultural operating environments (Liakos et al., 2018). These supervised machine learning methods yield better results than traditional image processing techniques, which were based on hand-engineered features to encode visual attributes (Gongal et al., 2015). Applications such as crop management (Kamilaris & Prenafeta-Boldú, 2018), livestock management (Qiao, Truman, & Sukkarieh, 2019), water management (Mehdizadeh, Behmanesh, & Khalili, 2017), (Feng, Peng, Cui, Gong, & Zhang, 2017) and soil management (Morellos et al., 2016), (Nahvi, Habibi, Mohammadi, Shamshirband, & Al Razgan, 2016), have been automated using computer vision (Liakos et al., 2018).

Although it requires a strong computational resource and a large resource of labelled data for training recently deep learning methods have gained more practical adoption, followed the development of graphical processing units (GPU) which allow to overcome the computational resource constraint (Guo et al., 2016).

## 1.2 Objectives

The main research objective is to develop a robust algorithm for detection and yield estimation of melons in agricultural environment using color images acquired from a digital camera mounted on an unmanned aerial vehicle. The specific objectives are to:

1. Detect melons in the open field and provide their exact location in the image.

2. Find the contour of each melon in the image.

3. Fit an ellipse on top of the contour and perform feature extraction of the ellipse axis.

3. Using statistical and machine learning models for yield estimation based on the extracted geometrical features.

## 1.3 Thesis structure

This thesis begins with a literature review presented in chapter 2. The review starts with computer vision research in general (2.1), followed by a review of methods which are used in computer vision based on deep learning, such as artificial neural networks (2.2) and convolution neural networks (2.3). Next, a review of different approaches for object detection task based on deep learning is provided (2.4), followed by the recent advancements of object detection in agriculture using deep learning (2.5). The methodology of the two main research tasks conducted as part of this thesis is depicted in chapter 3. Estimating open-field melon yield by machine-vision processing of UAV images is described in chapter 4. A deep learning system for yield estimation of melons using UAV images is described in chapter 5. Conclusions and future research are discussed in chapter 6.

# 2. Literature Review

The scientific background related to the different parts of the research are reviewed in this section including computer vision (2.1), artificial neural networks (2.2) convolution neural networks (2.3), object detection (2.4) and finally a review of image detection in agriculture using deep learning (2.5).

## 2.1 Computer Vision

### 2.1.1 Computer vision background

Computer vision is an interdisciplinary scientific field that deals with how computers can be made to gain insight from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do (Sonka, Hlavac, & Boyle, 2014) (Huang, 1996) and beyond. Computer vision has been applied to a wide variety of applications, such as in medical imaging (Faust, Hagiwara, Hong, Lih, & Acharya, 2018), (Li, Zhang, Müller, & Zhang, 2018), (Anwar et al., 2018) autonomous cars (Nguyen, Jenssen, & Roverso, 2018), (Voulodimos, Doulamis, Doulamis, & Protopapadakis, 2018) face detection (Kumar, Kaur, & Kumar, 2019) and many more (Szeliski, 2010).

Computer vision began in the late 1960s, researchers from universities were pioneering artificial intelligence by trying to mimic the human visual system, as a stepping stone to endowing robots with intelligent behavior (Huang, 1996), (Szeliski, 2010). As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. These systems rely on artificial intelligence, machine learning, robotics, signal processing and geometry theory (Floyd & Sabins, 1987).
The process of computer vision includes three main stages (Floyd & Sabins, 1987), (Klette, 2014), (Morris, 2004):
a) Image acquisition – an action of capturing an object using a sensor and storing the information into a matrix as a raw data. Each cell in the matrix is called a pixel, the pixel values typically correspond to light intensity in one or several spectral bands captured (Floyd & Sabins, 1987).
b) image processing and analyzing –image processing is a procedure that converts the raw data into an image, while analyzing refers to several techniques to understand the image data (Floyd & Sabins, 1987). These techniques deal with feature extraction, extraction of regions that differ in properties such as intensity, color, texture, or any other image statistics (Morris, 2004). The combination of several features together assists to understand and interpret the image (Szeliski, 2010). Through these features the machine vision algorithm defines an object in the image (Groover, 2007).
c) Image interpretation – converting the image into information that is meaningful and valuable for a wide range of users. One popular task of interpretation is recognizing the type of the objects in the image by comparing the extracted feature from the previous stage to predefined models or standard values.

## 2.1.2 Computer vision in agriculture

Agriculture plays a critical role in the global economy. With the continuing expansion of the human population, together with the increase of global competition, pressure on the agricultural system will always increase (Liakos et al., 2018). In addition, the advance in intelligent computer technologies along with hardware cost reduction creates an opportunity to automate many tasks, Applications based on computer vision take a significant role in those tasks (Tillett, 1991). Applications (figure 1) such as, (a) crop management (Kamilaris & Prenafeta-Boldú, 2018), including applications on yield prediction, disease detection, weed detection crop quality and species recognition (Kung, Kuo, Chen, & Tsai, 2016), (Ali, Cawkwell, Dwyer, & Green, 2016), (Pantazi, Moshou, Alexandridis, Whetton, & Mouazen, 2016), (b) livestock management, including applications on animal welfare  and livestock production (Qiao et al., 2019), (Hansen et al., 2018) , (c) water management (Mehdizadeh et al., 2017), (Feng et al., 2017)  and (d) soil management  (Morellos et al., 2016), (Nahvi et al., 2016), were automated using computer vision (Liakos et al., 2018).

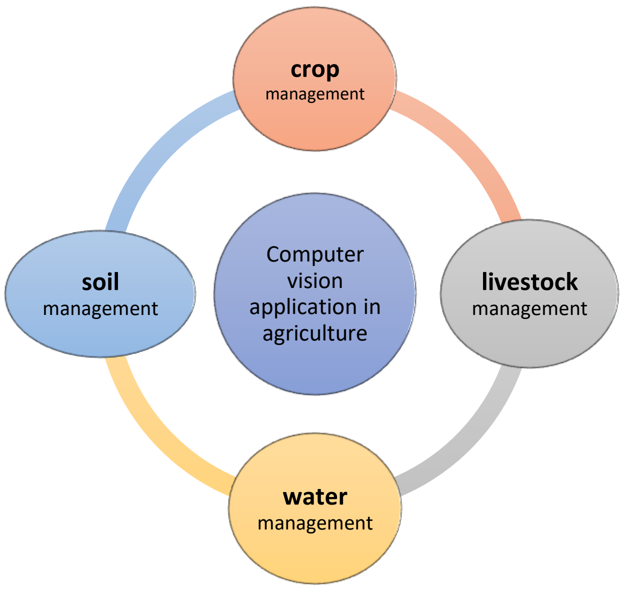


*Figure 1 - Computer vision applications in agriculture (Liakos et al., 2018)*

Despite many years of research of computer vision in agricultural environments, there are still many problems that hinder implementation of agricultural applications (Gongal et al., 2015), (Lobell et al., 2009), (Pereira et al., 2017). The highly variable and unstructured outdoor environment with changing illumination conditions and obstructions, along with the complex plant structure and variable product shape, size, color, texture and location make it hard to find a global solution to the detection of objects in the complex agricultural environment (Gongal et al., 2015), (Kapach et al., 2012), (Rawat & Wang, 2017), (Liakos et al., 2018), (Koirala et al., 2019a).

In recent years, new approaches of computer vision have emerged, based on machine learning algorithms, such as neural networks (NN). These algorithms together with big data technologies and high-performance computing, create new opportunities to unravel, quantify, and understand data intensive processes in agricultural operational environments (Liakos et al., 2018). One of the most powerful implementations of the NN is the convolution neural network (CNN).

## 2.2 Artificial Neural Networks

### 2.2.1 Background

Artificial neural networks (ANN) are computing systems that were inspired by the human brain (Hemming, 2003). The elementary units in an ANN are the artificial neurons, were presented first by Rosenblatt et al. on purpose to model the work of biology neurons in the human brain (Lee, 1988; Rosenblatt, 1958). The human brain contains an enormous amount of nerve cells called neurons. Each of these cells are connected to many other similar cells, creating a very complex network of signal transmission. Each cell collects inputs from all other neural cells it is connected to, and if it reaches a certain threshold, it signals to all the cells it is connected to (Hemming, 2003). Similar to the biology neurons, the artificial neuron receives one or more inputs and sums them to produce an output. Illustration of artificial and live neural cells is presented in figure 2.

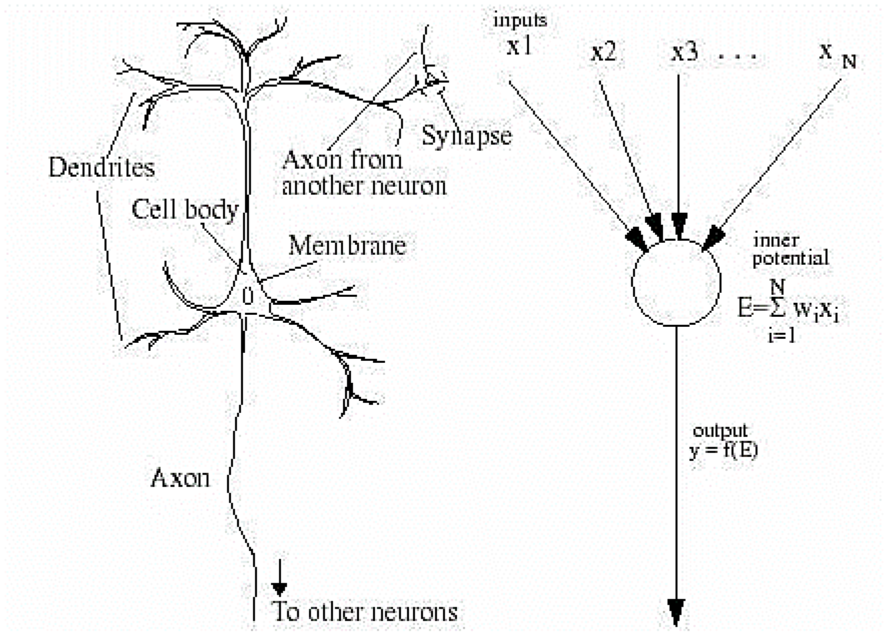


*Figure 2 – Artificial Neuron (right) inspired by Biological Neuron (left)* (Hemming, 2003)

### 2.2.2 Neural networks characteristics

Artificial neural networks include three fundamental characteristics (Hemming, 2003) : a) the network architecture. b) activation functions c) the weight of input connections.
The network architecture together with the activation functions are chosen at the initial stage and remain the same during the training process. The performance of the neural network depends on the value of the weights. The weights are tuned during the training process to obtain a specific output.

## Architectures of Neural networks

Neural networks include at least three different layers (figure 3): the input layer which contains the input feature vector Xq = ($X_1$, $X_2$, $X_3$), the output layer that consists of the neural network response, and a hidden layer, the layer in between that contains the neurons that connect to both the input and output layers (Jain, Mao, & Mohiuddin, 1996). An example for neural networks is presented in figure 3 where the input layer receives a vector, two fully connected hidden layers, each layer has 3 neurons multiplied by vector W, and a final output layer Y- vector with 3 output elements.

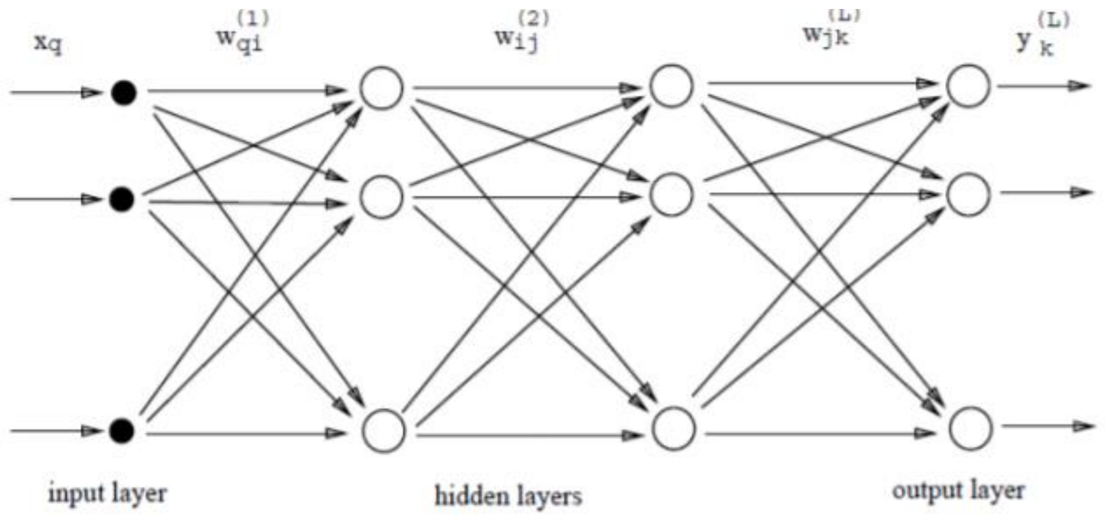


*Figure 3 - The Parts of an Artificial Neural Network* (Jain et al., 1996)

Each hidden layer contains several artificial neurons neuron units that in the basic scheme are fully connected to the neurons of the next layer; the artificial neuron output is the summed multiplication between the input vector X and a weights vector W. This sum goes through an activation function which determines if the neuron is activated and accordingly an output is generated.

## Activation function

Artificial neural networks are designed as universal function approximators and are capable to calculate and learn polynomial degree functions. Thanks to the non-linear activation functions, stronger learning of networks can be achieved.

The activation functions (equation 1) are executed after the input vector $x$ is multiplied with the weight vector $W$

$$x^{[l]} = f(W^{[l]} \cdot x^{[l-1]}) \qquad (1)$$

where $f$ is an activation function applied to each of its elements. Different nonlinear activation functions can be used (LeCun, Bengio, & Hinton, 2015) e.g. sigmoid (Han & Moraga, 1995) ,ReLU (Nair & Hinton, 2010) , Tanh (Nwankpa, Ijomah, Gachagan, & Marshall, 2018), softmax (Goodfellow, Bengio, & Courville, 2016)

### *Sigmoid*

A sigmoid function is a bounded differentiable real function that is defined for all real input values that have a positive derivative everywhere (Han & Moraga, 1995), (figure 4a). The output of the sigmoid logistic function is in the range (0,1) compared to (-inf, inf) of a linear function. Additionally, the sigmoid function provides the option to have a continuous output instead of a binary result such as signal/not signal according to some threshold. Due to the function shape most of its' outputs will be very close to the extremes of 0 or 1 (Nwankpa et al., 2018)

### *Relu*

Rectified linear unit (ReLU) is an activation function defined as the positive part of its argument (figure 4b). As long as the input has a value below zero, the output will be zero but, when the input rises above, the output is a linear relationship with the input variable (Nair & Hinton, 2010). The ReLU activation function has proven to work in many different situations and is currently widely used, considered as the most popular non-linear function since it typically learns faster than other activation functions in networks with many layers (LeCun et al., 2015).

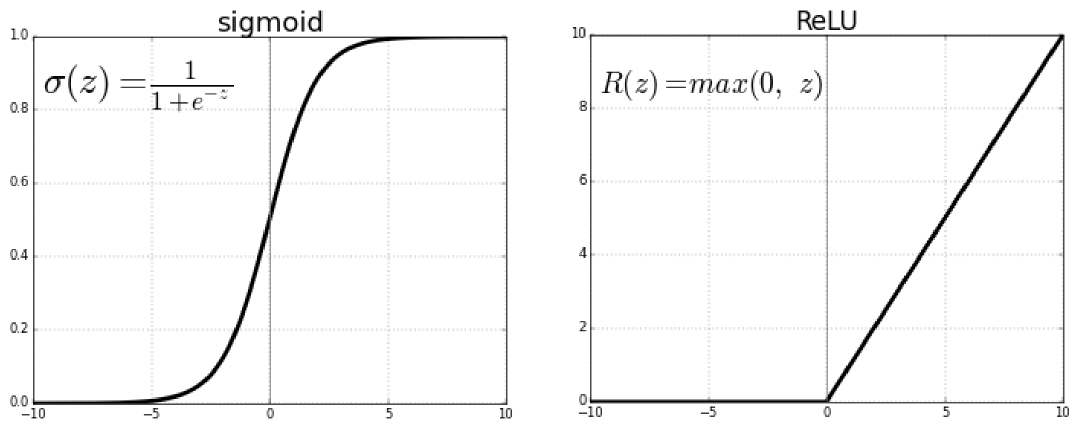


*Figure 4 - Sigmoid function Vs. Relu a) sigmoid, b) ReLU*

### *Softmax*

The softmax function takes a N-dimensional vector of real numbers and transforms it into a vector of a real number in range (0,1) (Goodfellow et al., 2016). The Softmax function (equation 2) is a soft version of the max function. Instead of selecting one maximum value, it will transform the values in the vector in a way that the larger input components will correspond to larger probabilities. Since the output is a probability distribution it is suitable for probabilistic interpretation in classification tasks.

$$P(y = j|x, W) = \frac{exp\left(x_j^L\right)}{\sum_{c=1}^{C} exp\ (x_c^L)} \tag{2}$$

## 2.2.3 Training neural networks

### Overview

The training process in neural networks, relates to learning the values of the parameters in the network (Wij - weights, Bj - biases), The training process include three main stages that operate in sequence in an iterative way: forward-propagation, back-propagation, and optimization (figure 5).

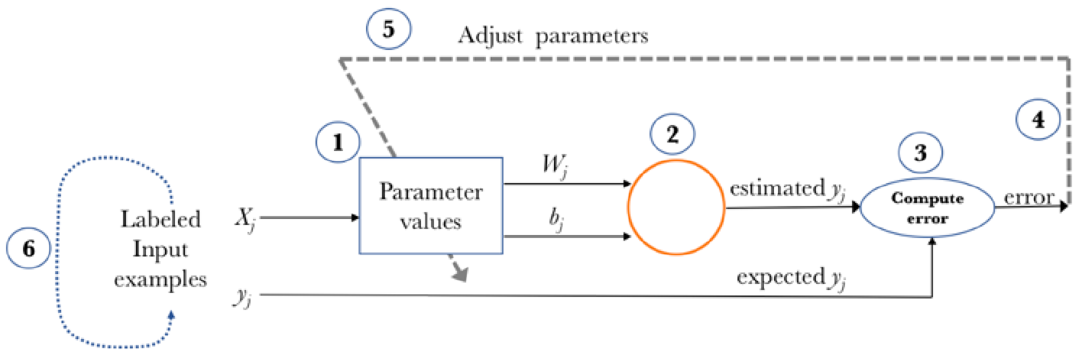


*Figure 5 – Main stages of neural network training process*

The training process (figure 5) includes the following steps:

1. Start with values (often random) for the network parameters (wij - weights and bj - biases).
2. **Forward-propagation** - Take a set of examples of input data and pass them through the network to obtain their prediction.
3. Compare these predictions obtained with the values of expected labels and calculate the **loss** with them.
4. Perform the **backpropagation** in order to propagate this loss to each and every one of the parameters that make up the model of the neural network.
5. **Optimize** the performance of the network by using the propagated information, update the parameters of the neural network with the gradient descent in a way that the total loss is reduced and a better model is obtained.
6. Continue iterating in the previous steps until a good model is obtained.

### Forward-propagation

The first forward-propagation phase occurs when the network is exposed to the training data. The input data passes through the network in such a way that all the neurons apply their activation function to the information they receive from the neurons of the previous layer and send it to the neurons of the next layer. When the data has crossed all the layers, and all its neurons have made their calculations, the final layer will be reached with a result of a label prediction for those input examples. Next, a loss function is used to measure how good the prediction result was in relation to the correct result.

### *Loss function*

A loss function quantifies how close a particular neural network is to the ideal weight during the training process. Therefore, it could be used as a target function that must be minimized. Ideally, the error should be zero, that is, without divergence between estimated and expected value. Therefore, as the model is being trained, the weights of the interconnections of the neurons will gradually be adjusted until good predictions are obtained. There are many types of loss functions, while each loss can be fitted to specific task (Janocha & Czarnecki, 2017). One of the most common losses in the classification task is the cross-entropy (Janocha & Czarnecki, 2017), also known as the log loss. The log loss presented in equation 3, measures the performance of a classification model whose output is a probability value between 0 and 1. The loss increases as the predicted probability diverges from the actual label. Hence, a perfect model would have a log loss of 0.

$$CE(\widehat{p_b}|y_b^{true}) = \begin{cases} -\log(\widehat{p_b}) & ,y_b^{true} = 1 \\ -\log(1-\widehat{p_b}) & ,y_b^{true} = 0 \end{cases} \qquad (3)$$

### Back-propagation

In order to optimize the performance of the network a gradient descent algorithm is used. The partial derivatives of the loss function are calculated and used to back-propagate to adjust each weight in the network in proportion to how much it contributes to overall error (Guo et al., 2016).  The propagation process done layer by layer, first, the output layer error is calculated and passed to the hidden layer before it. After calculating the hidden layer error, its error value is passed on back to the previous hidden layer before it. As we keep on moving back through the network, we use the chain rule at every layer to calculate the derivative of cost with respect that layer's weights. This resulting derivative tells us in which direction to adjust the weights to reduce overall cost.

### Optimization using Gradient descent

Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks (Ruder, 2016). The algorithms minimize a target function parameterized by a model's parameters by updating the parameters in the opposite direction of the gradient of the target function to the parameters. The gradient descent relies on the resulting derivative found for each layer at the back-propagation stage and on the learning rate $\varepsilon$ that determines the size of the steps we take to reach a local minimum (equation 4).

$$W_t = W_{t-1} - \varepsilon \frac{dL}{dW} \qquad (4)$$

At each iteration, the derivative of the loss with respect to the weights is calculated, the process continues until reaching a local minimum.

One of the main challenges of NN implementation is deriving the correct hidden layer size. When the number of neurons is not determined properly, the derived system does not generalize well to unseen instances (Kon & Plaskota, 2000). On the other hand, when too much nodes are used, overfitting may occur and the desired optimum may not be found at all (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Deriving the right quantity of neurons is discussed in a study by Kon and Plaskota (Kon & Plaskota, 2000).

## 2.3 Convolutional neural networks

### 2.3.1 Motivation

Convolutional Neural Networks (CNN), also known as convnets, are a specialized kind of an artificial neural network for processing data that has a known grid-like topology (Goodfellow et al., 2016). The name convolutional neural network comes from the mathematical operation the network executes. Convolution is a specialized kind of linear operation applied on two functions that produce a third function that expresses how the shape of one is modified by the other (Kim & Casper, 2013). In general, convolution is used to apply filters to signals (Burrus & Parks, 1985), to perform functions such as extracting edges (Rawat & Wang, 2017) and reducing unwanted noise (Gustafsson, Claesson, & Nordholm, 2001). In the context of NN convolution term is used to describe a spatially repetitive local filtering, where not all of convolution characteristics are pressed. CNN commonly applied to analyzing images for classification, the task of taking an input image and outputting a class (Wang et al., 2016). Images typically contain a large number of pixels ordered in matrix structure where each pixel composed from three layers of colors RGB. In order to analyze image with traditional ANN, the image structure must be reshaped. The network will dense all the layers into one-dimension, create a large input vector that will require a strong computational power in order to solve the task. In addition, by breaking the relations between the layers a vital information about the local features of the object is lost.  Since traditional ANN is limited in processing high dimensional features data and require a one-dimensional input vector, the convnets are established to overcome this limitation by using local filters, instead of fully connected layers that are used in traditional ANN.  The idea for using local filters arose from the fact that in images, pixels that are located far away from the object provides no further information about the object. Since objects are defined by their local structure, and not by pixel which is located in distant from them, it is possible to understand that meaningful structure is found within local image patches. Convolution leverages three important ideas that can help improve a machine learning system: sparse interactions, parameter sharing and equivariant (Ke et al., 2018)

#### Sparse interactions

The structural information contained in local regions of images motivated the use of patch-like connections between layers instead of full connections. Traditional neural network layers use matrix multiplication by a matrix of parameters with a separate parameter describing the interaction between each input unit and each output unit. This means that every output unit interacts with every input unit. Convolutional networks, however, typically have sparse interactions, this is the same as using connections where every weight is zero, except the weights within some patch region. The zeros represent the fact that information outside of the patch do not determine anything related to the filter's local area. Only the neurons within a filter would be fully connected to single neurons in the next layer (Ke et al., 2018) (see figure 6). By reducing the number of connections, the number of weight parameters is reduced and less computational power and memory is required (Ke et al., 2018) ,(Mahmood et al., 2017).

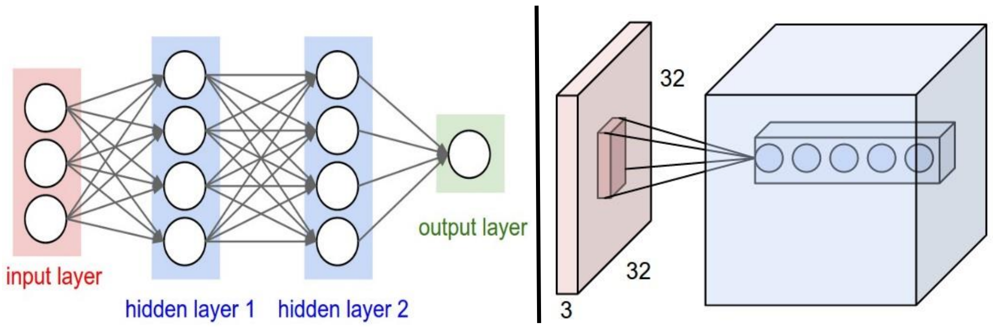


*Figure 6 - Fully connected ANN vs. CNN , in CNN Each neuron in the convolutional layer is connected only to a local region in the input volume spatially (Ke et al., 2018)*

## Parameter sharing and equivariant

Parameter sharing refers to using the same parameter for more than one function in a model. Rather than learning a separate set of parameters for every location, we learn only one global set that prevents overfitting of the network and reduces the number of parameters. In convnets, filters keep the same weights as they are convolved through various positions in a feature map. This means that a filter is using the same weights for detecting the same sort of feature at multiple locations in a feature map (figure 7). The parameter sharing causes the layer to have a property called equivariance to translation, shifting the image and then feeding it through a number of layers is the same as feeding the original image through the same layers and then shifting the resulting feature maps (Cohen & Welling, 2016). In other words, the symmetry is preserved by each layer, enabling to exploit it also in higher layers of the network.

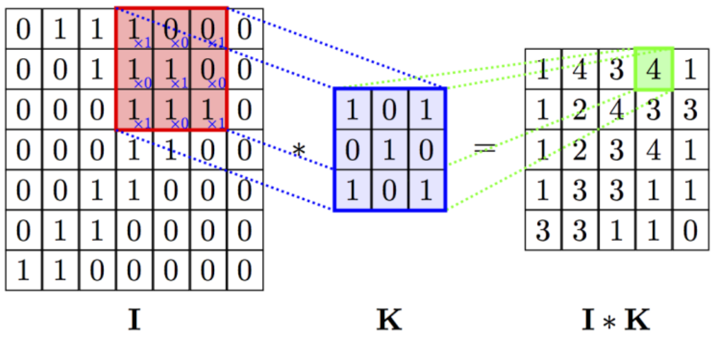


Figure 7 - A diagram expressing parameters sharing of a two-dimensional convolutional operator as an operation of sliding the same filter matrix (Kernel) across the target

Image and recording elementwise products into the feature map (I*K) *(Cohen & Welling, 2016)*

### 2.3.2 Convolutional neural networks architecture

There are several variations of CNN architectures (Koirala et al., 2019a). However, in general, most of them include three fundamental units (LeCun et al., 2015): convolutional layers, pooling layers followed by fully connected output layer, same as in a standard feedforward neural network. Usually convolutional layers and pooling layers are grouped together into one module, these modules are often stacked on top of each other to form a deep CNN model (Rawat & Wang, 2017). A typical architecture can be described by the next few steps (see figure 8): first an image is input directly to the network followed by several stages of convolution and pooling layers, each stage produces a feature map. Thereafter, representations from these operations feed one or more fully connected layers. Finally, the last fully connected layer outputs the class label. Development and application of novel CNN architectures have been investigated (LeCun et al., 2015) in order to improve image classification accuracy or reduce computational costs.

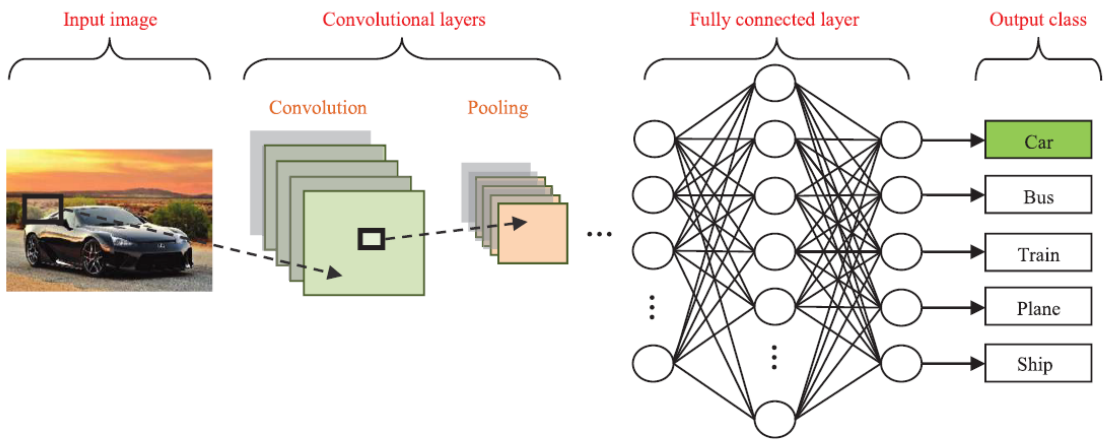


Figure 8 - A typical convolutional neural network architecture

#### Convolutional layer

The convolutional layers serve as feature extractors that learn the feature representations of the images (LeCun, Bottou, Bengio, & Haffner, 1998). The neurons in the convolutional layers are arranged into feature maps, each neuron in a feature map has a receptive field, which is connected to a neighborhood of neurons in the previous layer. The neuron in the feature map (**M**) is a result of convolution between the receptive field (**R**) and a specific filter (**K**) applied by nonlinear activation function ($\boldsymbol{f}$) (see equation 5). Traditionally sigmoid activation function were used, recently, rectified linear have become popular (LeCun et al., 2015).

$$M = f(R * K) \tag{5}$$

The principle of parameter sharing is reflected by using the same filter which convolve through all the pixels in the image and create a unique feature map. Each feature map represents a specific feature that can be located in several places in the image. However, different feature maps within the same convolutional layer have different filters so that several features can be extracted at each location (LeCun et al., 2015), (LeCun et al., 1998). By using the local receptive fields (figure 9), neurons can extract elementary visual features such as oriented edges, corners. These features are then combined by subsequent layers in order to detect higher-order feature.

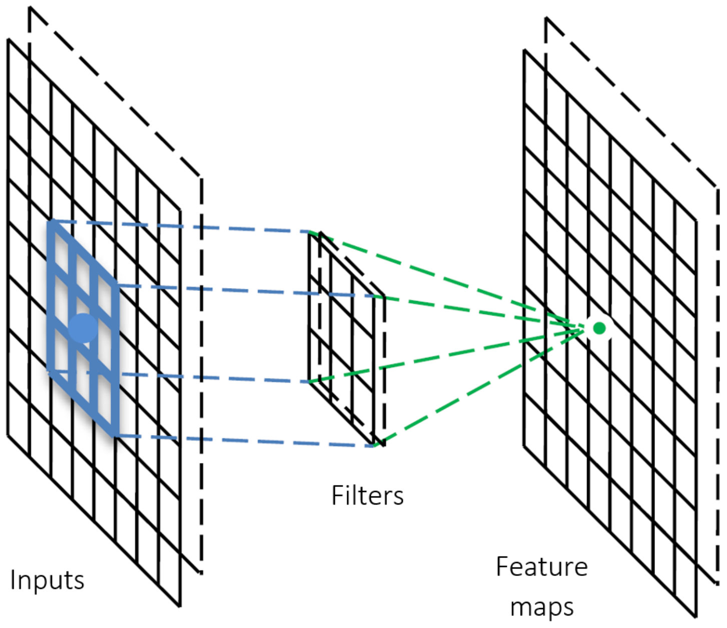


Figure 9 - Convolution operation between input image and filters, receptive field mark in blue

### Pooling layer

Often convolutional layers are followed by a pooling layer.  Pooling layers provide a summary statistic, by computing the maximum or average mathematical operation over a small region in the feature map (figure 10). This results in a subsampling of the input. Pooling reduces the dimensionality of the feature maps, increases the scale, and also supports translation invariance in the sense that a slight translation of the input does not change the output (Pai et al., 2017). Once the feature is detected, the exact location of the feature in the feature map is less important, an approximate position with the relative position to the other feature is essential. In general, the operation of maximum pooling is more effective than average pooling (Boureau, Ponce, & LeCun, 2010). It might expect that a better way to pool would be to keep the average intensity per pixel the same, rather than increasing the intensity of pixels after operating the maximum calculation. However, in practice, this case was discovered to be not correct.

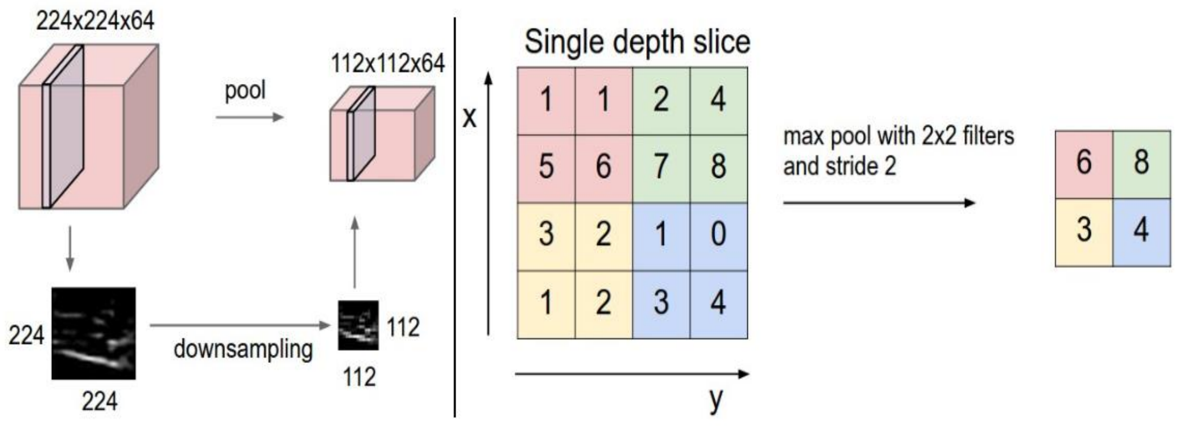


Figure 10 -Max pooling layer taken over 4 numbers (little 2x2 square).

## Milestone architectures in deep convolutional neural network

Several architectures in the field of convolutional networks have been developed as part of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where the main task is to correctly classify and detect objects and scenes. The first successful applications of convolutional networks were developed by Yann LeCun in 1990's by using an architecture called LeNet-5 (LeCun et al., 1995). The LeNet-5 architecture was used to read zip codes and digits, it includes two sets of convolutional and average pooling layers, followed by a flattening convolutional layer, then two fully-connected layers and finally a SoftMax classifier. A limitation of the network was the input size of image, the network could process only small images with 32x32 pixels. Follow the LeNet -5, the first work that popularized convolutional networks in computer vision was the AlexNet, developed by Alex Krizhevsky, Ilya Sutskever and Geoff Hinton (Krizhevsky, Sutskever, & Hinton, 2012). The AlexNet was submitted to the ILSVRC challenge in 2012 and significantly outperformed all other networks. The network had a very similar architecture to LeNet, but was deeper, bigger, and featured convolutional layers stacked on top of each other, previously it was common to only have a single convolution layer always immediately followed by a pooling layer. In ILSVRC 2013 winner was a ZFNet , the network was an improvement of AlexNet by tweaking the architecture hyper-parameters, in particular by expanding the size of the middle convolutional layers and making the stride and filter size on the first layer smaller. Improved performance of AlexNet as compared to LeNet was achieved by adding additional layers to the network. In 2014 two groups came with deep CNN architectures, VGGNet and GoogLeNet which had 16 and 22 layers respectively. The VGGNet main contribution was in showing that the depth of the network is a critical component for good performance. The VGGNet is a very simple structure (figure 11), it contains 16 layers that only performs 3x3 convolutions and 2x2 pooling from the beginning to the end. A drawback of the VGGNet is that it is more expensive to evaluate and uses a lot more memory and parameters, where most of these parameters are located in the first fully connected layer.

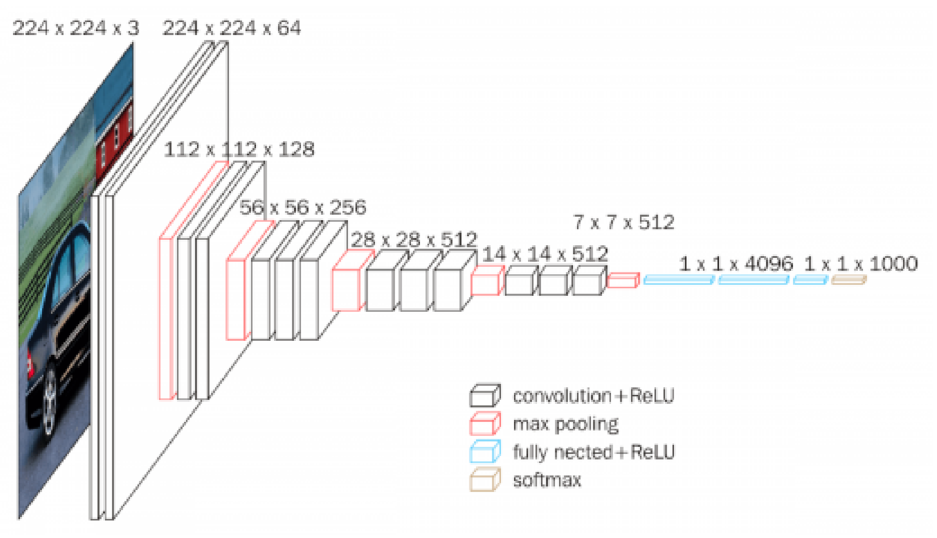


Figure 11 - the VGGNet-16 architectures, contains 16 layers

In parallel to the VGGNet , google released their own convolutional network named GoogLeNet (Szegedy et al., 2015). Its main contribution was the development of an Inception Module that dramatically reduced the number of parameters in the network. Additionally, in contrast to VGGNet, the GoogLeNet uses average pooling instead of fully connected layers at the top of the ConvNet, with that, it successfully eliminating a large number of parameters. The GoogLeNet was the winner of 2014 ILSVRC, it achieved a top-5 error rate of 6.67% which was very close to human level performance. Followed the VGGNet and GoogLeNet success researchers tried to perform much deeper networks. However, despite the conjecture that a deeper model should be able to perform at least as well as the shallower model, in practice, deeper networks start converging, resulting with a vanishing gradients problem exposed. With the increasing network depth, accuracy gets saturated and then degrades rapidly. The Residual Network (ResNet) developed by Microsoft group (He, Zhang, Ren, & Sun, 2016) overcame this problem by using special skip connections and a heavy use of batch normalization (figure 12), this architecture was the winner of ILSVRC 2015 with a top-5 error rate of 3.57% which beats human-level performance on this dataset. The idea behind the skip connection is to let the network to skip layers that do not improve network performance.

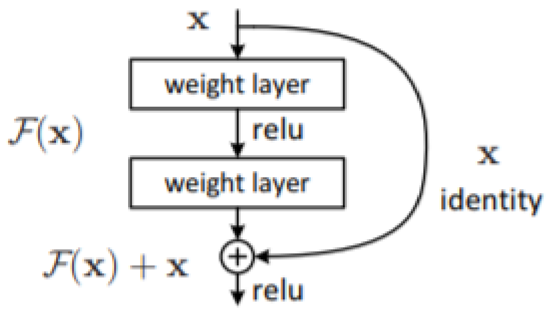

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
$\mathbf{x}$
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 12 - Residual unit with skip connection (He et al., 2016)

Previously, increasing network depth past a certain point was shown to actually decrease network performance. With residual networks, network performance continues to increase beyond depths at which regular convnets yielded decreased performance. With the skip connection technique, it was possible to train a convnets up to 152 layers. The ResNet also used as a pre-trained network (Lei et al., 2018), by training the network with transfer learning method, it can be implemented to solve different tasks.

## Training CNN with transfer learning

Transfer learning is a very popular method widely used among computer vision researchers (Huh, Agrawal, & Efros, 2016) when one has sufficient training data for one supervised learning task (the source/training domain), but only very limited training data for a second task (the target/test domain) that is similar but not identical to the first (Wang & Schneider, 2014). By using transfer learning, we try to store the knowledge gained in form of the feature extractor, while solving the source task in the source domain and then apply it to new problem of interest (figure 13). Since the training was performed on a large set of data, the knowledge that was gained is translated as a feature extractor in the first layers, those features are global and can be used also for different tasks, more specifically in tasks where the number of images is limited (Yosinski, Clune, Bengio, & Lipson, 2014). A very popular source domain is the ImageNet dataset, it contains 1.2 million high-resolution images over 1000 different classes (Krizhevsky et al., 2012). Using ImageNet pre-trained CNN features, impressive results were obtained on several image classification datasets (Donahue et al., 2014), (Sharif Razavian, Azizpour, Sullivan, & Carlsson, 2014), as well as in object detection tasks (Sermanet et al., 2013),(Girshick, Donahue, Darrell, & Malik, 2014).

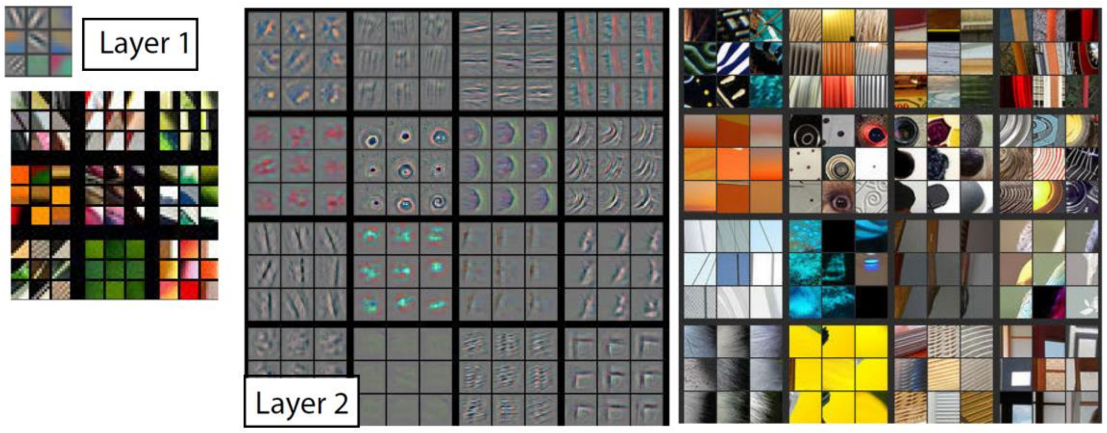


Figure 13 - Visualization of feature extractors at first and second layer.

The main reason of using transfer learning for tasks with small data is the problem of overfitting. Training a CNN from scratch on a small dataset will often lead to overfitting, since the number of parameters (neurons) in the CNN are much larger than the number of input images (LeCun et al., 2015). Transfer learning was implemented very successfully in such cases (Litjens et al., 2017). It requires to change the output layer according to the new task and retrains the network on the new dataset, a process noted as fine-tuning. As a result, the weights of the network will be updated according to the new task. In order to fine tune the network successfully additional methods such as data augmentation, dropout and regularization terms which helps to avoid overfitting are required.

### *Data augmentation*

Data augmentation is a technique that increases the variety of samples while not degrading the quality of the data set. As aforementioned, in order to train a deep neural network it usually needs a large amount of training data (Wang et al., 2014). With data augmentation more data that could be feed into the network is generated resulting in improved neural network performance and generalization by reducing overfitting. To augment an image data set, the original images can be flipped horizontally and vertically, and subsamples of the original images can be selected at random positions in the original. These subsampled images can then also be flipped horizontally and vertically (Agrawal, Girshick, & Malik, 2014).

### *Dropout*

Dropout is a technique developed for preventing overfitting of a network to the particular variations of the training set (Dahl, Sainath, & Hinton, 2013). The term “dropout” refers to dropping out units (hidden and visible) in a neural network with some defined probability, p. By dropping a unit out of the network (figure 14), the unit temporarily removes from the network, along with all its incoming and outgoing connections during the training process (Srivastava et al., 2014). After training is over, the units are replaced in the network with their original weights. This prevents the fully connected layers from overfitting to the training data set, and improves performance on the validation set.

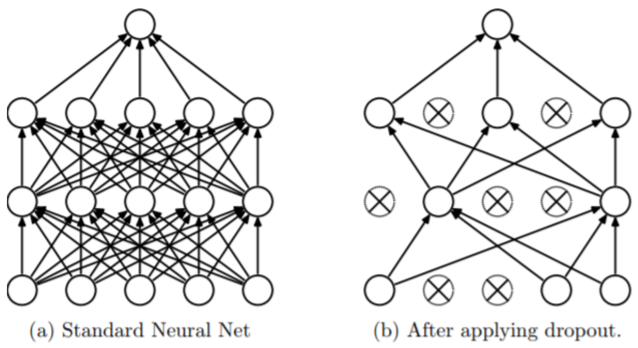


Figure 14 - Dropout neural network model (LeCun et al., 2015)

### *Weight decay*

Weight decay is another regularization mechanism for preventing networks from overfitting to the data. Weight decay adds a term to the loss function that suppresses any irrelevant components of the weight vector by choosing the smallest vector that solves the learning problem (Krogh & Hertz, 1992). A regularization technique called L2 regularization (equation 6), places a penalty on the sum of weights squared, with a weight decay parameter denoted $\lambda$. This sort of regularization results in smaller, more distributed weights, which reduces overfitting.

$$L(W,b) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\left(h_{W,b}\left(x^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{l=1}^{n_{l-1}}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 \quad (6)$$

Another technique, called L1 regularization (equation 7), places a penalty on the sum of the absolute values of the weights. L1 regularization promotes sparsity, and many weights end up being zero.

$$L(W,b) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\left(h_{W,b}\left(x^{(i)}\right) - y^{(i)}\right)^2 + \frac{\lambda}{2}\sum_{l=1}^{n_{l-1}}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left|W_{ji}^{(l)}\right| \quad (7)$$

Convolutional networks were some of the first neural networks to solve important commercial applications and remain at the forefront of commercial applications of deep learning today [26]. In recent years, CNN have become the leading architecture for most image recognition, classification and detection tasks (LeCun et al., 2015), (Rawat & Wang, 2017), (Mazzini, Buzzelli, Pauy, & Schettini, 2018), (Yao, Lei, & Zhong, 2019).

## 2.4 Object Detection

Object detection is a task related to computer vision, which includes two closely related sub-problems (Girshick, 2015). The first is the classification problem, where one dominant object in a given image should be determined and labelled. The second is the localization problem. The second problem is more complicated than the first, since, in addition to labelling the dominant object, it also must be localized in the image, usually by determining a bounding box around the image region that is occupied by the object and providing their coordinates. The difficulty of this task increases if not only one but all objects in an image must be labelled and multiple objects of the same category can appear in one image. A general and simple approach for object detection can be implemented using a sliding window (bounding box) over the image and classifying each box. The big drawback of this approach is that it is not applicable if the number of bounding boxes is unknown. Also, it is extremely expensive due to the huge search space. In order to reduce the amount of times that the classifier runs, it could be applied with a bigger window, but this has the risk of missing the ideal bounding box. An early method that implements the sliding window approach is the Viola-Jones detector (Wang, 2014), (Viola & Jones, 2001) which use Haar feature and AdaBoost to train a series of cascaded classifiers for face detection. The Viola-Jones detector that was first presented in (Viola & Jones, 2001) includes a cascade object detector composed of ensemble of a number of weak classifiers. The cascade object detector is arranged in stages with increasing complexity. The role of each stage is to decide whether the current windows is certainly NOT an object. If a stage decides that the current windows is not an object the rest of the stages are not evaluated, so only true object windows trigger the entire cascade of stages.

### 2.4.1 CNN based models for object detection

Deep learning convolutional neural networks have dramatically improved object detection performance as compared to previous methods (Girshick, Donahue, Darrell, & Malik, 2016). Below we present some of the most influencing CNN architectures for object detection. All the presented networks rely on pre-trained convolutional networks, such as ResNet or VGGNet, considered as 'backbone' networks that could be fine-tuned with processes of transfer learning to the relevant domain. The object detection task dominated by the CNN-based detectors can be roughly divided into two-stage and one-stage approaches.

#### The two-stage approach

The two-stage approach includes two parts, where the first one generates a sparse set of candidate object proposals, and the second one determines the accurate object regions and the corresponding class labels (figure 15).

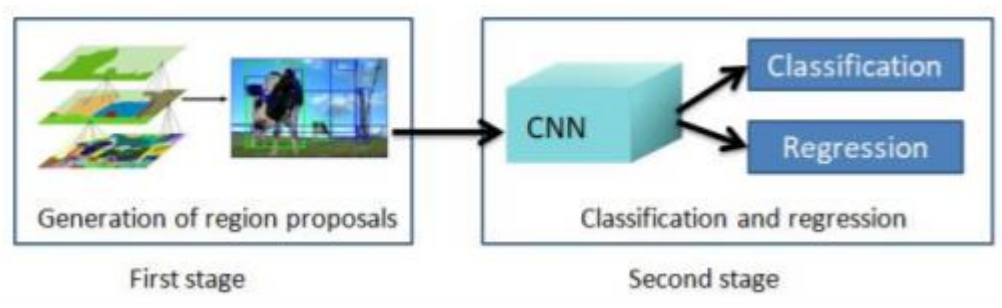


Figure 15 - Two stage approach

This approach was successfully introduced by Girshick et al. (Girshick et al., 2014) in their work called R-CNN. The R-CNN generates 2000 regions from the image based on a selective search algorithm (Uijlings, Van De Sande, Gevers, & Smeulders, 2013) and called them region proposals (figure 16.a). Next, each candidate region proposals are wrapped into a square and fed into a convolutional neural network. The CNN acts as a feature extractor and outputs a dense layer including the features extracted. These features are fed into an SVM to classify the presence of the object within that candidate region proposal (figure 16.a). In addition to predicting the presence of an object within the region proposals, the algorithm also predicts four values which are offset values to increase the precision of the bounding box. Despite the big success of R-CNN achieved, it has two main drawbacks. The first is related to the selective search algorithm which is not a learning algorithm. A second weakness is the long time to train the network since the image contains 2000 region proposals that should be classified.

The Fast R-CNN (Girshick, 2015) solves some of the drawbacks of R-CNN to build a faster object detection algorithm. Instead of feeding each region proposals to the CNN, the entire input image is fed to the CNN to generate a convolutional feature map. From the convolutional feature map, the region of proposals was identified (using selective search algorithm) and wrap them into squares, by using a region of interest (RoI) pooling layer the squares were reshape into a fixed size so that it can be fed into a fully connected layer. From the RoI feature vector, we use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box (figure 16.b). Due to the fact that the Fast R-CNN feeds the entire image to the network only one time instead of 2000 times and shares computation of convolutional layers between proposals for an image, the training time of Fast R-CNN is much faster. However, the test time of the network was not satisfying because it was not able to process in a real time. This is caused since the extraction of the region proposal is with algorithms that are not a part from the CNN architecture. Both previous detectors use selective search to find out the region proposals. As aforementioned, selective search is a slow and time-consuming process, affecting the network performance. Therefore, the Faster R-CNN replaces the selective search algorithm and lets the network learn the region proposals (Ren, He, Girshick, & Sun, 2015). A region proposals network (RPN) is a fully convolutional network that takes an image as input and outputs a set of anchors boxes (rectangles) candidate object proposals, each anchor has different aspect ratio and scale, so all the anchors are not similar. With the anchor, the RPN minimizes two loss functions: 1) binary classification loss - an object or not in the box. 2) regression loss for corrections of the anchors boxes coordinates. Similar to Fast R-CNN, the entire image is provided as an input to the convolution layers of Faster R-CNN to produce a convolutional feature map, then RPN is used to predict the region proposals. The predicted region proposals are then reshaped using a RoI pooling layer and fed into FC layers to classify the image within the proposed region and predict the offset values for the bounding boxes (figure 16.c). As a result, Faster R-CNN is much faster than the previous network and can be used for real-time object detection.

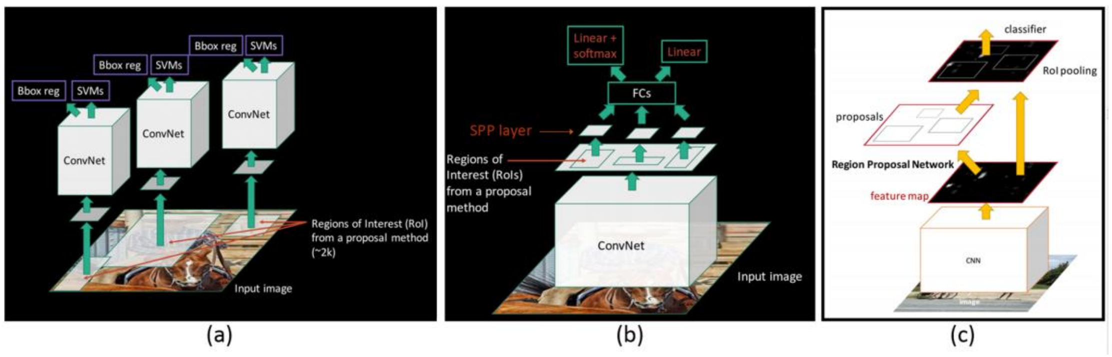


Figure 16 -The development of the R-CNN models: (a) R-CNN model, (b) Fast R-CNN model, (c) Faster R-CNN model

## The one-stage approach

So far, all the methods discussed handled detection as a classification problem. This was achieved by building a pipeline where first, object proposals are generated and then these proposals are sent to classification and regression stages. The network does not look at the complete image. Instead, it focuses on parts of the image which have high probabilities of containing the object. Several other methods pose detection as a regression problem. These methods implement a one-stage approach which produce bounding boxes in an end-to-end fashion. In this approach, classification and regression is done in a single shot using regular and dense sampling with respect to locations, scales and aspect ratio. The "You Only Look Once" (**YOLO**) (Redmon, Divvala, Girshick, & Farhadi, 2016) object detector was one of the first models that implemented the one-stage approach. It is considered an extremely fast network that uses a single feedforward convolutional network to directly predict object classes and locations. YOLO, (shown on figure 17), divides each image into a fixed S x S grid, within each cell of the grid we take B bounding boxes. For each of the bounding box, the network outputs a class probability and offset values for the bounding box. The bounding boxes having the class probability above a threshold value is selected and used to locate the object within the image.

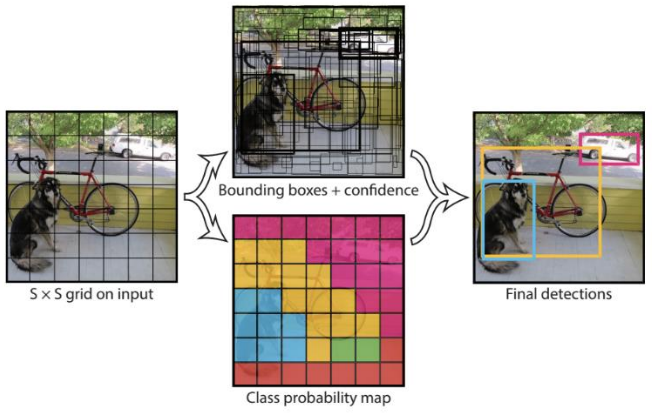


Figure 17 -The YOLO model (Redmon et al., 2016)

YOLO is a very fast network. However, it is not good at recognizing irregularly shaped objects or a group of small objects due to a limited number of bounding box candidates. The "Single Shot Detector" (SSD) (Liu et al., 2016) object detector is designed in a way that overcomes these problems and successfully achieves a good balance between speed and accuracy. The SSD network is able to detect objects of various sizes by using pyramidal feature hierarchy. SSD uses the VGG-16 model as a backbone, it takes an image as input which passes through multiple convolution layers with different filter sizes. Feature maps from conv layers at different positions of the network are used to predict the bounding boxes. They are processed by a specific convolution layers called extra feature layers, to produce a set of bounding boxes which are similar to the anchor boxes of the Fast R-CNN. For each bounding box (shown on figure 18), the model produces a vector of probabilities corresponding to the confidence over each class of object. In order to handle the scale, SSD predicts bounding boxes after multiple conv layers. Since each conv layer operates at a different scale, it is able to detect objects of various scales.

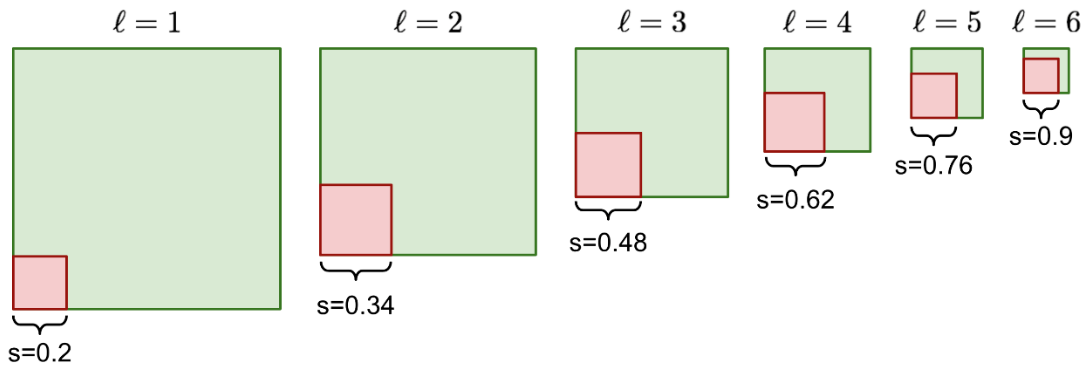


Figure 18 -An example of how the anchor box size is scaled up with the layer index $\ell$, the scale increase from 0.2 to 0.9 as moving forward (deeper) in the layer.

Although the one-stage detectors have made good progress, their accuracy are still lower than that of two stage methods (Lin, Dollár, et al., 2017). The main reason for this gap is that, two-stage detectors use a large amount of proposals for each object.
More specifically, one-stage detectors perform poorly in data with extreme class imbalance and also struggle to detect small objects. The RetinaNet is designed to solve these problems.

### 2.4.2 RetinaNet

Focal loss for dense object detection, known as **RetinaNet**, is a simple one-stage unified object detector which works on dense sampling of object locations in an input image. The model consists of a backbone network and two task-specific subnetworks. The backbone network relies on a pre-trained network such as ResNet and it implements the concept of feature pyramid network (FPN) (Lin, Dollár, et al., 2017). The output of the FPN is feed into two subnetworks, which perform convolutional object classification and convolutional bounding box regression.

#### FPN

Feature Pyramid Network (FPN) is a feature extractor designed for pyramid concept with accuracy and speed in mind. It generates multiple feature map layers that are used as multi-scale feature maps with better quality information than the regular feature pyramid for object detection. FPN includes a bottom-up and a top-down pathway.

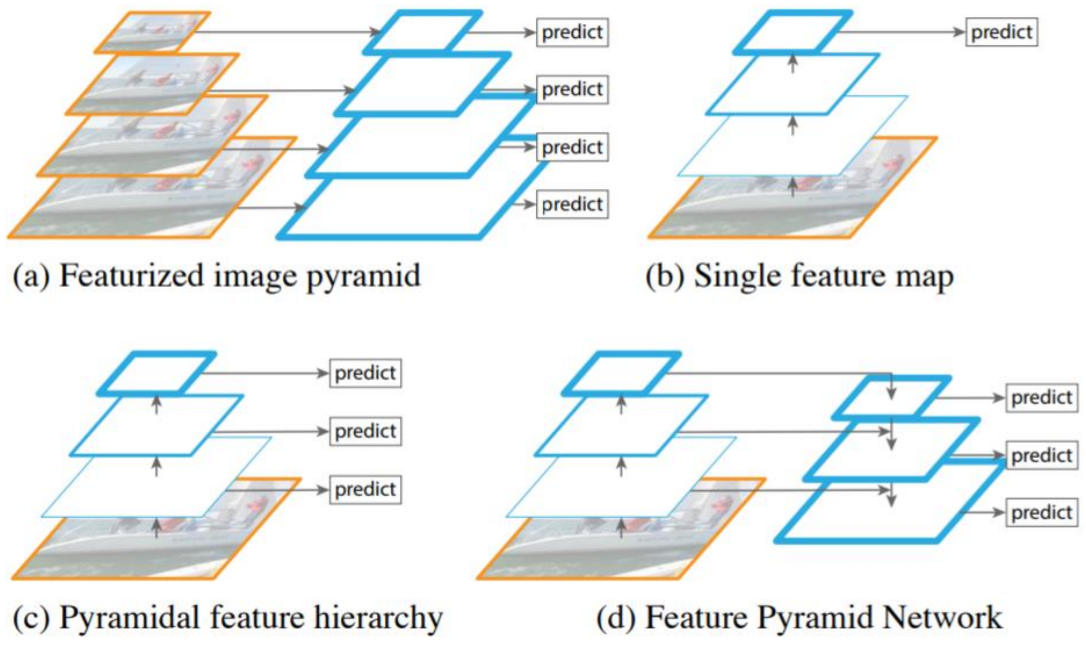

predict
(a) Featurized image pyramid
(b) Single feature map
(c) Pyramidal feature hierarchy
(d) Feature Pyramid Network

*Figure 19 - (a) Using an image pyramid to build a feature pyramid. Features are computed on each of the image scales independently, which is slow. (b) using only single scale features for faster detection, e.g. Fast-RCNN. (c) An alternative is to reuse the pyramidal feature hierarchy computed by a ConvNet as if it were a featurized image pyramid, e.g. YOLO. (d) proposed Feature Pyramid Network (FPN) is fast like (b) and (c), but more accurate, e.g. RetinaNet. feature maps are indicated by blue outlines and thicker outlines denote semantically stronger features.*(Lin, Dollár et al., 2017)

The bottom-up pathway is the regular convolutional network for feature extraction. As we go up, the spatial resolution decreases as well as the semantic value for each layer increases. As a result, by using the top layers more high-level structures can be detected. However, although the bottom layers are in high resolution there are not selected for object detection, since their semantic value is not high enough to justify its use. In addition, using these layers in this form will slow-down the speed of the network and will hurt the performance. Hence, only upper layers used for detection. As a result small objects are hard to detect.
The top-down pathway constructs higher resolution layers with a semantic rich layer. Using nearest neighbor up sampling, the last feature map from the bottom-up pathway is rescaled to the same scale as the second-to-last feature map. These two feature maps are then merged by element-wise addition to form a new feature map. This process is iterated until each feature map from the bottom-up pathway has a corresponding new feature map connected with lateral connections. The top-down pathway and lateral connections produces a multi-scale feature representation in which all levels are both semantically and spatially strong, including the high-resolution levels. As a result, this property enables a model to detect objects across a large range of scales by scanning the model over both positions and pyramid levels, providing better performance in accuracy.

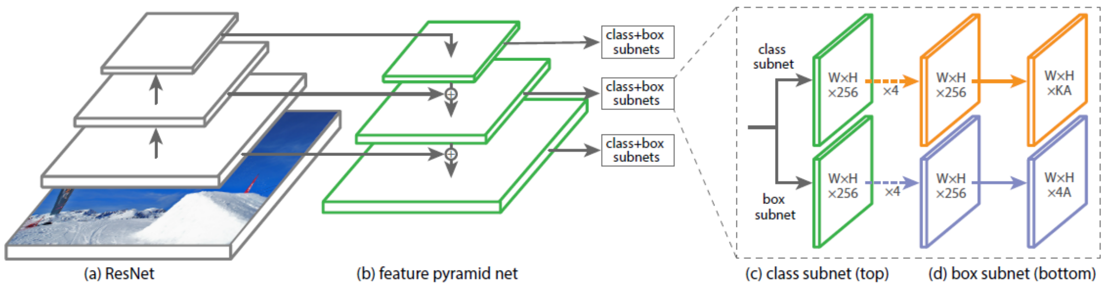


*Figure 20 - RetinaNet structure, (a) using ResNet as backbone network, (b) FPN, (c) Following the FPN, at each feature map layer anchor boxes are generated in different sizes and aspect ratio, feeding into two subnetworks. The first performs convolutional object classification and the second performs convolutional bounding box regression.*

Detecting a small object at high-resolution levels creates a problem since foreground and background classes are extremely imbalanced. As a result, it hurts the performance of the detector since most of the proposal regions can be easily classified as background and contribute no useful learning information, since the large portion of the input is background, they can overwhelm the loss and computed gradients and lead to degenerated models. The RetinaNet overcomes this problem by using a re-designed loss function named Focal loss for the classification task.

## Focal loss

The focal loss (FL) is a novel loss function that was first proposed in (Lin, Goyal, Girshick, He, & Dollár, 2017) in order to handle one stage object detection scenario where the network suffers from extreme foreground and background class imbalance (equation 8). The loss function is reshaped to down-weight easy examples and thus focuses training on hard negatives. To do this a modulating factor $(1-\widehat{p_b}\,)^{\gamma}$ is added to the original cross-entropy loss function. Additionally, an $\alpha$ parameter is added to deal with the imbalanced number of examples per class, but it doesn't contributes to differentiate between easy and hard examples.

$$FL(\widehat{p_b}|y_b^{true}) = \begin{cases} -\alpha * (1-\widehat{p_b}\,)^{\gamma} * \log(\widehat{p_b}) & , y_b^{true} = 1 \\ -(1-\alpha) * \widehat{p_b}^{\gamma} * \log(1-\widehat{p_b}) & , y_b^{true} = 0 \end{cases} \tag{8}$$

When an example is misclassified and $\widehat{p_b}$ (probability of ground truth class y) is small, the modulating factor is near 1 and the loss is unaffected. As $\widehat{p_b}$ goes to 1, the factor goes to 0 and the loss for well-classified examples is down-weighted. The focusing parameter $\gamma$ smoothly adjusts the rate at which easy examples are down-weighted. When $\gamma$ = 0, focal loss is equivalent to cross-entropy. When $\gamma$ is increased, the effect of the modulating factor is likewise increased.

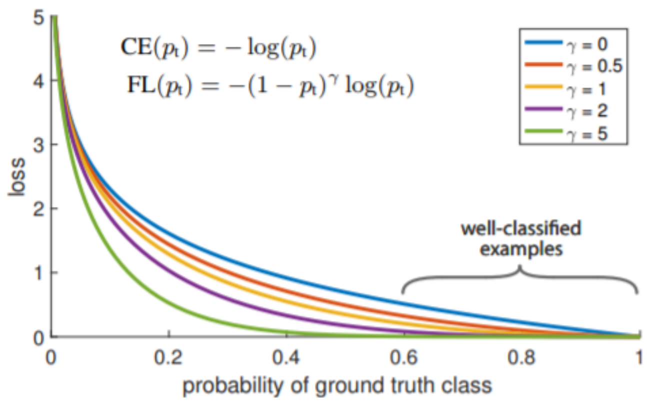


Figure 21 - Focal loss Vs. Cross-entropy loss ($\gamma = 0$) (Lin, Goyal, et al., 2017)

Since objects in the image can be predicted by multiple anchor boxes, RetinaNet selects at most 1000 anchor boxes per class with the highest probability score from each FPN level. In order to remove duplicated anchor boxes, it applies, non-maximum-suppression (NMS) (Neubeck & Van Gool, 2006) algorithm for each class, which iteratively chooses an anchor box with the highest probability score and removes any overlapping anchor boxes with an intersection over union (IoU) greater than 0.5 (Ren et al., 2015),(Girshick et al., 2014). In the final stage, for each remaining anchor, the regression subnet gives offset predictions that can be used to refine the anchor to obtain a bounding box prediction. Similar to the Fast R-CNN regressor, the convolutional bounding box regression subnet output four numbers, the first two numbers specify the offset between the centers of anchor and ground-truth object, while the last two numbers specify the offset between the width and height of the anchor and the ground-truth. The second subnetwork performs convolutional object classification.

## 2.5 Image detection in agriculture using deep learning

Agricultural objects detection is an extremely challenging task due to the highly variable and unstructured nature of both the objects (variable shape, color, size) and environment (illumination, adverse weather ) and the presence of obstructions (leaves, branches, dust, insects) and other unavoidable image noises (Pereira et al., 2017), (Koirala et al., 2019a), (Kamilaris & Prenafeta-Boldú., 2018).

Fruits detection based on supervised machine learning methods has yielded better results than simple image processing techniques (Koirala et al., 2019b), (Guo et al., 2016), which were based on hand-engineered features to encode visual attributes. Although it requires a strong computational resource and a great resource of labelled data for training, it provides a promising method. Detectors have been developed with a wide variety of strategies, from classification by using low-level keypoint extractions to segmentation and detection (Koirala et al., 2019b), (Guo et al., 2016), (Kamilaris & Prenafeta-Boldú., 2018) . A summary of research that uses deep learning with convolution neural network approach as the main method of recognition is presented in the following section, in particular - classification, object detection and yield estimation.

### 2.5.1 Object classification

Various studies in the agriculture field have applied classification using convolution neural networks (Guo et al., 2016), (Koirala et al., 2019a), (Kamilaris & Prenafeta-Boldú., 2018), (Pereira et al., 2017). The studies (Tables 1, 2) were performed on different types of crops and used various techniques that combine convolution neural network algorithms, starting from basic networks AlexNet (Reyes, Caicedo, & Camargo, 2015), (Lee, Chan, Wilkin, & Remagnino, 2015), (Yalcin, 2017) and LeNet (Amara, Bouaziz, & Algergawy, 2017) up to deep neural networks such as VGGNet (Mortensen et al., 2016), (Dyrmann, Karstoft, & Midtiby, 2016) and GoogleNet (Mohanty, Hughes, & Salathé, 2016). Others try to build CNN classifiers by themselves (Hall, McCool, Dayoub, Sunderhauf, & Upcroft, 2015), (Kussul, Lavreniuk, Skakun, & Shelestov, 2017), (Grinblat, Uzal, Larese, & Granitto, 2016). Since standard convolution neural network were trained on regular RGB images, the various research on crop images was done mostly on RGB cameras. In these images the crop was located at the middle of the images and captured most of the image space. Images from a satellite multi-spectral camera were analyzed using a custom developed CNN that achieved 0.946 accuracy (Kussul et al., 2017).

Table 1 - Classification tasks with CNNS for different types of land crops.

| Crop | Sensors | Environment | # train images | # test images | Algorithm | Results | Ref |
|---|---|---|---|---|---|---|---|
| 33 types of leaves | Digital RGB camera + scanner | Laboratory | 1.2 million (Huh et al., 2016) | 1907 | Author-defined CNN + Random Forest | 0.973 ±0.006 Accuracy | (Hall et al., 2015) |
| 15 types of fruit leaves | Web images | Laboratory | 30880 | 2589 | CaffeNet | 0.96 Accuracy | (Sladojevic, Arsenovic, Anderla, Culibrk, & Stefanovic, 2016) |
| 14 types of crop Leaves | Mobile phone HD camera | Laboratory | 43445 | 10861 | GoogleNet | 0.9935 F1 score | (Mohanty et al., 2016) |
| Banana Leaves | Wtandard digital camera | Field | 1850 | 1850 | LeNet | 0.9971 F1 score | (Amara et al., 2017) |
| Wheat, maize, soybeans sunflower and sugar beet | Multi-spectral camera acquired from Landsat-8 and Sentinel-1A RS satellites | Field | 274 | 273 | Author-defined CNN | 0.946 accuracy | (Kussul et al., 2017) |
| Barley weed | Sony a7 with a 35 mm lens | Field | 36 | 12 | VGGNet | 0.79 accuracy | (Mortensen et al., 2016) |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 22 different types of crops | Canon IXUS 220 HS with a CMOS de 12.1 MP sensor and a 24 mm equivalent focal length | Field | 180 | 36 | CNN-HistNN network | 0.9 F1 score | (Rebetez et al., 2016) |
| 7 views of different plants: entire plant, branch, flower, fruit, leaf, stem and scans | RGB digital camera | Field | 91758 | 21446 | AlexNet | 0.487 Precision | (Reyes et al., 2015) |
| 44 different plant species | - | Laboratory | - | - | AlexNet | 0.996 Accuracy | (Lee et al., 2015) |
| White bean | Hewlett Packard Scanjet-G 3110 scanner | Laboratory | 155 | 17 | Author-defined CNN | 0.9 Accuracy | (Grinblat et al., 2016) |
| Red bean | | | 245 | 27 | | 0.983 Accuracy | |
| Soybean | | | 380 | 42 | | 0.988 Accuracy | |
| 91 types of weed seeds | Standard digital camera | Laboratory | 3155 | 825 | PCANet | 0.91 F1 score | (Xinshao & Cheng, 2015) |
| 22 types of weed | Data provided from different RGB cameras and scanners | Laboratory | 6248 | 4165 | VGG16 | 0.86 Accuracy | (Dyrmann, Karstoft, et al., 2016) |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Wheat | Standard digital camera | Open-field | - | - | AlexNet | 0.8368 Accuracy 0.8364 F1 score | (Yalcin, 2017) |
| Barley | | | | | | 0.7843 Accuracy 0.7715 F1 score | |
| Lentile | | | | | | 0.7417 Accuracy 0.7376 F1 score | |
| Cotton | | | | | | 0.8658 Accuracy 0.8654 F1 score | |
| Pepper | | | | | | 0.8728 Accuracy 0.8714 F1 score | |
| Corn | | | | | | 0.8658 Accuracy 0.8654 | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | F1 score | |
| Wheat and barley | Canon PowerShot G15 camera | Field | 4000 | 500 | DenseNet | 0.97 Accuracy | (Sørensen, Rasmussen, Nielsen, & Jørgensen, 2017) |
| Wheat | Consumer grade 12MP camera | Field | 415 | 105 | Stacked hour- glass CNN | 0.83-0.89 for spikes 0.88-0.96 for spikelets, F1 score | (Pound, Atkinson, Wells, Pridmore, & French, 2017) |

### 2.5.2 Object detection and yield estimation researches in agriculture

Recently, yield estimation has gained particular attention and the latest works have shown impressive results in different scenarios, such as apple orchards and mango orchards (Koirala et al., 2019a). Much of the published work has focused on improving the accuracy of algorithms to accurately predict the number of fruits within images. Less work that relates image fruit counts to actual yield have been reported (Koirala et al., 2019a). In the majority of those research the problem of yield estimation is devised as a fruit detection task that can be formulated as a more generic object counting problem and solved either indirectly by using object detectors (Bargoti & Underwood, 2017a) or explicitly with architectures that learn to count and set up a regression problem to directly infer the number of object instances in the image (Rahnemoonfar & Sheppard, 2017a), (Chen et al., 2017). Since object detection is essential for counting, learning schemas such as based on deep convolution neural networks, are taking a significant part on both approaches of yield estimation, counting by detection and counting by regression.

#### Counting by detection

Counting by detection is considered straightforward. In order to do yield estimation, it is required to use a detector that detects all the instances of the object in the scene followed by counting all the objects. Convolution neural network detectors have been successfully used in the context of supervised fruit counting for yield estimation (Koirala et al., 2019a). The first report which uses deep learning with CNN for fruit detection was by Sa et al. (Sa et al., 2016) . The objective of the research was to create a neural network that would be used by autonomous robots that can harvest fruits. These authors implemented a multi-modal extension of faster R-CNN architecture which was trained on color RGB and Near-Infrared (NIR) images. The multimodal network obtains fruit detection and performs yield estimation much better than the existing networks (obtained a 0.84 F1 score). Another study that used Faster R-CNN for yield estimation in context of fruit counting was presented by Bargoti and Underwood (Bargoti & Underwood, 2017a). The study included detection of fruits, in apple, almond and mango orchards and resulted in detecting apples and mangos, with a F1 score >0.9, The results were superior to work using pixel wise CNN segmentation and regression for apple counting (Bargoti & Underwood, 2017b) which achieved a 0.861 F1 score. Additional examples of research that employed a deeper CNN for counting yield using object detection pre-define networks can be found in (Liang et al., 2018),(Bresilla et al., 2019) and (Lamb & Chuah, 2018) where the use of architectures of YOLO and SSD were implemented for detection of mango (Liang et al., 2018), apple (Bresilla et al., 2019) and strawberry (Lamb & Chuah, 2018) resulting in F1 scores of 0.91, 0.9 and 0.842 average precision (AP) respectively. A recent research, present by Kestur et al. (Kestur, Meduri, & Narasipura, 2019), proposed a new deep CNN named 'MangoNet'. The network used a full convolutional deep CNN to segment mango fruit in the images followed by connected object detection for fruit counting in images. Results of the network demonstrate the robustness of detection for a multitude of factors characteristic to open field conditions such as scale, occlusion, distance and illumination conditions.

### Counting by Regression

Another approach to provide yield estimation is counting by regression. This approach aims to directly map visual features extracted globally or locally from image patches to the number of object instances (Koirala et al., 2019a). With this approach, the model explicitly learns to count not only those crops that are seen in the image but also those that are occluded, by involving calculation of a correction factor for the occluded fruits, which results in more robust and precise estimations. A very common and simple implementation of the approach is to use the slope of the linear regression between the machine vision image count and harvest count for a set of calibration trees for fruit load estimation. Among the research that implemented this approach, the most significant works that use the regression approach for yield estimation, resulting in more than 0.9 accuracy were (Rahnemoonfar & Sheppard, 2017a) and (Chen et al., 2017). Additional research by (Stein, Bargoti, & Underwood, 2016), (Koirala et al., 2019b) also reported promising results using this approach. In (Chen et al., 2017) the authors designed a count architecture composed from three stages. In the first stage, a blob detector based on a fully convolutional network (FCN) extracts candidate regions in the images. A counting algorithm based on a second convolutional network then estimates the number of fruits in each region. Finally, a linear regression model maps that fruit count estimate to a final fruit count. The presented architecture was trained on oranges and apples and achieved 0.97 and 0.91 ratio count respectively.

In (Rahnemoonfar & Sheppard, 2017a), the researchers implemented a tomato fruit counting task. They proposed a modified version of Inception-ResNet architecture to output the fruit count without detection and localization of objects. The model was trained entirely on synthetic tomato fruit images and tested on natural images. The algorithm was robust to varying degrees of lighting conditions, occlusions and fruit overlaps and reached 91% average test accuracy. Both (Stein et al., 2016), (Koirala et al., 2019b) deal with the task of fruit detection, localization and yield estimation in a mango orchard. In the first research (Stein et al., 2016), the authors combine the Faster R-CNN model for object detection with a monocular multi-view tracking module that relies on a GPS system to locate the fruit in 3D dimension, enabling a number of spatial statistics per tree. This improved the yield estimation resulting in an error rate of only 1.36% for individual trees. The second research was published recently (Koirala et al., 2019b) and dealt with comparing detection results for between different architectures: Faster-RCNN, SSD and YOLO, resulting with F1 scores of 0.939, 0.959 and 0.968 respectively. Additionally, the authors redesigned the YOLO architecture by passing the information from early detection layers to that of the later detection layers. This architecture known as MangoYOLO outperformed the previous one with 0.968 F1 score for mango detection.

Table 2 – Agriculture object detection research using deep learning

| Crop | Sensors | Environment | # train images | # test images | Algorithm | Results | Ref |
|---|---|---|---|---|---|---|---|
| Sweet pepper and rock melon | Multi-Spectral and RGB cameras, the JAI AD 130GE and Microsoft Kinect 2 | Greenhouse | 209 | 48 | Faster R-CNN | 0.84 F1 score | (Sa et al., 2016) |
| Apple | PointGrey LadyBug + strobe lightning | Orchards | 729 | 112 | Faster R-CNN | 0.904 F1 score | (Bargoti & Underwood, 2017a) |
| Mango | Prosilica GT3300c + strobe lightning | | 1154 | 270 | | 0.908 F1 score | |
| Almond | Handheld Canon EOS60D | | 385 | 100 | | 0.775 F1 score | |
| Tomato | Synthetic generated images | Field | 24,000 | 2,400 + 100 real images | modified Inception-ResNet | 0.91 Accuracy | (Rahnemoonfar & Sheppard, 2017b), (Rahnemoonfar & Sheppard, 2017a) |
| Wheat plants root | Nikon D5100 DSLR camera | Field | 2,500 | 20 | Author- defined CNN | 0.984 Accuracy | (Pound, Atkinson, Townsend, et al., 2017) |
| Wheat plants shoot | Canon 650D cameras | | 1,664 | 20 | | 0.973 Accuracy | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Orange | Bluefox USB 2 camera at 10 Hz | Orchards | 36 | 35 | FCN+CNN + regression | 0.91 Ratio counted | (Chen et al., 2017) |
| Apple | PointGrey USB 3 camera at 6 Hz | | 11 | 10 | | 0.97 Ratio counted | |
| Apples | Point Grey Lady-bug3 spherical digital video camera containing six 2MP cameras oriented to capture a complete 360-deg panoramic view. | Orchards | 1000 | 100 | Pixel Wise CNN | 0.861 F1 sore | (Bargoti & Underwood, 2017b) |
| Strawberry | Nikon COOLPX S6500 camera | Greenhouse | 298 | 75 | Author- defined CNN | 0.88 mAP | (Habaragamuwa et al., 2018) |
| Mango | Prosilica GT3300c + strobe lightning | Orchards | 1154 | 270 | SSD based on VGG | 0.91 F1 score | (Liang et al., 2018) |
| Apple | webcam, DSLR camera, smartphone | Orchards | 50 | 130 | YOLO | 0.9 F1 score | (Bresilla et al., 2019) |
| Peach, Apple, orange | standard digital camera | Orchards | - | - | Faster R-CNN | 0.915, 0.942, 0.902 AP | (Tao, Zhou, Wang, & Shen, 2018) |
| Strawberry | RGB camera | Field | 3640 | 910 | SSD | 0.842 AP | (Lamb & Chuah, 2018) |
| Green citrus | RGB camera | Orchards | 1200 | 300 | Faster R-CNN | 0.855 mAP | (XIONG et al., 2018) |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Mango | Basler, Canon 5 Mega-pixel RGB camera and Kinect camera | Orchards | 1430 | 300 | MangoYOLO based on YOLO | 0.968 F1 score | (Koirala et al., 2019b) |
| Mango | spectrum camera (RGB) | Orchards | 11,096 | 1500 | MangoNet based on CNN | 0.84 F1 score | (Kestur et al., 2019) |
| Apple, litchi, navel orange, Huangdi gan | Standard digital camera | Orchards | - | - | SSD based on ResNet | 0.96 F1 score | (Koirala et al., 2019a) |
| 22 types of weeds | Standard digital camera | Field | 1274 | 153 | DetectNet based on GoogLeNet | 0.603 F1 score | (Dyrmann, Jørgensen, & Midtiby, 2017) |
| Sugar beets and weeds | JAI camera in nadir view capturing RGB+NIR images | Field | 1674 | 295 | Author- defined CNN | 0.97 F1 score | (Milioto et al., 2017) |
| Weeds | Multi-spectral camera, JAI AD 130 | Field | 1,600 | - | Author- defined CNN | 0.913 mAP | (Potena, Nardi, & Pretto, 2016) |
| Weeds, soil or maize crop | Simulated images | Field | 6744 | 1686 | VGGNet-16 | 0.94 accuracy | (Dyrmann, Mortensen, Midtiby, & Jørgensen, 2016) |
| Mango | Prosilica GT3300c + strobe lightning and Velodyne HDL64E 3D LiDAR | Orchards | 1250 | 250 | Faster R-CNN | 0.881 F1 score | (Stein et al., 2016) |

In this work we will try not only to detect all the melons in the field but also estimate the total yield weight. To date, most yield estimation research focused on fruit counting. Below we review research that estimated fruit weight using computer vision.

## Fruit weight estimation using computer vision

Estimating total yield in terms of weight is a complex task that requires estimation of fruit size together with the fruit amount in field. This require first fruit detection. Fruit weight can be correlated to fruit dimensions in many fruits. Such a relationship allows fruit weight to be estimated using machine vision, given a measure of camera to fruit distance (Koirala et al., 2019a).

This relationship was implemented in (Stajnko, Rakun, & Blanke, 2009) where the authors proposed a model for estimating apple fruit yield at harvest, based on segmentation that relied on color, shape and texture image analysis. The model provided simple counting of apples and measurement of their diameters needed for objective modelling of forecasting yield. The forecasting yield was based on the formula:

$$Y_t = N \times a \times D^b \tag{8}$$

where $Y_t$ represents the yield per tree, N the number of fruits per tree, D average diameter and a, b constants depending on apple variety (Stajnko et al., 2009). Another researcher that uses the correlation between weight fruit and its lineal dimensions was presented in (Calixto, Neto, da Silveira Cavalcante, Aragão, & de Oliveira Silva, 2019), where the weight of yellow melons were estimated using an RGB image. The authors detect melon's contours by using an image processing method called Otsu (Otsu, 1979) which separates the melon from the image background, allowing to calculate the area of the melon. By using area-based linear regressions, it was possible to predict the weight of each melon in kilograms (Calixto et al., 2019) followed by estimating the total yield weight. Results indicated an average absolute error inferior to 0.05 kg in weight measurement.

ANN has been used to predict total weight by regressing the inputs of fruit segmentation. In (Cheng, Damerow, Sun, & Blanke, 2017) , the researchers extract four features from segmented image : total cross-sectional area of fruits, fruit number, total cross-section area of small fruits and cross-sectional area of foliage. The features were used as inputs to a back propagation neural network (BPNN) which predicted the total weight yield. Another study that used ANN for estimating fruit yield (Rahman, Robson, & Bristow, 2018) the potential of high-resolution satellite imagery to estimate yield of mango by integrating both geometric (tree crown area) and optical (spectral vegetation indices) data using artificial neural network (ANN) model was evaluated.

For the best of our knowledge, deep learning has yet to be applied to such tasks of fruit estimation (Koirala et al., 2019a). In practice, there are several recent researches that deals with the task of estimating fruit yield in terms of weight, with the use of computer vision methods such as image segmentation or implementation of Otsu's method. However, most of the current work focus on yield estimation based on fruit counting.

# 3. Methods

The research goal was to develop a pipeline algorithm for detection and yield estimation of melons in the agricultural environment using color images acquired from a digital camera mounted on an unmanned aerial vehicle. The system receives as input an aerial RGB image of a melon field, and the output is a report that includes each melon's location and weight.

In order to build such a system, we performed an experiment in which we acquired images (2018) for system training. In addition, we used images from two previous seasons (2016, 2017) to enrich the data we have in our hands and to provide a more robust system.

As part of the development, two different systems have been developed. The first system (described in chapter 4) relies on an algorithm pipeline composed of a classical and light CNN algorithm that requires less computational effort. To solve drawbacks that arise from the classical region proposal and the nature of a hybrid classical/CNN scheme used in the first system a second system was suggested. This system relies on an advanced deep learning classification schema requiring greater amount of computational power (described in chapter 5).

Both systems consist of the three following sequential main stages: 1) melon recognition\detection, 2) feature extraction, 3) yield estimation. The implementation of the first two stages in each system was different. The yield estimation stage was identical in both systems and was based on a linear regression.

## 3.1 First method, research narrative and training

For training and testing of the first system images from 2016, 2017 were used (2 images from 2016 and 3 images from 2017). During the training of the second system we used only 4 images from 2018 season, which contain an overall of 4000 labeled melons. For testing the performance of the second system, we selected different environments from all three seasons. One image from each season of 2016 and 2017 and two images from 2018 season were used for the testing.

The difference in the systems resulted in differences in trainings needs. Starting with the first system, (chapter 4) the first step of the algorithm dealt with melon recognition. Since the melons cover only a small portion of the field, there are relatively rare targets among other possible objects in the UAV image. To reduce unneeded computational effort, the recognition process for a melon target was split into two sub-stages: region proposal, and region classification. Regions of interest (ROI) were found using the Viola–Jones (VJ) algorithm. For the classification task an CNN pretrained network was used. The recognition process was trained with 3031 negative images (without a melon) and 1933 positive images (with melon) that were extracted from 2016 and 2017 season images. During the training process several augmentation techniques were applied to increase the number of training samples to 44,256 positive images and 43,168 negative images (without a melon). After finding a candidate for ROI, each proposed region undergoes a classification process using a pretrained CNN trained by transfer learning, resulting in an average precision of 0.82 and F1 score of 0.85.

For each ROI classified as a melon, the melon's geometrical features were extracted by fitting an ellipse for the melon contour. The ellipse-fitting process was achieved by minimizing a cost function related to the region homogeneity between the melon and the background using a gradient descent method. The weight of each melon was predicted using the feature size of the melon in a regression model trained to estimate the weight of a single melon.

Testing resulted in an individual melon weight accuracy of 16%. Analyses revealed this can be improved to 4% by accurate estimation of local ground sample distance.

## 3.2 Second method, research narrative and training

Reviewing the first system, it was possible to identify that there is room for improvement at the melon detection stage; using an object detector based on deep neural network could perform better. In addition, several models for improving the ellipse-fitting problem were suggested, such as the Chan–Vese model for active contours that is more efficient in term of time processing and ellipse-fitting. These suggestions were constituted as guidelines for the development of the second system. Since acquired images were too big a pre-processing activity was necessary. Hence, the system splits each image to grids of ten by ten sub-images. After each sub-image is separately processed, the algorithm recomposes the original image by using the initial coordinates of each sub image in the original image. This results with a complete image with all the detected melons anchor boxes coordinates. To avoid duplicate identification of melons which were located at the overlapping areas between two images a non-maximum-suppression (NMS) algorithm was applied. Current state of the art deep convolution neural networks (DCNN) are challenged to detect objects in the constraints we had - the images contain an unbalanced ratio between the background and the melons, and in addition, the size of the melons is small. RetinaNet DCNN was applied to overcome these limitations and provide the required accuracy. The network was modified and then fine-tuned using the transfer learning method. 4220 labeled melons taken from 4 different images were used for the transfer learning. This data set was enlarged by data augmentation performed during the fine tuning process of melon fruit detector. The detection process achieved an average precision score of 0.92 with a F1-score more than 0.9 in a variety of agriculture environments. For each detected melon, feature extraction was applied by using the Chan Vese active contour algorithm and PCA ellipse fitting method. By using the same regression that was used in the first system, the weight of each melon was estimated.

The system results for estimating the weight of a single melon measured by the mean absolute percentage error index achieved 16%. Analysis revealed that this can be decreased to 12% error with more accurate geometrical feature extraction. Overall yield estimation derived by summarizing the weights of all melons in the field and resulted in only 3% underestimation from total actual yield.

# 4. Estimating melon yield by machine vision processing

Submitted to: Precision Agriculture

Author's: A. Dashuta, A. Kalantar, Y. Edan, A. Dafna, A. Gur, I. Klapp

# Estimating open-field melon yield by machine-vision processing of UAV images

Artium Dashuta[1,3], Aharon Kalantar[1,2,†], Yael Edan[2], Asaf Dafna[4], Amit Gur[4,], Iftach Klapp[1]*

[1] Institute of Agricultural Engineering, Agricultural Research Organization (ARO), Volcani Center, Rishon-LeZion, Israel
[2] Department of Industrial Engineering & Management, Ben Gurion University of the Negev, Beer Sheva 8410501, Israel
[3] School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel
[4] Newe Ya'ar, ARO, Volcani Center, Ramat Ishay, Israel

† The first and second authors contributed equally.

*iftach@volcani.agri.gov.il

## Abstract

Monitoring plants for yield estimation in melon breeding is a highly labor-intensive task. An end-to-end algorithmic pipeline for yield estimation of melons from top-view UAV images of a melon field was developed. The pipeline has three main stages: melon recognition, geometric feature extraction, and individual melon weight estimation. The proposed regions of interest undergo a classification process using a pretrained convolutional neural network trained by transfer learning, resulting in an average precision of 0.82 and an F1 score of 0.85. For each region of interest classified as a melon, the melon's geometrical features are extracted by fitting an ellipse for the melon contour. The weight of each melon is predicted using the feature size of the melon in a regression model trained to estimate the weight of a single melon. The modified $R^2$ value of the regression model was 0.94. Testing resulted in an individual melon weight accuracy of 16%. Analyses revealed this can be improved to 4% by accurate estimation of local ground sample distance.

**Keywords:** Precision agriculture, Machine learning, CNN, Active contour, Melon, Yield estimation.

## Introduction

Accurate yield determination is important in many agricultural tasks, such as phenotyping (Busemeyer et al. 2013; Fukai and Fischer 2012), maturity determination for logistics planning (Bargoti and Underwood 2017b); Nuske et al. 2014; Stein et al. 2016), and disease/stress monitoring (Lu et al. 2017; Mohanty et al. 2016; Sladojevic et al. 2016).
Melons are delicate, fleshy fruits with more than 200 known species that vary in size, shape, texture, and color (Edan et al. 2000). They grow randomly scattered over the field, either individually or in small groups (Edan and Simon 1997), and are often hidden by leaves. Melon breeding requires a detailed account of information at the field level, such as accumulated yield and its distribution, and individual information such as melon size and location, which are essential for connecting yield to treatment/melon genetics. Currently, melon yield is estimated manually, requiring intensive and costly human resources. Hence, automation of the yield-estimation process could be beneficial (Gongal 2016; Kestur et al. 2019).
Different methods have been applied for yield estimation, including airborne hyperspectral imaging (Ye et al. 2007), smartphone imaging (Qian et al. 2018), Kinect camera imaging (Koirala et al. 2019a), and high-resolution multispectral satellite imaging (Rahman et al. 2018). Some work has been done on image-based yield measurements in fruit, but mostly for trees or in greenhouse-growing practices using a static camera (Chen et al. 2017; Kestur et al. 2019; Payne et al. 2013; Rahnemoonfar and Sheppard 2017; Yamamoto et al. 2014). Automated yield analysis for melons can be achieved by applying computer-vision and machine-learning methods to an unmanned aerial vehicle (UAV) view of a melon field. However, in most applications, yield measurements have focused on counting the number of objects (Bargoti and Underwood 2017b); Bresilla et al. 2019; Kestur et al. 2019; Lamb and Chuah 2018; Liang et al. 2018; Liu et al. 2013; Nuske et al. 2014; Sa et al. 2016) or estimating their volume (Chaivivatrakul et al. 2010; Herrero-Huerta et al. 2015; Moonrinta et al. 2010). To the best of our knowledge, very few studies have dealt with actual weight measurements of the individual fruit (Koirala et al. 2019b).
Yield estimation has garnered particular attention of late, and recent studies have shown impressive results in different scenarios, such as with apple (Bargoti and Underwood 2017b; Bresilla et al. 2019; Chen et al. 2017; Tao et al. 2018), orange (Chen et al. 2017; Tao et al. 2018) and mango orchards (Bargoti and Underwood 2017a; Koirala et al. 2019a; Liang et al. 2018). These studies used computer-vision methods such as convolutional neural networks (CNN) to obtain results with over 0.9 precision.
Machine-learning is extensively employed for fruit detection and recognition (Kamilaris and Prenafeta-Boldú 2018a; Kamilaris and Prenafeta-Boldú 2018b; Koirala et al. 2019b; Liakos et al. 2018) using different recognition methods, such as k-nearest neighbors, k-means with color information (Anisha et al. 2013; Qureshi et al. 2017),

and Faster Region-CNN (Bargoti and Underwood 2017a). Detection of almonds, mangoes, and apples resulted in precisions ranging from 0.7 to 0.9 (Bargoti and Underwood 2017a). Machine-learning techniques have also been used to estimate yield loss, such as a system that detects fruit on the orchard ground (Choi et al. 2013), and yield detection (Koirala et al. 2019b). Despite intensive research on object detection in agricultural environments, there are still many problems that hinder the implementation of agricultural applications (Gongal et al. 2015). The highly variable and unstructured outdoor environment, with changing illumination conditions and obstructions, along with the complex plant structure and variable product shape and size, make it hard to find a global solution for object detection in the complex agricultural environment (Kapach et al. 2012).

The present work proposes a framework for the detection and yield estimation of melons from color images acquired from a digital camera mounted on a UAV. It extends our previous studies, which proposed a detection algorithm with preliminary results (Dashuta and Klapp 2018), and provided initial analyses (Kalantar et al. 2019). Here, object detection under highly variable outdoor conditions, along with feature extraction and models for yield prediction, are detailed in the described algorithmic pipeline and results are presented. The focus in this work was on the prediction of individual melon weight, a feature not previously investigated in yield-detection work which is usually dealt with by counting the number of fruit at the field/tree level.

## Materials and methods

### Field trials

Melon plants for this experiment were grown in an open field in the summers of 2017 and 2018. Seedlings were transplanted in early April in Newe Ya’ar (32°43'05.4"N 35°10'47.7"E). The seedlings were spaced 0.5 m apart on raised beds covered with silver-colored plastic mulch (Ginnegar), 2 m between bed centers. The soil type was grumusol, and the plants were drip-irrigated and drip-fertilized to approximately 180 L $m^{-2}$ over the course of the growing season.

### Data acquisition

A UAV hovering above the site was used to acquire the data. To improve image detection, foliage coverage was reduced by stopping irrigation 1 week prior to acquisition. Acquisition was performed at midday on 17 Aug 2017, and when , 2018 at the fruit-ripening stage. RGB images were taken with a Sony ILCE-5000 camera mounted on a drone (Quad-Copter) hovering about 15 m above the field, with the camera facing vertically downward, of a 280 m x 15 m area. Before starting the image acquisition, 30 melons with different shapes, sizes, and colors were randomly tagged in the field and used for ground-truth data. These melons were analyzed in the laboratory after field image acquisition with a "Tomato Analyzer" tool (Gonzalo et al. 2009). The analysis provided a dataset of ground-truth geometrical features for each melon. The data acquired in 2018 were used to validate the statistical model by cross-validation (Hawkins 2004).

The images were saved in .jpg format with a resolution of 3064 × 5456 pixels. Overall, there were thousands of highly diverse melons. The variation in the dataset ensured the modeling of a robust and accurate detection algorithm that would generalize well under outdoor environmental conditions.

A typical UAV image is presented in figure 1. For the sake of training and validation of the extraction of geometrical features, 1056 melons were randomly chosen and manually tagged in the images. In addition, to tie mass to geometry, an additional 30 melons were randomly chosen and tagged with signboards, each with a different character (with different font size and thickness) which enabled their recognition in the UAV images. An example of such a tagged melon is shown in Fig. 1.

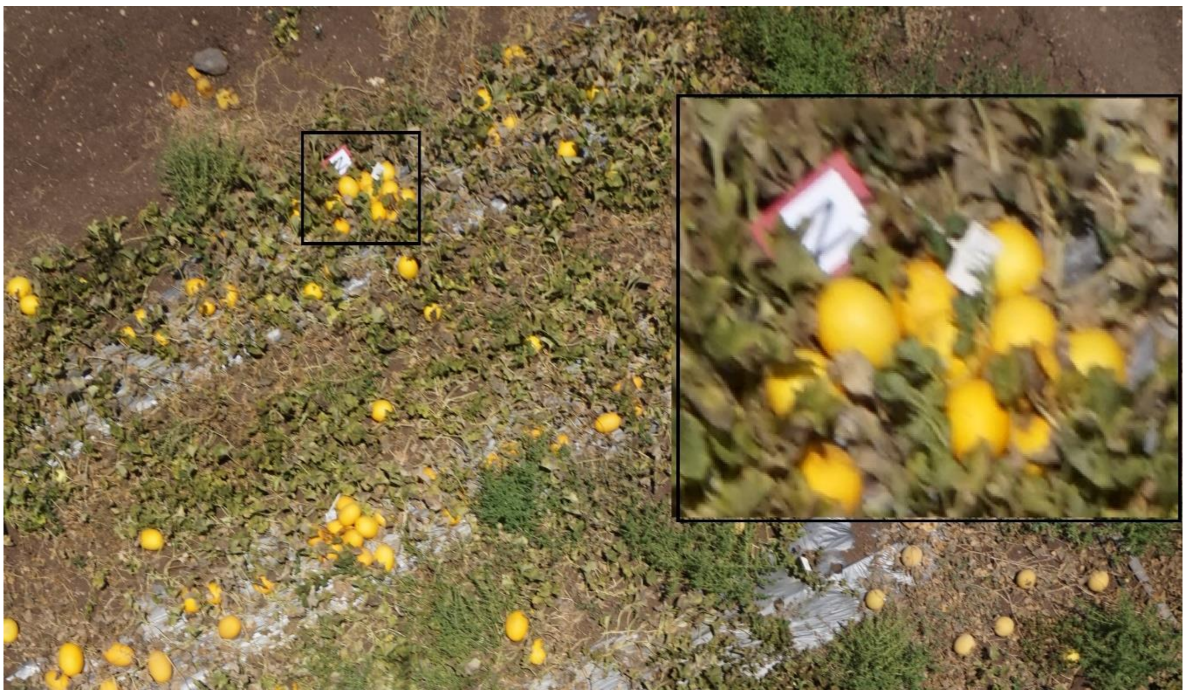


**Fig. 1** A typical UAV image. In the black box, example of a selected melon which was labeled with a sign with the letter "N" to allow its recognition in the UAV image. This melon was then weighed in the laboratory providing accurate ground truth data.

**Data augmentation**

To train the cascade object detector, 3031 negative training images (of background) and 1933 positive training images (of melons) were created. CNN training was based on 1383 of the positive, and 1349 of the negative training images. All melons were randomly selected. Augmentation techniques were used to increase robustness from every single training image; 31 additional, new images were produced by subjecting the training image to every single combination of the following transformations: rotate by $0^o$, $90^o$, $180^o$ or $270^o$, reflect about a vertical axis (or not), convolve with a Gaussian blur kernel (or not), add Gaussian noise (or not). This augmentation resulted in a training set with 44,256 positive images (with melon) and 43,168 negative images (without a melon). An example of data augmentation on a single positive training image is presented in Fig. 2.

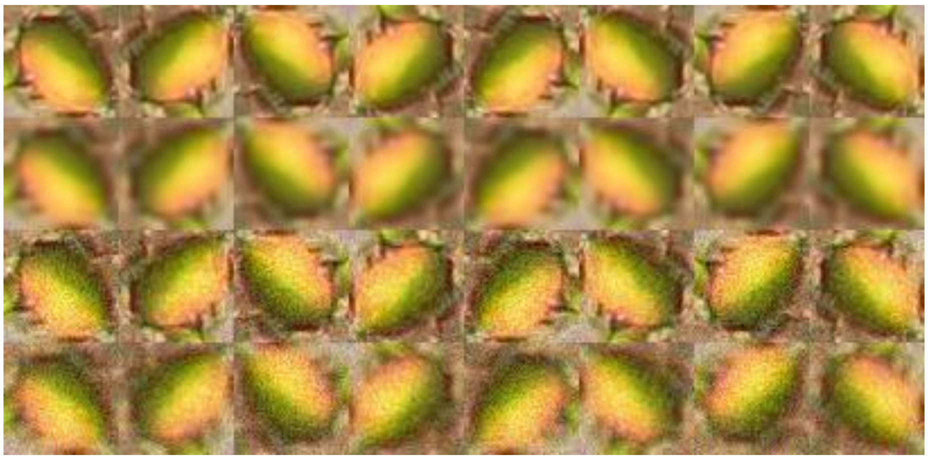


**Fig. 2** Each row contains the input image and its possible rotations and reflections. The first row has no noise and no blurring applied to it, the second row has only blurring applied, the third row has only noise applied, and the fourth row has both noise and blurring applied

**Metrics**

The following metrics were used to evaluate the developed algorithms.

<u>Goodness of geometrical feature extraction</u>

Detection performance was evaluated using recall and precision indicators together with the F1-score metric. A fruit detection is considered to be a true detection (true positive, TP) if the predicted and ground-truth bounding box had an intersection over union (IoU) greater than a fixed threshold (Bargoti and Underwood 2017b). False detection (also denoted as false positive, FP), refers to an algorithm's mistake in predicting background as a melon. A miss by the algorithm (denoted as false negative, FN), refers to its failure to detect a real melon. Precision indicates the fraction of the algorithm's predictions that are melons. Recall is the fraction of melons in the image that was detected by the algorithm. Increasing recall usually comes at the expense of precision. The harmonic mean so-called F1 score provides a balance between the two.

$$P = \frac{TP}{TP+FP} \;,\quad R = \frac{TP}{TP+FN} \;,\quad F1 = \frac{2 \cdot P \cdot R}{P+R} \tag{1}$$

Precision–recall curves depend upon the IoU threshold. In previous work, Zhu (2004) suggested using the average precision (AP) as a figure of merit for algorithm accuracy, such that:

$$AP = \sum_{n} \left[ R(n) - R(n-1) \right] \cdot P(n) \tag{2}$$

where *n* is the number of recognized melons. AP is a summation of the precision–recall curve and is calculated as the weighted mean of the precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight. The AP measurement has been used in the Pascal visual object classes (VOC) challenge, which is considered a benchmark in visual object category recognition and detection (Everingham et al. 2010).

To estimate the goodness of extraction of the melons' geometrical features and the ellipse border, labeled melons were tagged manually such that the label was the actual contour of the melon section. For each detection, the IoU was calculated between a binary mask in the labeled fruit borders to a binary mask in the ellipse border, such that it reflects the goodness of the extraction of the geometrical borders.

Regression quality

A regression model was developed to tie the melon's geometrical features to its weight. An adjusted $R^2$ statistic was used to avoid dependency on the number of terms used (Montgomery 1997):

$$R^2_{Adj} = 1 - \frac{SSE/(n-p)}{SST/(n-1)} \tag{3}$$

where SST is the total sum of squares and SSE is the sum of squares of residuals, the number of points is *n* = 30, and the number of parameters is *p* = 2.

## Algorithm

The proposed algorithm pipeline for automated yield prediction from RGB images of a melon field includes the following three sequential main stages (Fig. 3): 1) melon recognition, 2) feature extraction, 3) yield estimation. The input for the system are the RGB images of a melon field, and the output is a report that includes each melon's location and weight. Each stage is based on results from the previous stage.

The algorithm was implemented on a standard laptop equipped with an Intel Core i7-7500U, 64-bit double-core 2.9 GHz CPU, 8 GB memory running on the Microsoft

Windows 10 system. Efficient processing was emphasized along development to avoid the need for using a GPU.

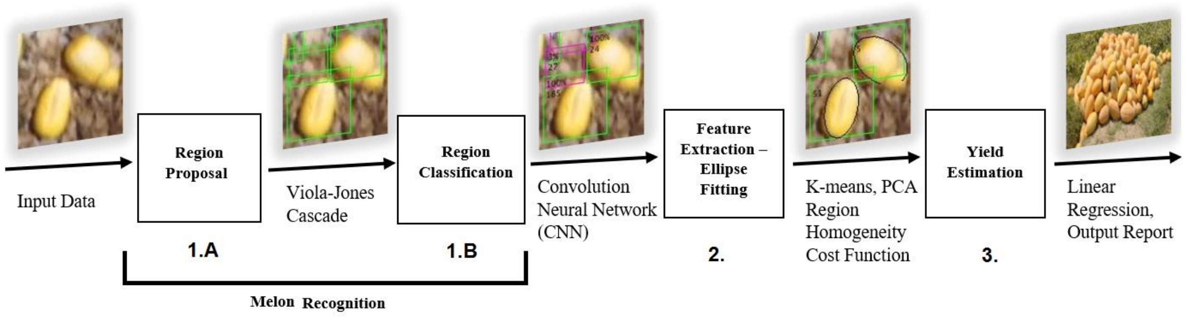

Input Data
Region Proposal
1.A
Viola-Jones Cascade
Region Classification
1.B
Melon Recognition
Convolution Neural Network (CNN)
Feature Extraction – Ellipse Fitting
2.
K-means, PCA Region Homogeneity Cost Function
Yield Estimation
3.
Linear Regression, Output Report

**Fig. 3** Algorithm pipeline for automated yield tracking

## Melon recognition

The first step of the algorithm dealt with melon recognition. Since the melons cover only a small portion of the field, they are a relatively rare target among other possible objects in the UAV image. To reduce unneeded computational effort, the recognition process for a melon target was split into two substages: region proposal (1A), and region classification (1B).

### Candidate region proposal

“Candidate” regions suspected as "melon" regions of interest (ROIs) were proposed by running a sliding window over the entire input image. This requires an exhaustive search and therefore must be implemented with low time complexity. The step was also designed to have a high recall, at the expense of low precision. The Viola–Jones face detector (Viola and Jones 2001) was chosen to realize this coarse preliminary detection. The Viola–Jones detector includes a cascade object detector composed of an ensemble of a number of weak classifiers. The cascade object detector is arranged in stages with increasing complexity. The role of each stage is to decide whether the current window is definitely not an object. If a stage decides that the current window is not an object, the remaining stages are not evaluated, so only true object windows trigger the entire cascade of stages. This mechanism ensures low computational complexity. The feature type selected for the detector in the Matlab implementation was a histogram of oriented gradients (Dalal and Triggs 2005), which was found suitable for capturing the elliptic nature of the melon.

## Region classification

Having suspected melon candidates, the algorithm classifies the suspected regions using a CNN schema Error! Reference source not found.. This was performed as a ‘transfer-learning’ methodology (Li et al. 2016) using pretrained Matlab implementation on the CIFAR-10 dataset (Krizhevsky 2009), a 10-category 32 x 32 color image dataset. The final fully connected layer, which is essentially an image classifier based on a 64-dimensional feature vector produced by the hidden layers, was changed from a layer with output size of 10 (the original dataset had 10 classes) to one with an output size of 2 labels—melon and background. Replacing the final layer also implies that its weights are randomly initialized  while the remaining previous layers keep their weights. In accordance with the transfer-learning methodology, the learning rate for the weights in the final layer is 20 times higher than the learning rate of the previous layers. This implies that most of the learning is done at the final layer, while the previous layers adjust only slightly. The structure of the CNN net is presented in Table 1. Each ROI classified as containing a melon was further processed to derive melon features in the next stage.

**Table 3** The architecture of the convolutional neural network used as the region classification model

| Layer | Layer type | Properties |
|---|---|---|
| 1 | Input image layer | InputSize = [32,32,3], Normalization = ’zero-center’ |
| 2 | Convolution 2D layer | FilterSize = [5,5], NumChannels = 3, NumFilters = 32, Stride = [1,1], PaddingSize = [2,2,2,2] |
| 3 | Rectified linear unit layer | |
| 4 | Max pooling 2D layer | PoolSize = [3,3], Stride = [2,2], PaddingSize = [0,0,0,0] |
| 5 | Convolution 2D layer | FilterSize = [5,5], NumChannels = 32, NumFilters = 32, Stride = [1,1], PaddingSize = [2,2,2,2] |
| 6 | Rectified linear unit layer | |
| 7 | Max pooling 2D layer | PoolSize = [3,3], Stride = [2,2], PaddingSize = [0,0,0,0] |
| 8 | Convolution 2D layer | FilterSize = [5,5], NumChannels = 32, NumFilters = 64, Stride = [1,1], PaddingSize = [2,2,2,2] |
| 9 | Rectified linear unit layer | |
| 10 | Max pooling 2D layer | PoolSize = [3,3], Stride = [2,2], PaddingSize = [0,0,0,0] |
| 11 | Fully connected layer | InputSize = 576, OutputSize = 64 |
| 12 | Rectified linear unit layer | |
| 13 | Fully connected layer | InputSize = 64, OutputSize = 2 |
| 14 | Softmax layer | |
| 15 | Classification output layer | OutputSize = 2, LossFunction = ‘cross-entropy’ |

**Geometrical feature extraction**

The outcome of the first stage is the set of ROIs containing a single melon. Using the pre-knowledge that the shape of each individual melon can be well approximated by a spheroid (Heinzen et al. 1998), the top view of the melon contour should be recognized as an ellipse. The aim is to fit an ellipse model to the melon’s contour, from which the melons’ geometrical features are derived.

Once the ellipse with the best fit for each detected melon is derived, the feature extraction is trivial. The ellipse partitions the patch into two regions: inner region (inside the ellipse) and outer region (in the proposed candidate region, outside the ellipse), using the following decision rule: pixels in the inner region are classified as ‘melon’ and pixels in the outer region are classified as ‘background’. Every possible ellipse can be parametrized by five parameters: semi-major axis ( $c$ ), semi-minor axis

($a$), centroid x coordinate ($x_0$), centroid y coordinate ($y_0$), and angle of tilt ($\theta$). Finally, the contour of every melon in the field can be approximated as an ellipse:

$$\frac{\left[(x-x_0)\cos(\theta)-(y-y_0)\sin(\theta)\right]^2}{a^2}+\frac{\left[(x-x_0)\sin(\theta)+(y-y_0)\cos(\theta)\right]^2}{c^2}=1 \quad (4)$$

The five parameters defining the ellipse are a point in $\mathbb{R}^5$ such that, every point $\mathbf{x}=(a,c,x_0,y_0,\theta)\in\mathbb{R}^5$ in the parameter space corresponds to a single ellipse (single solution) in the given ROI. To derive these parameters, the following algorithm is proposed; first, an initial ellipse that fits the contour is derived, then parametrization is conducted, followed by solution optimization using a cost function minimization problem (Dashuta and Klapp 2018).

## Finding the initial ellipse

Finding the initial ellipse for each ROI included the following three main stages:

1. The input ROI is segmented using the k-means clustering algorithm Error! Reference source not found. with two centroids for melon and background. The set of data points that are clustered by the k-means algorithm are the RGB pixel values of all pixels in the image. The k-means algorithm is set to repeat three times (for different initial centroids). After clustering, a heuristic rule is applied to decide which cluster corresponds to the melon and which cluster corresponds to the background by computing the average distance between the spatial coordinates of pixels in each cluster to the center of the image: $(x,y)=(0.5W,0.5H)$, where $W$ is the input image's width and $H$ is the input image's height. Assuming the melon is located roughly at the center of the input image, the cluster with the lower average distance to the center of the image is chosen as the melon cluster. Only the largest connected (melon) component present in the binary mask is left. Figure 4a shows eight different arbitrary melons. The resultant binary mask is presented in Fig 4b.
2. The obtained melon binary mask is used to approximate the blob in the center, corresponding to the melon, by an ellipse using principal component analysis (PCA) (Paplinski and Wijewickrema 2004). To ensure computational efficiency, the binary mask is first decimated. This is achieved by constructing a binary decimation mask where in each row, every $d^{th}$ pixel is set, and in each column, every $d^{th}$ pixel is set, where the decimation factor $d$ is the smallest co-prime with $H$ or $W$. The segmentation mask and the decimation mask are logically AND-ed to receive the decimated segmentation mask (Fig. 4d). From this point, only the spatial

coordinates of the pixels that are set in the decimated segmentation mask are used as the sample set.

3. To determine the initial ellipse, the mean of the sample set $\boldsymbol{\mu} = \left(\mu_x, \mu_y\right)$ is computed. Then, PCA (Tseng and Yun 2009) is performed on a mean-centered sample set. The PCA algorithm returns a set of two normalized eigenvectors (arranged as two columns in a matrix) and their two corresponding eigenvalues:

$$V = \begin{pmatrix} v_x^1 & v_x^2 \\ v_y^1 & v_x^2 \end{pmatrix}, \boldsymbol{\lambda} = \left(\lambda_1, \lambda_2\right), \lambda_1 \geq \lambda_2 = 1 \tag{5}$$

The ellipse parameters are then estimated according to the following formulae: $a = 2\sqrt{\lambda_1}$ $b = 2\sqrt{\lambda_2}$ $x_0 = \mu_x$ $y_0 = \mu_y$ $\theta = \arctan\left(v_y^1 / v_x^1\right)$ . Examples for the resultant initial ellipses are presented in Fig. 4d.

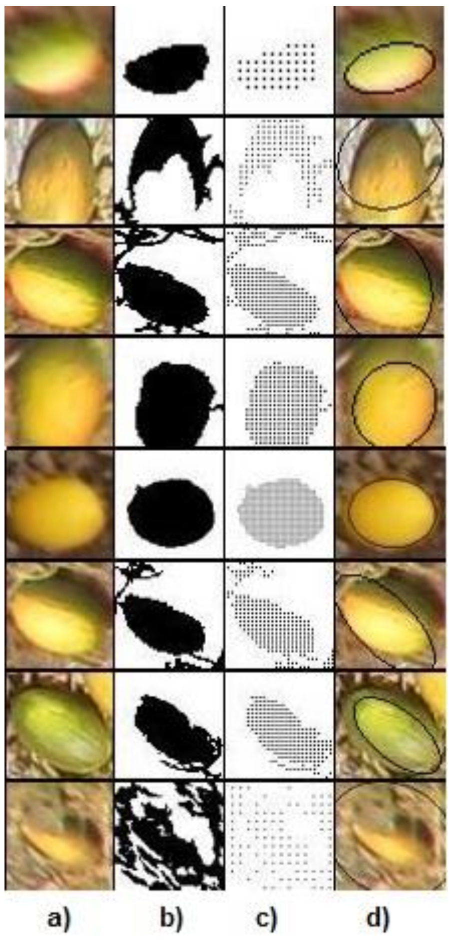


**Fig. 4** Examples of the initial ellipse generation algorithm. From left to right: (a) input image; (b) segmentation mask from k-means (largest connected component only), where the black pixels belong to the set identified as ‘melon’; (c) decimated segmentation mask; (d) output initial ellipse imposed on the input image

Finding an accurate ellipse parameter

To determine the optimal ellipse parameter that fits the melon's border, the 5-parameter active contour cost function was minimized. Assuming the melon is distinct from the background, the minimization criteria rely on the inner (melon) and outer (background) color. If the melon and the background each have their own characteristic colors when the ellipse exactly overlaps the melon, high homogeneity is induced in both the inner and outer regions. A change in ellipse size or orientation results in increased nonhomogeneity. Therefore, the best possible fitted ellipse is indeed a minimum, or at least a local minimum of a cost function which minimizes nonhomogeneity.

Applying a stopping criterion at a point where both inner quantities and outer quantities are minimized shares similarity with the Chan–Vese (Chan and Vese 2001) segmentation algorithm. The ellipse solution benefits from the direct estimation of ellipsoid parameters and results in a very high correlation with the melon's mass. In addition, constraining the solution to an ellipse bypasses possible errors in estimating the melons' contours, due to partial occlusion of the melons by leaves.

The following measure, based on sample covariance matrices, was used to quantify homogeneity of the image region: $C_{IN}, C_{OUT}$ are the sample covariance matrices computed from the sets of the pixels' RGB-n. Note that homogeneous regions have low variance values, so minimizing the values in the covariance matrices will maximize the homogeneity.

In color RGB images, each pixel holds three values, resulting in a 3 x 3 dimensional covariance matrix. The standardized generalized variance (Diago et al. 2014), defined as follows, was used as the cost function: $SGV = \sqrt[d]{\det(C)}$ where $C$ is a sample covariance matrix and $d$ is its dimensionality. The following quantities are also defined: $N_{IN}, N_{OUT}$, which are the number of pixels in the inner and outer regions, respectively. The suggested cost function is defined as:

$$f\left(\mathbf{x} = (a, b, x_0, y_0, \theta)\right) = (1-\alpha) N_{IN} \sqrt[3]{\det(C_{IN})} + \alpha N_{OUT} \sqrt[3]{\det(C_{OUT})} \tag{6}$$

The coefficient $\alpha \in (0,1)$ is a tunable parameter. Notice that the two terms of the cost function have units of "energy", in [gray level$^2$] units. For $0.5 < \alpha < 1$, the outer term is more dominant and such a cost function will tend to prefer solutions where the outer region has lower energy at the expense of more energy in the inner region. This essentially translates to a tendency toward expansion of the ellipse outward to decrease the outer energy by reducing the sample size. For $0 < \alpha < 0.5$, the inner term is more dominant and such a cost function prefers solutions where the inner region has lower energy at the expense of more energy in the outer region. This essentially translates to a tendency toward contraction of the ellipse inward to decrease the inner energy by reducing the sample size.

An $\alpha = 0.6$ was selected to ensure that the ellipse covers all melon parts. The minimization process was implemeted using the gradient descent algorithm (Tseng and Yun 2009), which was selected since it was both easy to implement and produced good results. The algorithm is outlined here:

```
$\mathbf{x}_0 = \mathbf{x}_{init}$
    $for\ n = 1, 2, ...$
        $\mathbf{x}_n = \mathbf{x}_{n-1} + \gamma \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{n-1}}$        (7)
    $end$
```

where $\mathbf{x}_{init}$ are the initial derived ellipse parameters, $\mathbf{x}_n = (a_n, b_n, x_{0n}, y_{0n}, \theta_n)$ is the current solution at iteration $n$, $f$ is the cost function, $\left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{n-1}}$ is the 5-dimensional gradient of $f$ evaluated at the point $\mathbf{x}_{n-1}$, and $\gamma$ is a tunable parameter known as “step size” or “learning rate”. The gradient is computed numerically by first evaluating $f$ at $\mathbf{x}_{n-1}$ and then perturbing every component of $\mathbf{x}_{n-1}$ separately, evaluating $f$ again and finally taking the difference. The algorithm is terminated when it reaches a maximum number of iterations (200) or if the maximal error magnitude among the 5 gradient components is lower than a threshold (0.01). At the end of this stage, the location of each melon in the field was derived, and the size of the melon in the image was extracted and is presented in centimeters.

## Yield estimation model

To estimate the yield, a model that ties the melon's geometry to its weight was developed. A spheroid model was applied using a 3D shape with same-sized width and depth axes (Fig. 5). The parameters derived from the fitted ellipse in the 2D image, in particular, the sizes of the minor and major axes, were correlated to the semi-height and semi-width of the melon based on which the melon weight was predicted.

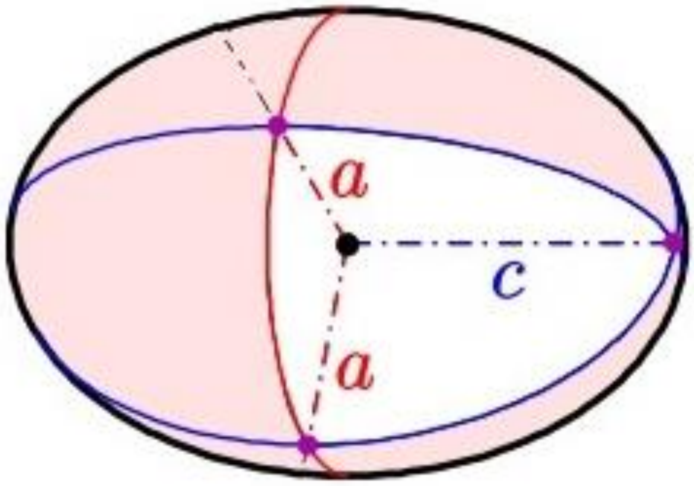


**Fig. 5** Ellipsoid model (Wikipedia 2019)

Regression model between melon's geometry and melon's weight

The regression model was built from 30 random individual melons which were both imaged by the UAV and measured for their weight and geometry in the laboratory. To estimate melons axes, each melon was cut in half along the long axes, where the maximum length and width were assumed as the measures of the ellipsoid axes. The best derived regression was:

$$\mathrm{W} = 0.1096653 + 0.003397929 \cdot \mathrm{c} \cdot a^2 \tag{8}$$

where *W* is the melon weight, *a* is the ellipse semi-width and *c* is the ellipse semi-height (semi-long axis). The ellipsoid model for a melon is presented in Fig. 5. The resulting weight is in linear proportion to the volumatic character, equal to $V = \mathrm{c} \cdot a^2$

Regression model

The regression model (Eq. 4, Fig. 6) resulted in a $R^2_{Adj}$ of 0.94 (for n = 30 melons).

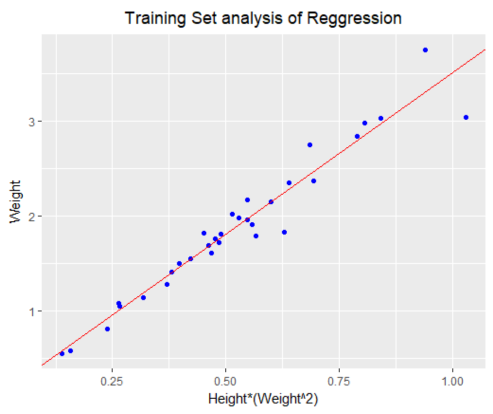


**Fig. 6** Training set analysis of the regression model

Converting imagery information to ellipsoid parameters

The regression in Eq. 4 relies on volumetric features given in metric units. Minimization of the active contour resulted in ellipse parameters given in pixels and was translated to centimeters using the ground sample distance (GSD) measurement; GSD is a measure of the spatial resolution of an image and is defined as the linear dimension of a single pixel's projection on the ground. The naïve first-order estimation for the GSD includes three measurements:

$$GSD = \frac{h \cdot \Delta p}{f} \tag{9}$$

where $h$ is the altitude at which the image was acquired, $\Delta p$ is the sensor's pixel size, and $f$ is the focal length setting at which the image was acquired. An accurate estimation of the GSD should consider the dependence of perspective and magnitude on the distance of each melon from the image center: the further a melon is from the image center, the smaller it appears in the image. In addition, the distance (*h*) depends on both the UAV height variation—which was not measured accurately due to the instability of the drone while hovering, and ground level, which is not a constant. Due to these problems, a-priori field calibrations using a signboard of known dimensions placed near each melon was used to derive a calibration ratio which was applied to the features of the marked melon in the image located near the signboards. For all other melons, the average GSD value was applied.
Finally, the overall end-to-end yield estimation was performed individually for each of the individual melons in the field using the following process (Fig. 7): images were captured by a UAV (Fig. 7a), and postprocessed to obtain an ellipse model for each individual melon (Fig. 7b); using the GSD, the ellipse parameters were converted into

metric parameters of an ellipsoid (Fig. 7c); using the regression model (Fig. 7d), each individual melon's weight was estimated (Fig. 7e).

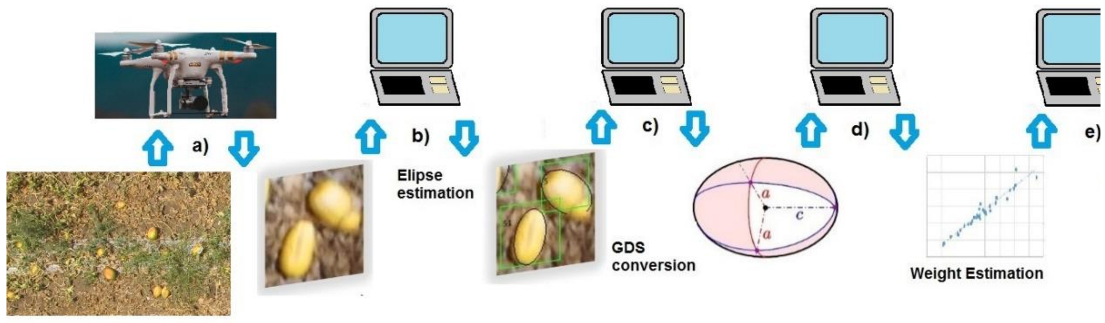


**Fig. 7** End-to-end yield estimation

## Results and discussion

Performance analysis was conducted for each stage: geometrical feature extraction, weight regression model, and yield estimation calculated as melon weight.

### Validation of the goodness of geometrical feature extraction

Precision–recall-based analysis was conducted for two different IoU thresholds. The first was 0.5, as used in Pascal VOC challenges (Everingham et al. 2010), which resulted in relatively small fruit size with respect to the image resolution; errors in detecting melons were obtained, causing them to be registered as false negatives. To reduce the false negative misclassification, a smaller threshold of 0.2 was applied, which equates to a 58% overlap along each axis of the object; this has been considered sufficient for fruit-mapping applications (Krizhevsky et al. 2012). Lowering the IoU reflects inherent inaccuracies stemming from the manual labeling errors in drawing the borders of the training and validation sets.
Attached melons may cause difficulty in melon segmentation. In several cases, a single ellipse captured two attached melons. Similarly, when a single ellipse catches part of two attached melons, only the melon with the larger overlap is counted.
The detection algorithms resulted in an average precision of 0.62 and F1 score of 0.71 for an IoU of 0.5. Reducing the threshold to 0.2 yielded improved performance, with average precision of 0.82 and F1 score of 0.85. Using an IoU of 0.2 improved both the recall rate and precision (Fig. 8), with 87% of the tagged fruit used for the detected calibration.

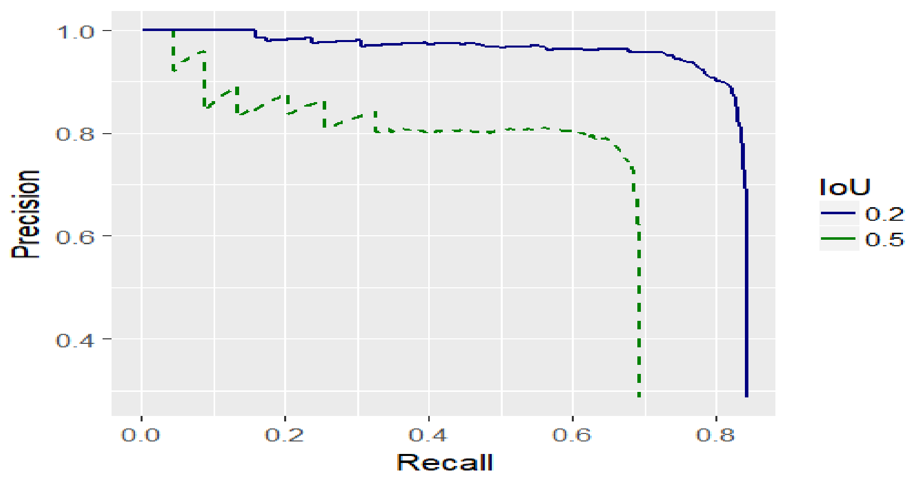


**Fig. 8** Average precision–recall curve from the experimental results (Kalantar et al., 2019)

Figure 9 shows a typical UAV image after processing to identify melons and extracting the geometrical features of the melon's envelope. The positive identifications are marked by green rectangular frames, which in most cases were true. The ellipse border associated with the resultant geometrical features was printed in black. A closer look at some typical results is given in Fig. 10. Figure 10a presents a correct identification of melon envelopes when the melons are well separated. The candidate proposal (subsection 1A in the algorithm section) may be sensitive in cases where two melons are attached to each other. When two attached melons are recognized as a single melon, only one frame and one ellipse are proposed, as shown in Fig. 10b. The tendency for this type of error requires a parallel attachment between the melons. In another activity not presented here, a pile of melons was shown to be divisible into sub-melons when considering the expected size of a melon. In cases where two attached melons arrange in a "row-like" order, the candidate proposal recognizes two separate melons. Since the estimation of the geometrical feature relies on the assumption of an elliptical shape, this resulted in correct border identification even under partial occlusion of one melon by the other (Fig. 10c). This was also true when partial occlusion originated from leaves or weeds (Fig. 10d).

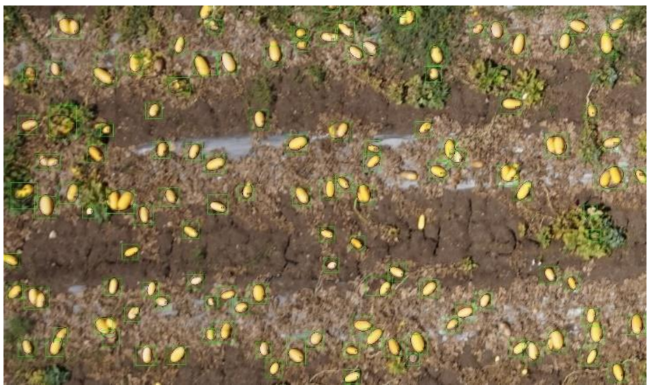


**Fig. 9** A selected region in an output image. Stage 1: melon recognition – green bounding boxes represent the ROI classified as a melon. Stage 2: feature extraction – a black ellipse-shaped contour drawn around every single ROI. From this example, the melons covered by flora are seen to be difficult to recognize, as are melons that are placed very close together and overlap, which are recognized as one melon. These challenging cases affect the results for feature extraction leading to missing fit of the ellipse

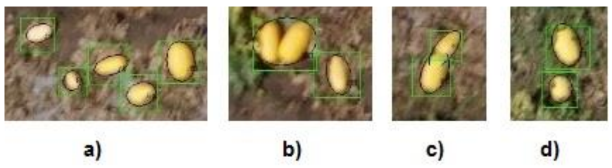


**Fig. 10** A close look at the atypical results of the selected region in an output image. (a) Correct ellipse fitting. (b–d) Overcoming partial occlusion due to the elliptical shape assumption

## Validation of regression quality

The validation results (Fig. 11) reveal satisfactory agreement between this setup and the proposed regression line (Eq. 8). The average weight estimation resulted in a 2% error of overweight (Fig. 12 presents the error histogram).

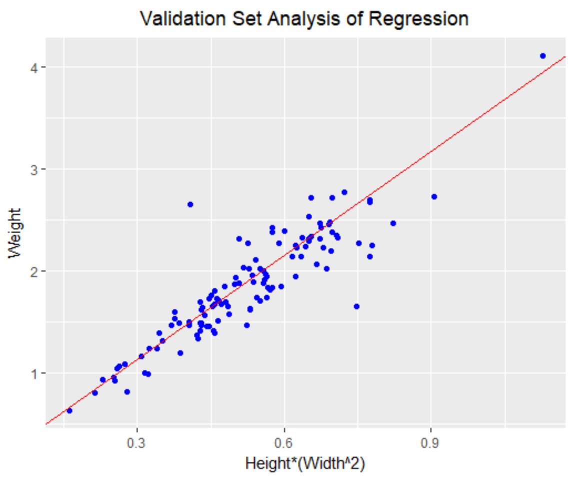


**Fig. 11** Validation set analysis of the regression model

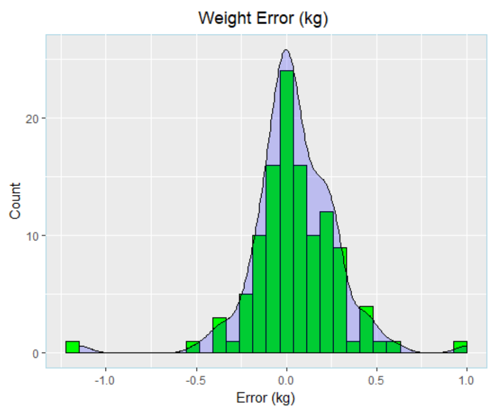


**Fig. 12** Histogram of error in weight estimation of the validation set. The kernel density estimation is shown as a continuous line

**End-to-end individual melon yield estimation – preliminary results**

A few of the 30 signs aimed for ground truth measurements were flipped by the wind during the experiments, and due to the random sampling, a few others were not found. A review of all of the images collected in the field experiments using the drone identified only 15 melons. Since the yield-estimation stage relies heavily on the melon-detection stage, 13 of these 15 melons were selected by the algorithm. From these 13 melons, an ellipse was adjusted during the feature-extraction process. Four melons were visually identified as a good match between the melon counter and the ellipse. The GSD metric was used to convert the pixel units to centimeter units with a derived ratio of 0.4 centimeters per pixel. Using this ratio, the actual yield was forecasted. A detailed account of the results for each melon is given in Table 2, where it can be seen that most of the weight errors originated from errors in ellipse fitting. In six cases—melons 5, 6, 9, 10, 11 and 12—the melon contours were underestimated; the weight estimation in these cases also tended to be deficicient. In cases where the error originated from a bad GSD estimation, it was corrected by evaluating the GSD locally. Finally, in some cases, the fitting error was damaged by occlusions caused by other objects in the field, such as foliage. The overall result was overestimated by 16% more than the actual yield.

**Table 2** Weight estimation results of individual melons

| # | Semi major axis [cm] | Semi minor axis [cm] | Estimated weight [kg] | Actual weight [kg] | Delta [kg] | Ellipse fitted | Gap reasons |
|---|---|---|---|---|---|---|---|
| **1** | 9.715 | 6.513 | 1.510 | 1.280 | 0.230 | good | |
| **2** | 5.959 | 3.417 | 0.346 | 1.688 | -1.342 | good | Occlusions, GSD |
| **3** | 6.804 | 5.145 | 0.722 | 0.578 | 0.144 | good | |
| **4** | 12.439 | 7.836 | 2.705 | 1.956 | 0.749 | good | GSD |
| **5** | 13.072 | 7.522 | 2.623 | 2.742 | -0.119 | underfit | |
| **6** | 8.811 | 5.951 | 1.170 | 1.822 | -0.652 | underfit | Viola–Jones BBOX not good |

|  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| **7** | 13.584 | 8.499 | 3.444 | 0.812 | 2.632 | bad fit |  |
| **8** | 6.412 | 5.509 | 0.771 | 3.040 | -2.269 | bad fit |  |
| **9** | 6.278 | 5.079 | 0.660 | 1.257 | -0.597 | underfit |  |
| **10** | 8.261 | 7.227 | 1.576 | 1.550 | 0.026 | underfit |  |
| **11** | 7.879 | 4.939 | 0.763 | 1.142 | -0.379 | underfit |  |
| **12** | 10.075 | 5.226 | 1.045 | 3.750 | -2.705 | underfit | occlusion |
| **13** | 10.516 | 8.001 | 2.397 | 1.974 | 0.423 | bad fit | contour includes another melon |

In cases where the gap between the estimated weight and actual weight could be explained by inaccurate GSD, variation might originate from: a variable distance between the drone and the ground due to variation in field flatness and/or a variation in flight height,  vibrations of the UAV, and wind causing movements in the drone while hovering. For the sake of the local GSD estimation, the metric size of the pixels near each melon was estimated from the targets that were located next to the labeled melons (Fig. 1), using the ratio between their actual size and pixel size; an adaptive ratio for each melon was found. To isolate the influence of possible error in geometrical feature estimation on the weight estimation, the five melons with the best-fitted ellipses were selected. Results revealed that the error was limited to about 4% of the total actual weight with an average deviation of 50 g per melon. This implies that finding the correct GSD is essential to making an accurate yield estimation; such a space-variant calibration can rely on a grid of simple man-made rectangular plastic targets, which is relatively easy to implement.

## Summary and future works

An algorithmic pipeline for yield estimation of melons from top-view UAV images of a melon field was developed. The pipeline includes three main stages: melon recognition, geometric feature extraction, and individual melon yield estimation. ROIs that might contain a melon are recognized using the Viola–Jones algorithm. The proposed regions undergo a classification process using a pretrained CNN trained by transfer learning, resulting in an average precision of 0.82 and F1 score of 0.85. For each ROI classified as a melon, the melon's geometrical features were extracted by fitting an ellipse for the melon contour. The ellipse-fitting process was achieved by minimizing a cost function related to the region homogeneity between the melon and the background using a gradient descent method. The weight of each melon was predicted using the feature size of the melon in a regression model trained to estimate the weight of a single melon.

End-to-end testing resulted in an individual melon weight accuracy of 16%, which can be improved to 4% by accurate estimation of local GSD. The overall yield was estimated by summing the weights of all melons in the field.  Ongoing work is focused on improving recognition by using improved object-detection techniques, based on CNNs. In addition, several models for improving the ellipse-fitting problem, such as the Chan–Vese model for active contours (Chan and Vese 2001), are under examination. An automated yield analysis system could minimize the labor required for this task.

## Acknowledgments

This work was partially supported by BARD program number IS-4911-16 and by the Rabbi W. Gunther Plaut Chair in Manufacturing Engineering at Ben-Gurion University of the Negev.

## References

Anisha, S., Divya, G., & Shanu, S. (2013). A survey of computer vision methods for counting fruits and yield prediction. *International Journal of Computer Science Engineering*, *2*(6), 346–350.

Bargoti, S., & Underwood, J. (2017a). *Deep fruit detection in orchards*. arXiv:1610.03677v2 [cs.RO]. Accessed 18 September 2017.

Bargoti, S., & Underwood, J. (2017b). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, *34*(6), 1039–1060.

Bresilla, K., Perulli, G. D., Boini, A., Morandi, B., Grappadelli, L. C., & Manfrini, L. (2019). Single-shot convolution neural networks for real-time fruit detection within the tree. *Frontiers in Plant Science*, https://doi.org/10.3389/fpls.2019.00611

Busemeyer, L., Mentrup, D., Möller, K., Wunder, E., Alheit, K., Hahn, V., et al. (2013). BreedVision—a multi-sensor platform for non-destructive field-based phenotyping in plant breeding. *Sensors*, *13*(3), 2830–2847.

Chaivivatrakul, S., Moonrinta, J., & Dailey, M. N. (2010). Towards automated crop yield estimation-detection and 3D reconstruction of pineapples in video sequences. In *VISAPP 2010 - Proceedings of the Fifth International Conference on Computer Vision Theory and Applications* (vol. 1, pp. 180–183). INSTICC Press.

Chan, T. F., & Vese, L. A. (2001). Active contours without edges. *IEEE Transactions on Image Processing*, *10*(2), 266–277.

Chen, S. W., Shivakumar, S. S., Dcunha, S., Das, J., Okon, E., Qu, C., et al. (2017). Counting apples and oranges with deep learning: a data-driven approach. *IEEE Robotics and Automation Letters*, *2*(2), 781–788.

Choi, D., Suk-Lee, W., & Ehsani, R. (2013). Detecting and counting citrus fruit on the ground using machine vision. In *2013 ASABE Annual International Meeting* (Paper no. 131591603). St. Joseph, MI: American Society of Agricultural and Biological Engineers.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (vol. 1, pp. 886–893). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Dashuta, A., & Klapp, I. (2018). Melon recognition in UAV images to estimate yield of a breeding process. Light, Energy and the Environment (E2, FTS, HISE, SOLAR, SSL), *OSA Technical Digest* (Optical Society of America), paper ET4A.2. https://www.osapublishing.org/abstract.cfm?URI=EE-2018-ET4A.2

Diago, M. P., Sanz-Garcia, A., Millan, B., Blasco, J., & Tardaguila, J. (2014). Assessment of flower number per inflorescence in grapevine by image analysis under field conditions. *Journal of the Science of Food and Agriculture*, *94*(10), 1981–1987.

Edan, Y., Rogozin, D., Flash, T., & Mile, G. E. (2000). Robotic melon harvesting. *IEEE Transactions on Robotics and Automation*, *16* ,831–835.

Edan, Y., & Simon, J. E. (1997). Spatial and temporal distribution of ripe Cantaloupe fruit. *HortScience*, *32*(7), 1178–1181.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Fukai, S., & Fischer, K. S. (2012). Field phenotyping strategies and breeding for adaptation of rice to drought. *Frontiers in Physiology*, *3*, 282.

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., & Lewis, K. (2015). Sensors and systems for fruit detection and localization: a review. *Computers and Electronics in Agriculture*, *116*, 8–19.

Gongal, A., Silwal, A., Amatya, S., Karkee, M., Zhang, Q., & Lewis, K. (2016). Apple crop-load estimation with over-the-row machine vision system. *Computers and Electronics in Agriculture*, *120*, 26–35.

Gonzalo, M. J., Brewer, M., Anderson, C., Sullivan, D., Gray, S., & Knaap, E. (2009). Tomato fruit shape analysis using morphometric and morphology attributes implemented in tomato analyzer software program. *Journal of the American Society for Horticultural Science*, *134*(1), 77–87.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: a k-means clustering algorithm. *Applied Statistics*, *28*, 100–108.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of Chemical Information and Modeling*, *44*(1), 1–12.

Heinzen, A., Shimmel, C., Groppe, R., & Davidson, E. A. (1998). *Apparatus for removing rind from spheroidal fruits and vegetables*. U.S. Patent 5,806,414, 15 Sep 1998.

Herrero-Huerta, M., González-Aguilera, D., Rodriguez-Gonzalvez, P., & Hernández-López, D. (2015). Vineyard yield estimation by automatic 3D bunch modelling in field conditions. *Computers and Electronics in Agriculture*, *110*, 17–26.

Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A., & Klapp, I. (2019). Estimating melon yield for breeding processes by machine-vision processing of UAV images. In *The 12th European Conference on Precision Agriculture*, https://doi.org/10.3920/978-90-8686-888-9

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018a). A review of the use of convolutional neural networks in agriculture. *The Journal of Agricultural Science*, *156*(3), 312–322.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018b). Deep learning in agriculture: a survey. *Computers and Electronics in Agriculture*, *147*, 70–90.

Kapach, K., Barnea, E., Mairon, R., Edan, Y., & Ben-Shahar, O. (2012). Computer vision for fruit harvesting robots – state of the art and challenges ahead. *International Journal of Computational Vision and Robotics*, *3*(1/2), 4–34

Kestur, R., Meduri, A., & Narasipura, O. (2019). MangoNet: a deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard. *Engineering Applications of Artificial Intelligence*, *77*, 59–69.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019a). Deep learning for real-time fruit detection and orchard fruit load estimation: benchmarking of '*MangoYOLO*'. *Precision Agriculture*, https://doi.org/10.1007/s11119-019-09642-0

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019b). Deep learning—method overview and review of use for fruit detection and yield estimation. *Computers and Electronics in Agriculture*, *162*, 219–234.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. Accessed 5 May 2019.

Krizhevsky, A., Sutskever, I., & Hinton G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25* (pp. 1097–1105).

Lamb, N., & Chuah, M. C. (2018). A strawberry detection system using convolutional neural networks. In *2018 IEEE International Conference on Big Data (Big Data)* (pp. 2515–2520). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Li, Z., Song, Y., Mcloughlin, I., & Dai, L. (2016). Compact convolutional neural network transfer learning for small-scale image classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 2737–2741). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Liakos, K., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: a review. *Sensors*, *18*(8), 2674.

Liang, Q., Zhu, W., Long, J., Wang, Y., Sun, W., & Wu, W. (2018). A real-time detection framework for on-tree mango based on SSD network. In Z. Chen, A. Mendes, Y. Yan, S. Chen (Eds.), *Intelligent robotics and applications. ICIRA 2018. Lecture notes in computer science* (vol 10985, pp. 423–436). Cham: Springer.

Liu, S., Marden, S., & Whitty, M. (2013). Towards automated yield estimation in viticulture. In *Proceedings of the Australasian Conference on Robotics and Automation* (vol. 24, pp. 2–6). Australian Robotics and Automation Association.

Lu, Y., Yi, S., Zeng, N., Liu, Y., & Zhang, Y. (2017). Identification of rice diseases using deep convolutional neural networks. *Neurocomputing*, *267*, 378–384.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*, 1419.

Montgomery, D.C. (1997). *Design and analysis of experiments* (fourth ed.) (chapter 13). Hoboken, NJ: John Wiley & Sons.

Moonrinta, J., Chaivivatrakul, S., Dailey, M. N., & Ekpanyapong, M. (2010). Fruit detection, tracking, and 3D reconstruction for crop mapping and yield estimation. In *2010 11th International Conference on Control Automation Robotics & Vision* (pp. 1181–1186). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Nuske, S., Wilshusen, K., Achar, S., Yoder, L., Narasimhan, S., & Singh, S. (2014). Automated visual yield estimation in vineyards. *Journal of Field Robotics*, *31*(5), 837–860.

Paplinski, A. P., & Wijewickrema, S. N. (2004). Principal component analysis for the approximation of a fruit as an ellipse. Full Papers/WSCG.

Payne, A. B., Walsh, K. B., Subedi, P. P., & Jarvis, D. (2013). Estimation of mango crop yield using image analysis – segmentation method. *Computers and Electronics in Agriculture*, *91*, 57–64.

Qian, J., Xing, B., Wu, X., Chen, M., & Wang, Y. A. (2018). A smartphone-based apple yield estimation application using imaging features and the ANN method in mature period. Scientia Agricola, 75(4), 273-280.

Qureshi, W. S, Payne, A., Walsh, K. B., Linker, R., Cohen, O., & Dailey, M. N. (2017). Machine vision for counting fruit on mango tree canopies. *Precision Agriculture*, *8*, 224, https://doi.org/10.1007/s11119-016-9458-5

Rahman, M., Robson, A., & Bristow, M. (2018). Exploring the potential of high resolution WorldView-3 Imagery for estimating yield of mango. *Remote Sensing*, *10*(12), 1866.

Rahnemoonfar, M., & Sheppard, C. (2017). Deep count: fruit counting based on deep simulated learning. *Sensors*, *17*(4), 905.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: a fruit detection system using deep neural networks. *Sensors*, *16*(8), 1222.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*, http://dx.doi.org/10.1155/2016/3289801

Stein, M., Bargoti, S., & Underwood, J. (2016). Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors*, *16*(11), 1915.

Tao, Y., Zhou, J., Wang, K., & Shen, W. (2018). Rapid detection of fruits in orchard scene based on deep neural network. In *2018 ASABE Annual International Meeting* (p. 1). St. Joseph, MI: American Society of Agricultural and Biological Engineers.

Tseng, P., & Yun, S. (2009). A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, *117*, 387–423.

Viola, P., & Jones, M. J. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (vol. 1, pp. 511–518). Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Wikipedia (2019). Spheroid. https://en.wikipedia.org/wiki/Spheroid#/media/File:Ellipsoid-rot-ax.svg. Accessed 24 June 2019.

Yamamoto, K., Guo, W., Yoshioka, Y., & Ninomiya, S. (2014). On plant detection of intact tomato fruits using image analysis and machine learning methods. *Sensors*, *14*, 12191–12206.

Ye, X., Sakai, K., Manago, M., Asada, S. I., & Sasao, A. (2007). Prediction of citrus yield from airborne hyperspectral imagery. Precision Agriculture, 8(3), 111-125.

Zhu, M. (2004). *Recall, precision and average precision*. Department of Statistics and Actuarial Science, University of Waterloo.

# 5. A deep learning system for yield estimation of melons using UAV images

Submitted to: Computers and Electronics in Agriculture

Author's:  A. Kalantar, Y. Edan, A. Gur, I. Klapp

## A deep learning system for yield estimation of melons using UAV images

A. Kalantar[1,2], Y. Edan[1], A. Gur[3], I. Klapp[2]

**Abstract** Estimating yield production before harvesting is a labor intensive and almost impossible task, since it requires a detailed account of accumulated yield and general yield distribution, in addition to detailed measurements of melon size and location. This work presents a framework for detection and yield estimation of melons from top view color images acquired by a digital camera mounted on an unmanned aerial vehicle based on deep learning techniques. The yield estimation provides both the number of melons and the weight of each melon, allowing to predict the overall weight yield from the entire field. The system includes three main stages: melon detection, geometric feature extraction, and individual melon yield estimation. The melon detection process was based on the RetinaNet deep convolutional neural network. Transfer learning was used for the training to successfully detect small objects in high resolution images. The detection process achieved an average precision score of 0.92 with a F1-score more than 0.9 in a variety of agriculture environments. For each detected melon, feature extraction was applied by using the Chan Vese active contour algorithm and PCA ellipse fitting method. A regression model that ties the ellipse features to the melon's weight is presented. The modified $R^2_{Adj}$ value of the regression model was 0.94. The system results for estimating the weight of a single melon measured by the mean absolute percentage error index achieved 16 percent. Analysis revealed that this can be decreased to 12 percent error with more accurate geometrical feature extraction. Overall yield estimation derived by summarizing the weights of all melons in the field and resulted in only 3 percent underestimation from total actual yield.

**Keywords:** Computer vision, Precision agriculture, Fruit detection, deep convolutional neural networks, Yield estimating/prediction, Weight estimating/prediction, Melon

✉ kalantar@post.bgu.ac.il

*[1] Department of Industrial Engineering & Management, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel*
*[2] Institute of Agricultural Engineering, Agricultural Research Organization (ARO), Volcani Center, Rishon-LeZion, Israel*
*[3] Newe Ya'ar, Agricultural Research Organization (ARO), Volcani Center, Ramat Ishay, Israel*

## Introduction

Precise yield estimation has become an essential skill in the agricultural sector. Generation of yield maps enables to base agronomic decisions related to resource management and marketing leading to improved production. In particular, during the growing season, accurate yield estimation is important for farmers to assess crop yield to retailers.

Estimating the yield production before harvesting at the single fruit level is very labor intensive, since it requires a detailed account of accumulated yield and general yield distribution, and detailed measurements of fruit size and location. Manual evaluation is infeasible  due to lack of human resources and the high cost of human labor (van't Ooster et al., 2014).

Significant progress in precision agriculture applications (Pereira et al., 2017)(Liakos et al., 2018)(Koirala et al., 2019a) has been achieved using advanced technologies including computer vision, satellite navigation systems, remote sensing, geographic information systems, and unmanned aerial vehicles (UAV) (Farjon et al., 2019)

The UAV has become an important tool for field monitoring and precision farming, it allows to explore the crop field in a short time to provide visual information associated with the yield. A prerequisite for observing and analyzing fields from UAV images is the ability to identify crops from image data (Milioto et al., 2017).

Image recognition in agriculture is known as an extremely challenging task due to the unstructured nature of the environment and the objects (Guo et al., 2016), (Carrio, Sampedro, Rodriguez-Ramos, & Campoy, 2017). Interpreting a digital color image of fruits captured in natural field environment is highly complicated due to adverse weather conditions, luminance variability and the presence of dust, insects, obstructions (branches and leaves) and other unavoidable image noises (Pereira et al., 2017). The high variability in the objects which differ by shape, color, size, and texture further complicates this task(Patricio & Rieder, 2018). Advanced R&D has dealt with a wide range of problems in the agriculture field (Liakos et al., 2018),(Koirala et al., 2019a),(Guo et al., 2016). Appling computer vision (CV) techniques such as image recognition on images acquired from UAV, can provide the required information about the fruit location in the field, its size, shape and maturity (Carrio et al., 2017).

Emerging new approaches of image recognition based on machine learning algorithms, such as convolution neural networks, together with big data technologies and high-performance computing, creates new opportunities to unravel, quantify, and understand data intensive processes in agricultural operational environments (Liakos et al., 2018),(Patricio & Rieder, 2018),(Koirala et al., 2019b) . These supervised machine learning methods yield better results than traditional machine learning techniques, which were based on hand-engineered features to encode visual attributes (Koirala et al., 2019a),(Grinblat et al., 2016),(Gongal et al., 2015),(Koirala, Walsh, Wang, & McCarthy, 2019c).

## Yield estimation

Yield estimation research has focused on improving the accuracy of algorithms to predict the number of fruits within images (Koirala et al., 2019a). Only few work relate fruit counts to actual yield (Stajnko et al., 2009),(Calixto et al., 2019),(H. Cheng et al., 2017),(Rahman et al., 2018). In the majority of research, the problem of yield estimation is devised as a fruit detection task that was formulated as a more generic object counting problem and solved by either indirectly by using object detectors (Bargoti & Underwood, 2017a) or explicitly with architectures that learn to count and set up a regression problem to directly infer the number of object instances in the image (Rahnemoonfar & Sheppard, 2017a),(Chen et al., 2017). R-CNN networks for fruit detection resulted with  F1 scores of 0.84 (Sa et al., 2016) and 0.9 (Bargoti & Underwood, 2017a) while detecting apples and mangos over 112 and 270 images respectively, which contains between 100-1000 fruit per image. In another study (Bargoti & Underwood, 2017b) a pixel wise CNN segmentation and regression scheme showed apple counting with F1 score=0.861.  Additional examples of research that used a deeper CNN for counting yield using an object detection pre-defined network with Yolo and SSD architectures were developed for mango (Liang et al., 2018),  apple (Bresilla et al., 2019) and strawberry (Lamb & Chuah, 2018) detection with F1 scores of 0.9 and average precision of 0.842. A new deep CNN named 'MangoNet' (Kestur et al., 2019) used a full convolutional deep CNN to segment mango fruit in the images followed by connected object detection for fruit counting in images. Results of the network demonstrate the robustness of detection for a multitude of factors characteristic to open field conditions such as scale, occlusion, distance and illumination conditions. Another approach to provide yield estimation is counting by regression. A very common and simple implementation of the approach is to use the slope of the linear regression between the machine vision image count and harvest count for a set of calibration trees for fruit load estimation (e.g., (Rahnemoonfar & Sheppard, 2017a),(Chen et al., 2017)).

## Fruit weight estimation using computer vision

Estimating total yield in terms of weight is a complex task that requires both accurate recognition of the fruits in the field and estimation of individual fruit size. Fruit weight can be correlated to linear dimensions in many fruits (Sa et al., 2016), (Calixto et al., 2019). Such a relationship allows fruit weight to be estimated using machine vision, given a measure of camera to fruit distance (Koirala et al., 2019a).  In another work, weight estimation of yellow melons was achieved using contour detection that relied on Otsu's segmentation (Otsu, 1979). An artificial neutral network was used in another research to learn the relation between fruit segmentation and the fruit weight (H. Cheng et al., 2017).

This work presents a framework for detection and yield estimation of melons from color images acquired from a digital camera mounted on an unmanned aerial vehicle drone. The work is a step forward from previous work (Dashuta & Klapp, 2018),(Kalantar et al.,

2019) in which a pipeline for yield estimation of melons from top view UAV images of a melon field was suggested. The pipeline included three main stages: melon recognition, geometric feature extraction, and individual melon weight estimation.  While the previous research (Dashuta & Klapp, 2018),(Kalantar et al., 2019) provided an end to end solution to determine individual melon weight, estimation suffered from inaccuracies, resulting from false positive detection of the melons regions. This was due to limitations of the relatively simple CNN classification model used and errors in the proposed active contour schema. In this paper, the the two step region proposal was replaced by a more accurate RetinaNet based neural network (Lin, Goyal, et al., 2017). Additionally, the active contour algorithm for feature extraction was improved. In the new model presented in this paper ellipse fitting by minimizing the PCA (Wijewickrema & Paplinski, 2005) is applied on the results of a Chan-Vese active contour (Chan & Vese, 1999), then perform.

The rest of the paper is organized as follows: Section 2 details the materials and methods applied in the research. The algorithm is presented in Section 3. Results and discussion of the tested system provided in section 4. Section 5 include conclusions and recommendations for future work.

## Materials and Methods

### Dataset acquisition

The data was acquired at midday on 23 July 2018 from a 180X260 meter open field at Newe Ya‘ar (32°43'05.4"N 35°10'47.7"E) in Israel. The melon plants were at ripening stage.

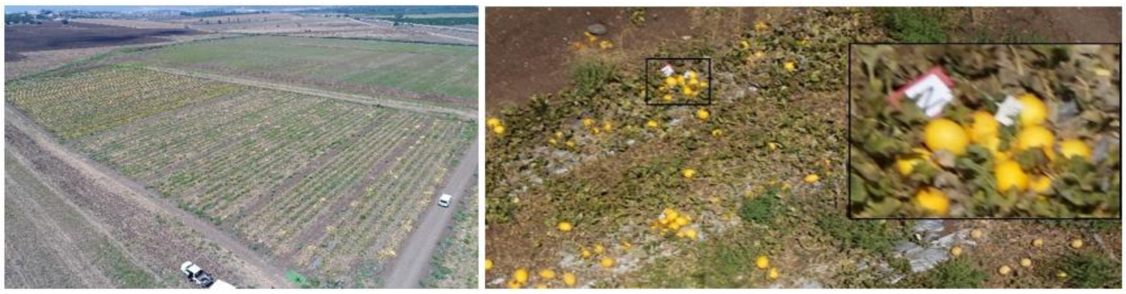


*Figure 22 – Tested melon field at Newe Ya‘ar (left), a typical image with marked melon (right)*

Data was collected using a “Phantom 4 Pro” UAV equipped with a color RGB camera DJI FC6310 type. The UAV hovered about 15 m above the field, with the camera facing vertically downward during the image acquisition. The images resolution were 5472 × 3648 pixels, with each image containing hundreds of melons of different size, shape and color. Before image acquisition the following operations were conducted in the field:

- Irrigation was stopped one week before to reduce foliage.

- Just before the image acquisition 138 melons were randomly selected and marked in the field by placing a sign next to them. The sign was faced up, towards the drone so that it could be recognized in the image. These melons were used as ground-truth data and were analyzed in the image processing stage by a "Tomato Analyzer" tool (Gonzalo et al., 2009) providing for each melon its size and weight characteristics.

Since several of melons were damaged during the collecting and measuring process, it was measured only 116 melons. As a result an additional data that were collected the year before at the same place with a Sony ILCE-5000 camera mounted on a Quad-Copter drone was also used (Kalantar et al., 2019). This data contained 32 measured melons which were used later to build the yield estimation regression model. The testing process was conducted on an image from another dataset that was collected 2 years before (Dashuta & Klapp, 2018).

## Data preparation

To reduce computational time the images were divided into 592 × 394 sub images with a small overlap between the sub images to prevent loss of melon's images. To preserve image aspect ratio, the height overlap was set to 30 pixels and the width overlap was set in the following proportion:

$$width\ overlap = 30 * \frac{width}{hieght} \quad (1)$$

All sub images were annotated by an expert who marked manually bounding boxes of the melons using a graphical image annotation tool "labelImg" version 1.8.1 ("Labelimg, graphical image annotation tool," 2018).

Eight images were selected, six of them from 2018 season, one of them from 2017 season (Kalantar et al., 2019) and one from 2016 season (Dashuta & Klapp, 2018). All images were subdivided into a grid of 10x10 resulted in 800 sub-images where each one of them was manually tagged. Four images from the 2018 season were allocated for the training process, the remaining four images were used for testing the model's performance.

## Algorithm, transfer learning and augmentation

The algorithm relied on transfer learning (Huh et al., 2016),(X. Wang & Schneider, 2014),(Yosinski et al., 2014), (Krizhevsky et al., 2012),(Donahue et al., 2014) using the RetinaNet pre-trained CNN (Lin, Goyal, et al., 2017). The RetinaNet network was originally trained on the ImageNet dataset (Krizhevsky et al., 2012) and then finetuned using the transfer learning method. 4220 labeled melons taken from 4 different images were used for the transfer learning. This data set was enlarged by data augmentation preformed during the fine tuning process of melon fruit detector. Several types of augmentation were performed in different combinations including rotation, translation, shear, scaling and flipping operation, (Table 1).

The algorithm was developed on a NVIDIA GeForce GTX 1080ti GPU, Intel® Core™ i7-8700, 64-bit six-core 3.2GHz CPU, 32 GB memory running on Microsoft Windows 10 system.

Table 4 - The augmentation operations values which were observed.

| Augmentation Type | Values |
|---|---|
| minimum rotation in radians for the transform as scalar. | -0.1 |
| maximum rotation in radians for the transform as scalar. | 0.1 |
| minimum translation for the transform as 2D column vector. | (-0.1, -0.1) |
| maximum translation for the transform as 2D column vector. | ( 0.1, 0.1) |
| minimum shear angle for the transform in radians. | -0.1 |
| maximum shear angle for the transform in radians. | 0.1 |
| minimum scaling for the transform as 2D column vector. | ( 0.9, 0.9) |
| maximum scaling for the transform as 2D column vector. | ( 1.1, 1.1) |
| chance (0 to 1) that a transform will contain a flip along X direction. | 0.5 |
| chance (0 to 1) that a transform will contain a flip along Y direction. | 0.5 |

## Performance metrics

The metrics used for individual melon detection and melon's weight estimation from the melon's geometrical features included detection performance and yield prediction.

### Detection performance measures

The IoU metric, also referred to as the Jaccard index, is used to quantify the percent overlap between the target bounding box and the predicted output (Koirala et al., 2019a). By applying the IoU, we can determine if a detection is valid (True Positive) or not (False Positive) comparing to a predefined threshold. The calculation of IoU is given by the overlapping area between the predicted bounding box and the ground truth bounding box divided by the area of union between them. In this work we used IoU>0.5 which is considered as standard by the Pascal Visual Object Classes (VOC) challenge (Everingham, Van Gool, Williams, Winn, & Zisserman, 2010). This threshold was used also by (Liang et al., 2018) ,(J. Wang et al., 2014) for detecting fruits.
Analyses was conducted using a common threshold equal to 0.5 for IoU (Everingham et al., 2010). This threshold is much higher than 0.2 which is considered sufficient for the fruit-mapping application (Krizhevsky et al., 2012), and provides a better localization of the fruit. The consideration in choosing a higher threshold despite the small melon size, stemmed from the fact that in further stages of the algorithm, it was necessary to recognize the melon by fully adapting it to the full melon fruit contour, not part of it. Hence, a better overlap area was required.

The four detection performances measures used were (Koirala et al., 2019a): Precision (P) indicates the fraction of the algorithm's predictions that are melons. Recall (R) is the fraction of melons in the image that were detected by the algorithm. F1-score (F1) is a metric that balances between precision and recall by calculating the weighted average (harmonic mean) such that:

$$P = \frac{TP}{TP+FP} \quad , \quad R = \frac{TP}{TP+FN} \quad , \quad F1 = \frac{2 \cdot P \cdot R}{P+R} \qquad (2)$$

Where (TP) True Positive, is a true detection, a fruit detection was considered to be a true positive if the predicted and the ground-truth bounding box had an IoU greater than a fixed threshold. (FP) False positive, is a false detection, refers to an algorithm's mistake of predicting background as a melon. (FN) False Negative, is a ground truth melon not detected, caused due to the failure to detect a real melon by the algorithm. (TN) True Negative, represent a corrected misdetection, and was not used since in the current object detection task there are many possible bounding boxes that should not be detected within an image. Average Precision (AP) is a summary of the precision-recall curve and is calculated as a weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight. The AP measure used in the PASCAL-VOC challenges which is a benchmark in visual object category recognition and detection (Everingham et al., 2010) was also used.

**Melon weight prediction**

The regression quality was evaluated using the adjusted $R^2$ statistic (Miles, 2014) since it is more robust to the number of variables (Montgomery, 2017):

$$R^2_{Adj} = 1 - \frac{SSE/(n-p)}{SST/(n-1)} \qquad (3)$$

Where, SST is the total sum of squares and SSE is the sum of squares of residuals, the number of points is n=30, the number of parameters is p=2.

The system results for estimating the weight of a single melon was measured by the mean absolute percentage error index comparing the weight estimated from the image processing to the ground truth measured. Overall yield estimation was derived by summarizing the weights of all marked melons in the field and comparing to their overall actual weight.

## Analysis

Evaluation for melon detection was done on four different images from different seasons and agriculture environments, with a total of 2,394 melons in four different environments (Table 3). A visualization of color and texture diversity in the different seasons is presented in Figure 8.

*Table 5 – The different environmental conditions for detection evaluation*

| | *Image 1* | *Image 2* | *Image 3* | *Image 4* |
|---|---|---|---|---|
| *Period* | 2 Years Before | Year Before | Current | Current |
| *Environment* | lots of foliage and occlusions | muddy, less foliage | regular foliage | lots of foliage |
| *Number of Melons* | 270 | 848 | 1071 | 205 |

A three-level qualitative analysis was performed to evaluate:

- melons with perfectly ellipse fits or slightly underfitted/overfitted ellipse.
- melons with ellipses that were not fitted well.
- all melons.

Two measures were calculated for each level: the mean absolute percentage error measure and the overall deviation of weight percentage. During the analysis all the melons were included since in practice it is almost impossible to define an outlier.

## Algorithm

The algorithm was developed to automatically detect and estimate the weight of an individual melon. The input for the system is an aerial RGB image of a melon field, and the output is a report that includes each melon's location and weight. The algorithm pipeline (Figure 2) includes the following three sequential main stages (Kalantar et al., 2019): 1) melon detection, 2) feature extraction, 3) yield estimation.

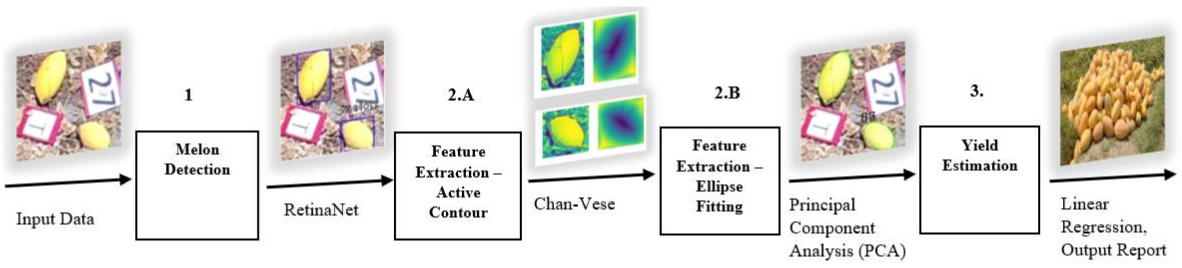

Input Data
1
Melon Detection
RetinaNet
2.A
Feature Extraction – Active Contour
Chan-Vese
2.B
Feature Extraction – Ellipse Fitting
Principal Component Analysis (PCA)
3.
Yield Estimation
Linear Regression, Output Report

*Figure 23 – System pipeline for automated detection and yield estimation of melons*

### Melon Detection using RetinaNet

In our previous works (Dashuta & Klapp, 2018),(Kalantar et al., 2019) , melons were detected using a two stage approach. First, a region of interest (ROI) suspected to contain a melon was implemented using the Viola-Jones detector (Viola & Jones, 2001). Then, by using a predefined trained convolution neural network system, each ROI was classified whether it contains a melon or not. Despite the high accuracies achieved for IoU set to 0.2, an average precision of 0.82 and F1 score of 0.85 was achieved. In some cases the ROIs were not accurate enough and limited the geometrical feature extraction in the next stage. To improve performance, an advanced detection algorithm was applied in the current work.

Several CNN schemes were considered including Faster R-CNN (Ren et al., 2015), You Only Look Once (YOLO) (Redmon et al., 2016), Single Shot Multi-Box Detector (SSD) (Liu et al., 2016) and RetinaNet (Lin, Goyal, et al., 2017) previously used for fruit detection (Koirala et al., 2019b). The R-CNN suffers from slow rate and minimal required target size, the YOLO was fast but not accurate enough, SSD performed poorly due to the extreme class's imbalance between the background and the melons (a minority in the image). RetinaNet (Lin, Goyal, et al., 2017) was applied to overcome the computational limits and the object size and provide the required accuracy.

**Implementation details**

The RetinaNet algorithm was built using the Microsoft COCO dataset properties (Lin et al., 2014) based on the ResNet50 as backbone network (Figure 2). In order to able the network to detect small objects, the major modification introduced was changing the anchor box. The initial parameters of the network regarding the anchor box included boxes with areas of {322, 642, 1282, 2562, 5122} with stride of {16, 32, 64, 128, 256} on pyramid levels P3 to P7 respectively, for each level anchors generated with scales values of {20, 21/3, 22/3 } and three aspect ratios = {1:2, 1:1, 2:1}. In total nine different anchors were generated at each level. Since these parameters were not suitable for detecting small object the anchor boxes were reduced to {162, 322, 642, 1282, 2562} with stride of {8, 16, 32, 64, 128} along with increasing the scales parameter values to {1, 1.2, 1.6} and updating aspect ratio values to {1:2, 1:1, 2:1, 3:1}, resulting with twelve anchors per level. These changes produce smaller anchors that more tightly fit the objects resulting in improved detection.

Additionally, before training the network the threshold value used to determine whether the box contains melon or background was set for IoU of 0.5.

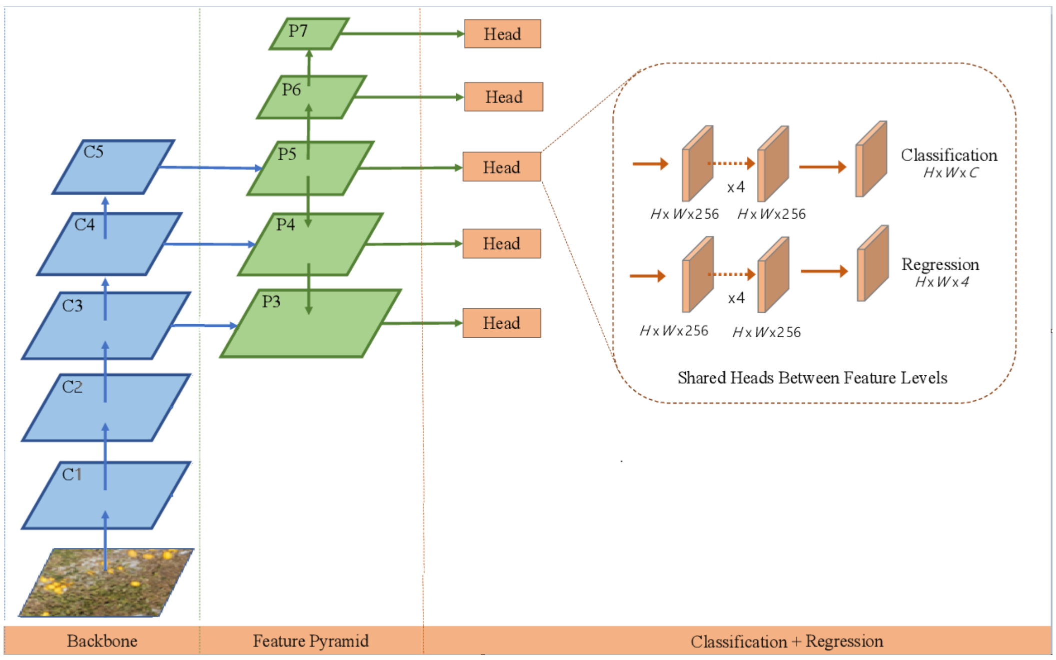


*Figure 24 – RetinaNet implementation in the proposed pipeline*

### Compose sub images

After each sub-image is separately processed by the RetinaNet the algorithm recomposes the original image by using the initial coordinates of each sub image in the original image. This results with a complete image with all the detected melons anchor boxes coordinates. To avoid duplicate identification of melons which were located at the overlapping areas between two images a non-maximum-suppression (NMS) (Neubeck & Van Gool, 2006) algorithm was applied separately from the RetinaNet. The algorithm removes all overlapped anchor boxes which have an intersection over union (IoU) greater than 0.5.

## Melon geometrical feature extraction

The first stage provides a list with anchor boxes with their coordinates that present the location of each detected object in the image. Since each anchor box contains only single melon, we can determine the location of all detected melons in the field.
Moreover, using the anchor boxes list, the entire image is split into many sub images which contain only one melon at the center of the image. This enables to perform geometrical feature extraction of the individual melon fruit.
The melons geometrical features were derived from an ellipse representation of the melon similar to our previous work (Dashuta & Klapp, 2018; Kalantar et al., 2019). An improved model was developed in this work to fit an ellipse model to the melon's contour, from which we gain the melons geometrical features. This stage was separated into two sub-stages, to overcome occlusions arising mainly from foliage that disturb the elliptical shape. First, the free form contour was estimated using the Chan-Vese active contour algorithm (Chan & Vese, 1999), also known as "Active Contours Without Edges". An output binary image where the fruit and the background segmented was derived. Then, an ellipse was fitted to the binary image using PCA method.

### Chan-Vese active contour

The algorithm starts with an initial pre-defined contour for each melon separately. It evolves the contour according to equation 5 which includes four terms so that it stops on the boundaries of the foreground region. This is conducted using a gradient descent method, which in each iteration the contour shrinks or expands.

$$\arg\min F(c_1, c_2, C) =$$
$$\mu_1 \cdot Lenght(C) + \lambda_1 \int_{inside(C)} |u_0(x,y) - c_1|^2\, dxdy + \lambda_2 \int_{outside(C)} |u_0(x,y) - c_2|^2\, dxdy \tag{5}$$

The first term is considered as regularization term. Length penalize limitation (total contour length (*C*) with weight $\mu_{1)}$ provides the solution's smoothness. The last two terms perform a pixel wise estimation to decide if the pixels belong to the inner object or to the background according to its gray scale u(x,y) distance from the average gray scale value of the two zones denoted $c_1$ and $c_2$ and weights $\lambda_1$, $\lambda_2$ respectively. The cost function F($c_1$,$c_2$,$C$) is solved iteratively.

To ensure the contour belongs to the melon, an initial condition of a square contour with size of eleven pixels was assumed. The integrals weights set equals 1 as in (Chan & Vese, 1999) with the maximum number of iterations was limited to 500.

### Ellipse fitting

The ellipse model is characterized by 5 parameters: Centroid x co-ordinate ($x_0$), centroid y co-ordinate ($y_0$), semi-major axis ($a$), semi-minor axis ($b$) and angle of tilt ($\theta$). Where every point $\mathbf{x} = (a, c, x_0, y_0, \theta) \in \mathbb{R}^5$ in the parameter space corresponds to a single ellipse in a given anchor box.

$$\frac{\left[(x-x_0)\cos(\theta)-(y-y_0)\sin(\theta)\right]^2}{a^2}+\frac{\left[(x-x_0)\sin(\theta)+(y-y_0)\cos(\theta)\right]^2}{b^2}=1 \qquad (6)$$

Principal Component Analysis (PCA) method was used in order to fit an ellipse to the contour (Wijewickrema & Paplinski, 2005). The overall geometrical feature extraction process is presented in figure 4.

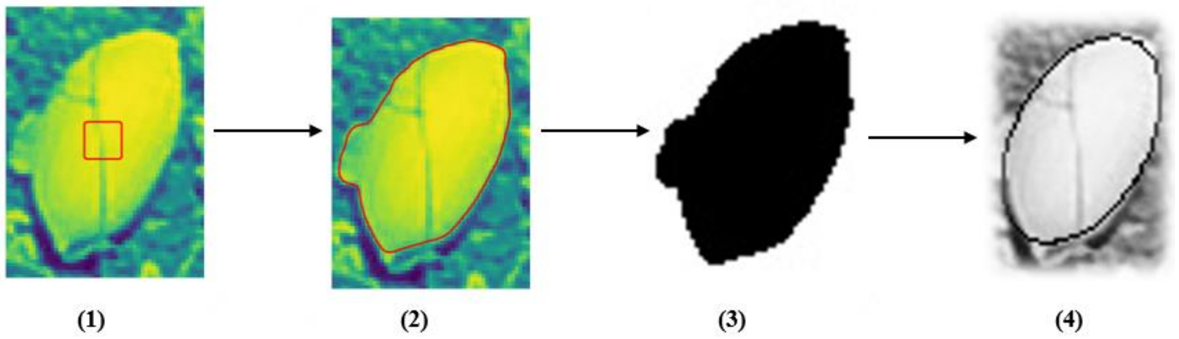


*Figure 25- Feature extraction process: (1) An initial contour defined at the center of each anchor box (2) Chane Vese active contour (3) Binary image (4) PCA ellipse fitting method on top of the binary image applied, as a result, geometrical feature extraction of the individual melon received.*

## Yield weight estimation model

Yield estimation relies on a regression model between the melon's geometrical representation by axis symmetric spheroid model (Figure 5) and the melon's weight based on our previous work (Kalantar et al., 2019). The parameters derived from the fitted ellipse section in the 2D image, the sizes of minor and major axes, were correlated to the semi-height and semi-width of the melon based on which the melon weight was predicted.

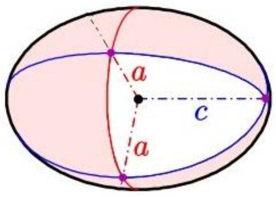


*Figure 26 - Ellipsoid model*

**Melon weight estimation regression model**

The regression model was built from with 30 randomly individual melons which were both imaged by the UAV and measured for their weight and geometry in the laboratory.
The max height (2*c) and max width (2*a) of each melon was acquired. Table 2 presents correlation scores to various regression models between H,W to the weight.

*Table 6 – different types of regression fitted for estimating melon weight*

| Type of correlation | Parameters combination | $R^2_{Adj}$ value |
|---|---|---|
| **Linear** | $c + a$ | 0.914 |
| **Area** | $c * a$ | 0.87 |
| **Volume** | $c * a^2$ | 0.94 |
| **Logarithm** | $\log(c) * (\log(a))^2$ | 0.91 |

The best regression was based on the $R^2_{Adj}$ value. The resulting weight is in linear proportion to the fruit volume:

$$Weight = 0.1096653 + 0.003397929 \cdot c \cdot a^2 \qquad (9)$$

The training set results are presented in Figure 6.

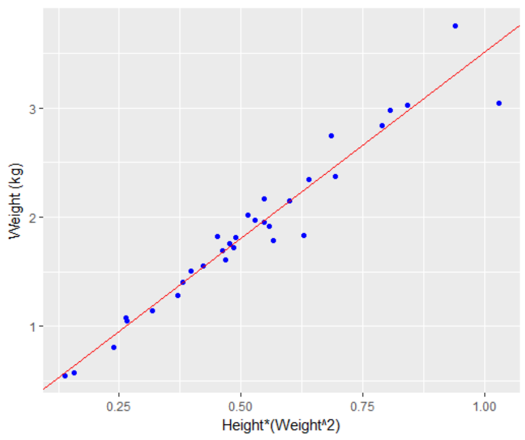


*Figure 27 - The regression model resulted with a* $R^2_{Adj}$ *of 0.94*

**Determining the spatial resolution of remote sensing sensor**

The regression in Eq.9 relies on volumetric features given in metric units (Dashuta & Klapp, 2018). The minimization of the active couture resulted in ellipse parameters given in pixels and was translated to millimeters using the ground sample distance (GSD) measurement (Kalantar et al., 2019). The naïve first order estimation for the GSD includes three measurements:

$$GSD = \frac{h \cdot \Delta p}{f} \tag{10}$$

Where $h$ is the Altitude at which the image was acquired, $\Delta p$ is the sensors pixel size and $f$ is the focal length setting at which the image was acquired.

In order to calculate this ratio, GPS values of each image separately were calculated to estimate the approximate height ($h$). The focal length and pixel size have a fixed value enabling to calculate the ratio for each image based on the estimated height. The overall regression process is illustrated in Figure 7.

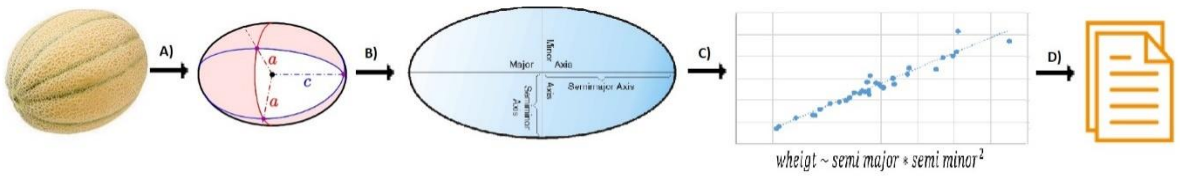


*Figure 28 - Yield estimation process: stage A) represent a melon by spheroid shape - use pre-knowledge of melon shape, stage B) convert from 3D to 2D – the width and depth would be minor axes and length is the major axes, stage C) use linear regression for predicting yield of each melon, stage D) generate a report for each melon location and yield estimation for the all field*

## RESULTS AND DISCUSSION

### Fruit detection results

Results (Table 4) reveal an overall average precision score of 0.92 with all images obtaining F1-scores higher than 0.9. The best results were obtained for image 3 since its environment was similar to the training images environment.
These results are better than in the previous work (Kalantar et al., 2019), in which we obtained 0.82 average precision and 0.85 of F1-score with 0.2 threshold for IoU. The improvement is due to the better detection of cluster melons and is reflected also in the Precision – Recall curve (Figure 9).

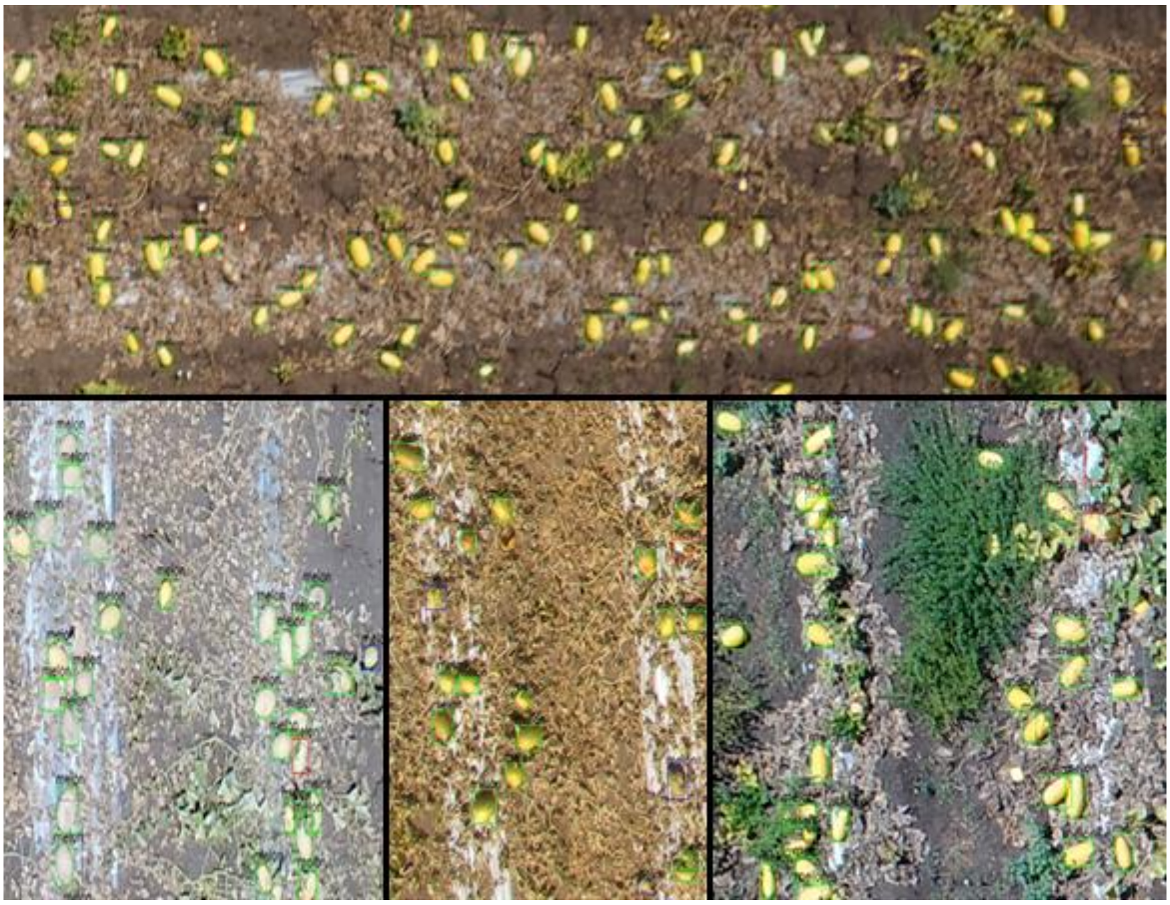


*Figure 29- Selected regions in output images from different seasons and environments. The green boxes present TP, red boxes present FP and blue boxes present FN.*

*Table 7 - Detection results*

| | *Image 1* | *Image 2* | *Image 3* | *Image 4* |
|---|---|---|---|---|
| *True Positive* | 252 | 800 | 1032 | 180 |
| *False Positive* | 18 | 48 | 39 | 25 |
| *False Negative* | 35 | 89 | 12 | 11 |

|  |  |  |  |  |
|---|---|---|---|---|
| *Precision* | 0.93 | 0.94 | 0.96 | 0.88 |
| *Recall* | 0.88 | 0.90 | 0.99 | 0.94 |
| *F1-score* | 0.90 | 0.92 | 0.98 | 0.91 |

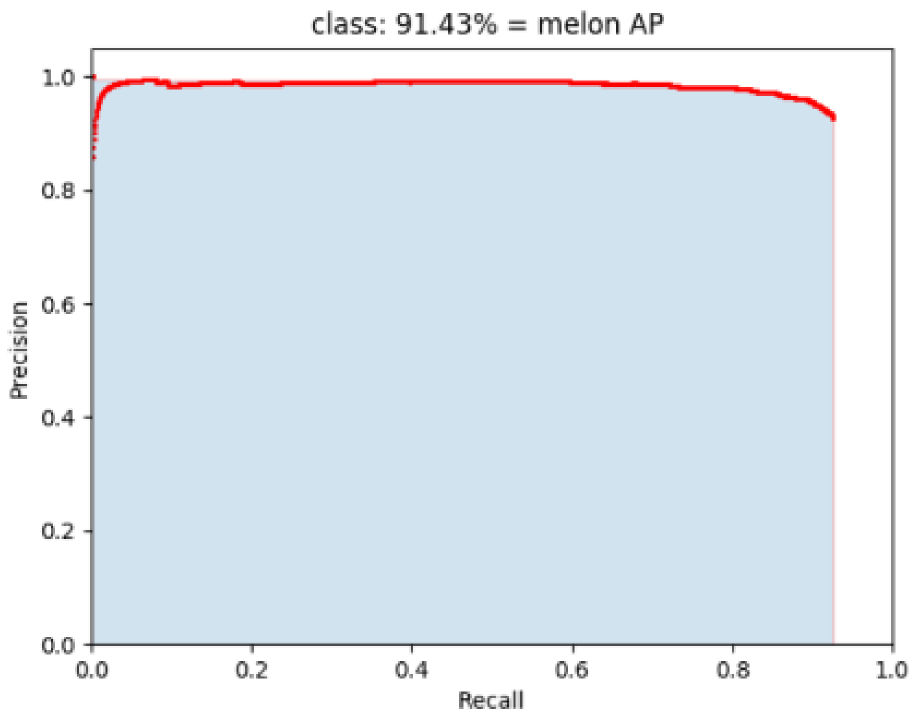


*Figure 9- Precision-Recall curve of melon detection IoU=0.5*

While detection is very promising, there are places where the algorithm failed. Observing the results, most of missed melon detection cases occurred when melons were occluded by foliage or by another melon. In some cases, the algorithm tagged stones as melons or dust that contained a melon that was removed but the melon’s shape was preserved. In addition, there were some cases where the algorithm detected melons but during the labeling process the labeler missed it. Another case for improvement relates to the process of composing the sub-images back to a major image with the NMS algorithm. During this process, it was noticed that in some cases melons that were placed in borders in one image and detected by the neural network in other overlapping image were cropped through the NNS process. This slightly reduced the detection performance. Example for this "edge effect" of the NMS algorithm presented in Figure 10. Satisfying results were achieved despite the relatively small fruit size with respect to the image resolution.

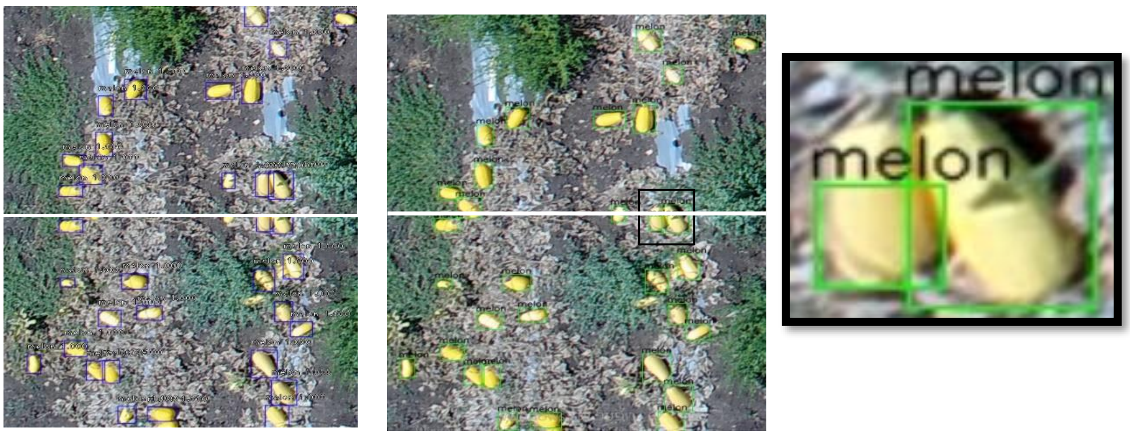


*Figure 10 - Example of an NMS failure, left presented 2 images with detected anchor boxes in blue color, then the images composed into one image using NMS algorithm for producing final anchor boxes presented in green, a white line illustrated the composition. The right image presents the failure of the NMS algorithm for detecting the left melon in the image.*

## Validation of regression quality and final yield estimation

The regression quality test [equation 10] was conducted on 116 randomly selected melons from 2018 year. The selected melons were different in their size, in order to estimate the size of the error predicted by regression use.

The mean absolute percentage error (MAPE) index for individual melon estimation was 9% with an overweight overall yield estimation error of 2.9%.

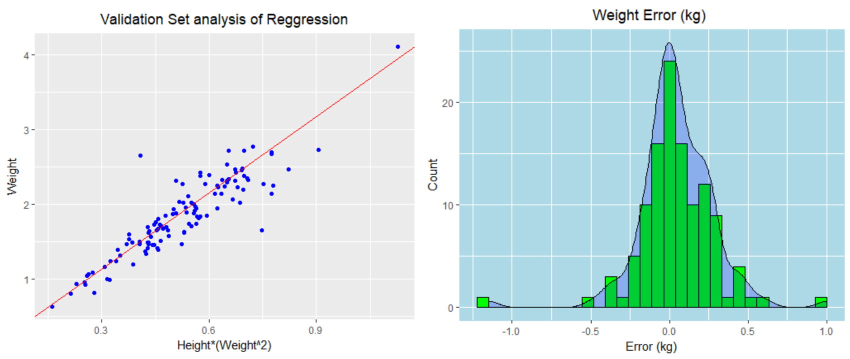


*Figure 11 - Validation set analysis of the regression model (left), Histogram of error in weight estimation of the validation set with the kernel density estimation shown as a continuous (right).*

## Chan-Vese performances in detecting of melon's border

All 116 marked melons were detected by the system. The ellipse fitting for 65 of the melons was perfect as qualitatively assessed (Figure 12). For 33 melons the ellipse was not good enough, with most of them obtaining an under fitted ellipse (Figure 13B). In 18 images an ellipse was not matched at all (e.g. Figure 13A and 13C).

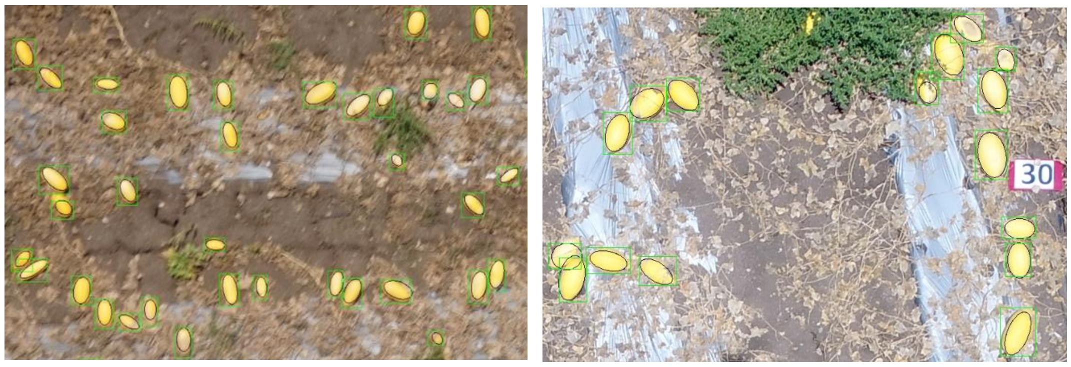


*Figure 12 - Selected region in an output image, in green presented the anchor box which indicates the location of the melon, in each box, an ellipse was fitted for feature extraction of the melon.*

The main cause of the lack of perfect ellipse fittings was the Chan Vese algorithm limitation, which is based on regional intensity differences which does not apply in all cases. In the images where the intensity difference between the fruit and the background was clear, the results were good. when intensities gap became lower, curve miss match was reflected in several situations. The most common situations occurred when the melon was covered by foliage (Figure 13A), or the melon was overlapped with another melon (Figure 13B). Another problem was related to heterogeneous color gradients of the melon surface (Figure 13C) resulting in partial ellipse fitting.

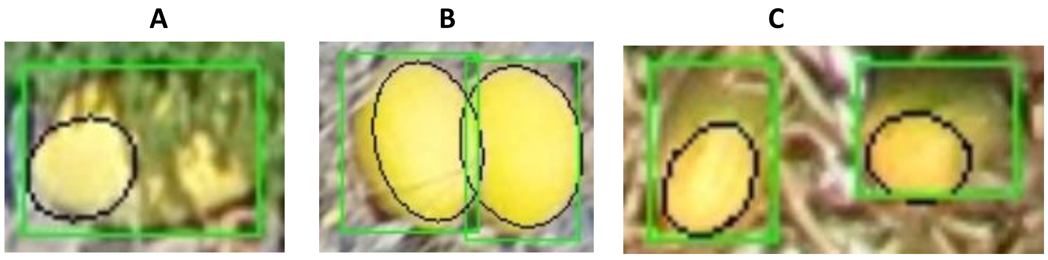


*Figure 30 - Examples for lack of ellipse adjustment caused by Chan Vese algorithm limitations.*

## Overall and individual performances in weight estimation

Results for each individual melon (Table 5) and overall results (Table 6) reveal that the melons that had good ellipse fitting achieved almost twice better estimation in each of the measures with an average percentage of error about ±12% (Table 6), comparing to melons with partial ellipse fitting,
This deviation can be explained by the fact that it included melons with not perfect match. Another reason for deviation can be related to the calculation of the GSD ratio and may be due to an error arising from the regression equation, which originally achieved 9 percent error of MAPE.

Overall yield prediction results in 3% error estimation for all the melons. The well fitting melons result in an under estimation error of 2% which corresponds to the 2% error of the regression quality.

*Table 8 - An example of detailed account for results for each melon*

| *Num #* | *Location and Yield estimation* | | | | | | *analysis* | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | yCenter | xCenter | angle | major [cm] | minor [cm] | Estimated Wieght (KG) | Actual Weight (KG) | Delta (KG) | Ellipse Fitted goodness | Gap analysis |
| *18* | 3369.30 | 4965.04 | 127.62 | 19.78 | 15.35 | 2.09 | 2.32 | -0.23 | slightly underfitted | illumination |
| *28* | 1067.02 | 4810.48 | 117.21 | 20.54 | 14.72 | 2.00 | 1.994 | 0.01 | perfect | |
| *29* | 280.04 | 4705.46 | 90.51 | 16.01 | 12.73 | 1.21 | 0.922 | 0.29 | overfitted | Chane Vese |
| *58* | 3152.33 | 1155.82 | 53.71 | 14.33 | 12.35 | 1.04 | 1.634 | -0.60 | underfitted | occlusion |
| *63* | 1527.32 | 5082.48 | 65.19 | 16.24 | 14.19 | 1.50 | 1.672 | -0.17 | slightly underfitted | |
| *64* | 1811.97 | 5081.09 | 73.53 | 17.67 | 14.55 | 1.70 | 1.63 | 0.07 | perfect | |
| *65* | 1866.00 | 5097.28 | 118.77 | 18.56 | 17.53 | 2.53 | 2.72 | -0.19 | slightly underfitted | |
| *84* | 513.37 | 2801.18 | 26.55 | 18.27 | 14.03 | 1.64 | 1.49 | 0.15 | overfitted | |
| *87* | 3290.75 | 1777.26 | 121.67 | 21.37 | 14.57 | 2.04 | 2.378 | -0.34 | underfitted | illumination |
| *88* | 3588.65 | 4839.86 | 147.63 | 16.05 | 14.41 | 1.53 | 1.694 | -0.17 | no match | cluster melon |
| *90* | 2966.70 | 5104.94 | 113.93 | 16.81 | 13.18 | 1.35 | 1.618 | -0.27 | underfitted | Chane Vese |
| *91* | 2974.83 | 5160.54 | 95.56 | 16.68 | 13.82 | 1.46 | 1.47 | -0.01 | slightly underfitted | |
| *1R* | 1288.31 | 2587.32 | 14.98 | 17.80 | 10.71 | 0.98 | 1.196 | -0.22 | perfect | GSD |
| *3R* | 1236.47 | 4902.30 | 25.16 | 23.10 | 13.87 | 2.00 | 1.83 | 0.17 | slightly overfitted | Chane Vese |
| *H* | 1369.05 | 2441.38 | 149.48 | 20.90 | 10.95 | 1.17 | 1.406 | -0.23 | underfitted | occlusion+ illumination |
| *K* | 1193.77 | 3661.82 | 172.47 | 19.35 | 13.31 | 1.57 | 1.704 | -0.14 | slightly underfitted | |
| *L* | 1199.33 | 4510.84 | 122.26 | 22.30 | 13.82 | 1.92 | 1.848 | 0.07 | perfect | |
| *S* | 1440.16 | 147.43 | 10.70 | 20.76 | 12.62 | 1.51 | 1.564 | -0.05 | no match | illumination |

*Table 9 – Weight estimation results*

| | NUMBER OF MELONS | TOTAL ACTUAL WEIGHT (KG) | OVERALL DEVIATION OF WEIGHT % | MAPE % |
|---|---|---|---|---|
| **GOOD FITTING OF ELLIPSE** | 65 | 117.362 | -2 | 12 |
| **PARTIAL FITTING OF ELLIPSE** | 51 | 97.014 | -4 | 20 |
| **OVERALL** | 116 | 214.376 | -3 | 16 |

### Influence of feature extraction on the weight estimation

Above we distinguished between "Good" and "Partial" ellipse fitting. The detailed examination of the relation between the feature extraction process and actual size of a single melon in these two groups revealed that the melon's minor axis extraction was performed less effectively than the major axis. Since the minor axis has polynomial degree in the regression for estimating the melon weight, it gains more influence on the final weight result. This explains significant gap between the yield estimation per single melon results and the MAPE measurement between the two ellipse fitting groups (Table 7).

*Table 10 - Correlation between extracted minor and major axis results with actual sizes of the melon, per type of ellipse fitting*

| | *Good fitting of ellipse* | *Partial fitting of ellipse* |
|---|---|---|
| *Minor Axis* | 0.608867 | 0.39205 |
| *Major Axis* | 0.687793 | 0.646899 |

### Conclusions and recommendations

A deep learning system for detection and yield estimation of melons from top view UAV images of a melon field was developed. Detection was based on the RetinaNet, a pre-defined deep convolutional neural network, which was trained using transfer learning in order to deal with detecting small objects in high resolution images such as melons from top view UAV images. The detection process achieved an overall average precision score of 0.92, and F1-score more than 0.9 for a variety of different agriculture environments. Estimating the yield of a single melon measured by the MAPE index achieved 16% which can be improved to 12%. Overall yield estimation resulted in 3% underestimation.

If improved yield performance is needed future work should apply more advanced feature extraction algorithms. Since the detection algorithm performed well, the Chan Vese method should be replaced perhaps with a CNN which can provide local segmentation of the melon (e.g., the U-net (Ronneberger, Fischer, & Brox, 2015)). Another direction could be to perform instance segmentation (e.g., Mask r-cnn (He, Gkioxari, Dollár, & Girshick, 2017)) and add another layer on top of the segmentation to regress the melon's weight. However, both these methods require to label images at the pixel level which is a complicated and labor intensive task.

## References

Agrawal, P., Girshick, R., & Malik, J. (2014). Analyzing the performance of multilayer neural networks for object recognition. *European Conference on Computer Vision*, 329–344.

Ali, I., Cawkwell, F., Dwyer, E., & Green, S. (2016). Modeling managed grassland biomass estimation by using multitemporal remote sensing data—A machine learning approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *10*(7), 3254–3264.

Amara, J., Bouaziz, B., & Algergawy, A. (2017). A Deep Learning-based Approach for Banana Leaf Diseases Classification. *BTW (Workshops)*, 79–88.

Anwar, S. M., Majid, M., Qayyum, A., Awais, M., Alnowami, M., & Khan, M. K. (2018). Medical image analysis using convolutional neural networks: a review. *Journal of Medical Systems*, *42*(11), 226.

Bac, C. W., van Henten, E. J., Hemming, J., & Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, *31*(6), 888–911.

Bargoti, S., & Underwood, J. (2017a). Deep fruit detection in orchards. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 3626–3633.

Bargoti, S., & Underwood, J. P. (2017b). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, *34*(6), 1039–1060.

Bergerman, M., Billingsley, J., Reid, J., & van Henten, E. (2016). Robotics in agriculture and forestry. In *Springer handbook of robotics* (pp. 1463–1492). Springer.

Boureau, Y.-L., Ponce, J., & LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 111–118.

Bresilla, K., Perulli, G. D., Boini, A., Morandi, B., Grappadelli, L. C., & Manfrini, L. (2019). Single-shot convolution neural networks for real-time fruit detection within the tree. *Frontiers in Plant Science*, *10*.

Burrus, C. S., & Parks, T. W. (1985). *and Convolution Algorithms*. Citeseer.

Calixto, R. R., Neto, L. G. P., da Silveira Cavalcante, T., Aragão, M. F., & de Oliveira Silva, E. (2019). A computer vision model development for size and weight estimation of yellow melon in the Brazilian northeast. *Scientia Horticulturae*, 108521.

Carrio, A., Sampedro, C., Rodriguez-Ramos, A., & Campoy, P. (2017). A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*, *2017*.

Chan, T., & Vese, L. (1999). An active contour model without edges. *International Conference on Scale-Space Theories in Computer Vision*, 141–151.

Chen, S. W., Shivakumar, S. S., Dcunha, S., Das, J., Okon, E., Qu, C., … Kumar, V. (2017). Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robotics and Automation Letters*, *2*(2), 781–788.

Cheng, G., & Han, J. (2016). A survey on object detection in optical remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, *117*, 11–28.

Cheng, H., Damerow, L., Sun, Y., & Blanke, M. (2017). Early yield prediction using image analysis of apple fruit and tree canopy features with neural networks. *Journal of Imaging*, *3*(1), 6.

Cohen, T., & Welling, M. (2016). Group equivariant convolutional networks. *International Conference on Machine Learning*, 2990–2999.

Dahl, G. E., Sainath, T. N., & Hinton, G. E. (2013). Improving deep neural networks for LVCSR using rectified linear units and dropout. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 8609–8613.

Dashuta, A., & Klapp, I. (2018). Melon Recognition in UAV Images to Estimate Yield of a Breeding Process. *Optics and Photonics for Energy and the Environment*, ET4A--2.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. *International Conference on Machine Learning*, 647–655.

Dyrmann, M., Jørgensen, R. N., & Midtiby, H. S. (2017). RoboWeedSupport-Detection of weed locations in leaf occluded cereal crops using a fully convolutional neural network. *Advances in Animal Biosciences*, *8*(2), 842–847.

Dyrmann, M., Karstoft, H., & Midtiby, H. S. (2016). Plant species classification using deep convolutional neural network. *Biosystems Engineering*, *151*, 72–80.

Dyrmann, M., Mortensen, A. K., Midtiby, H. S., & Jørgensen, R. N. (2016). Pixel-wise classification of weeds and crops in images by using a fully convolutional neural network. *Proceedings of the International Conference on Agricultural Engineering, Aarhus, Denmark*, 26–29.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Farjon, G., Krikeb, O., Hillel, A. B., & Alchanatis, V. (2019). Detection and counting of flowers on apple trees for better chemical thinning decisions. *Precision Agriculture*.

Faust, O., Hagiwara, Y., Hong, T. J., Lih, O. S., & Acharya, U. R. (2018). Deep learning for healthcare applications based on physiological signals: A review. *Computer Methods and Programs in Biomedicine*, *161*, 1–13.

Feng, Y., Peng, Y., Cui, N., Gong, D., & Zhang, K. (2017). Modeling reference evapotranspiration using extreme learning machine and generalized regression neural network only with temperature data. *Computers and Electronics in Agriculture*, *136*, 71–78.

Floyd, F., & Sabins, J. R. (1987). Remote sensing principles and interpretation. *2nd, Edition*, 1–12.

Girshick, R. (2015). Fast r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 1440–1448.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 580–587.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 142–158. https://doi.org/10.1109/TPAMI.2015.2437384

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., & Lewis, K. (2015). Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, *116*, 8–19.

Gonzalo, M. J., Brewer, M. T., Anderson, C., Sullivan, D., Gray, S., & van der Knaap, E. (2009). Tomato fruit shape analysis using morphometric and morphology attributes implemented in Tomato Analyzer software program. *Journal of the American Society for Horticultural Science*, *134*(1), 77–87.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Grinblat, G. L., Uzal, L. C., Larese, M. G., & Granitto, P. M. (2016). Deep learning for plant identification using vein morphological patterns. *Computers and Electronics in Agriculture*, *127*, 418–424.

Groover, M. P. (2007). *Automation, production systems, and computer-integrated manufacturing*. Prentice Hall Press.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, *187*, 27–48.

Gustafsson, H., Claesson, I., & Nordholm, S. (2001). *Signal noise reduction by spectral subtraction using linear convolution and casual filtering*. Google Patents.

Habaragamuwa, H., Ogawa, Y., Suzuki, T., Shiigi, T., Ono, M., & Kondo, N. (2018). Detecting greenhouse strawberries (mature and immature), using deep convolutional neural network. *Engineering in Agriculture, Environment and Food*, *11*(3), 127–138.

Hall, D., McCool, C., Dayoub, F., Sunderhauf, N., & Upcroft, B. (2015). Evaluation of features for leaf classification in challenging conditions. *2015 IEEE Winter Conference on Applications of Computer Vision*, 797–804.

Han, J., & Moraga, C. (1995). The influence of the sigmoid function parameters on the speed of backpropagation learning. *International Workshop on Artificial Neural Networks*, 195–201.

Hansen, M. F., Smith, M. L., Smith, L. N., Salter, M. G., Baxter, E. M., Farish, M., & Grieve, B. (2018). Towards on-farm pig face recognition using convolutional neural networks. *Computers in Industry*, *98*, 145–152.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 2961–2969.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hemming, C. (2003). Using neural networks in linguistic resources. *Department of Languages, University College of Skövde, Swedish National Graduate School of Language Technology*.

Huang, T. S. (1996). Computer Vision: Evolution and Promise. *University of Illinois*, 1–3. Retrieved from http://cds.cern.ch/record/400313/files/p21.pdf

Huh, M., Agrawal, P., & Efros, A. A. (2016). What makes ImageNet good for transfer learning? *ArXiv Preprint ArXiv:1608.08614*.

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, (3), 31–44.

Janocha, K., & Czarnecki, W. M. (2017). On loss functions for deep neural networks in classification. *ArXiv Preprint ArXiv:1702.05659*.

Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A., & Klapp, I. (2019). Estimating melon yield for breeding processes by machine-vision processing of UAV images. In *Precision agriculture'19* (pp. 1386–1393). Wageningen Academic Publishers.

Kamilaris, A., & Prenafeta-Boldú., F. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*(April), 70–90. https://doi.org/10.1016/j.compag.2018.02.016

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). A review of the use of convolutional neural networks in agriculture. *The Journal of Agricultural Science*, *156*(3), 312–322. https://doi.org/10.1017/S0021859618000436

Kapach, K., Barnea, E., Mairon, R., Edan, Y., & Ben-Shahar, O. (2012). Computer vision for fruit harvesting robots--state of the art and challenges ahead. *International Journal of Computational Vision and Robotics*, *3*(1/2), 4–34.

Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F., & Boussaid, F. (2018). Computer Vision for Human--Machine Interaction. In *Computer Vision for Assistive Healthcare* (pp. 127–145). Elsevier.

Kestur, R., Meduri, A., & Narasipura, O. (2019). MangoNet: A deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard. *Engineering Applications of Artificial Intelligence*, *77*, 59–69.

Kim, S., & Casper, R. (2013). Applications of convolution in image processing with MATLAB. *University of Washington*, 1–20.

Klette, R. (2014). *Concise computer vision*. Springer.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019a). Deep learning--Method overview and review of use for fruit detection and yield estimation. *Computers and Electronics in Agriculture*, *162*, 219–234.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019b). Deep learning for real-time fruit detection and orchard fruit load estimation: benchmarking of ‘MangoYOLO.’ *Precision Agriculture*, (0123456789).

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019c). Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of ‘MangoYOLO.’ *Precision Agriculture*, 1–29.

Kon, M. A., & Plaskota, L. (2000). Information complexity of neural networks. *Neural Networks*, *13*(3), 365–375.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Krogh, A., & Hertz, J. A. (1992). A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems*, 950–957.

Kumar, A., Kaur, A., & Kumar, M. (2019). Face detection techniques: a review. *Artificial Intelligence Review*, *52*(2), 927–948.

Kung, H. Y., Kuo, T. H., Chen, C. H., & Tsai, P. Y. (2016). Accuracy analysis mechanism for agriculture data using the ensemble neural network method. *Sustainability (Switzerland)*, *8*(8), 1–11. https://doi.org/10.3390/su8080735

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, A. (2017). Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778–782.

Labelimg, graphical image annotation tool. (2018). Retrieved from https://github.com/tzutalin/labelImg

Lamb, N., & Chuah, M. C. (2018). A Strawberry Detection System Using Convolutional Neural Networks. *2018 IEEE International Conference on Big Data (Big Data)*, 2515–2520.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., Jackel, L. D., Bottou, L., Cortes, C., Denker, J. S., Drucker, H., … others. (1995). Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural Networks: The Statistical Mechanics Perspective*, *261*, 276.

Lee, D. S. (1988). Neural network models and their application to handwritten digit recognition. *IEEE 1988 International Conference on Neural Networks*, 63–70.

Lee, S. H., Chan, C. S., Wilkin, P., & Remagnino, P. (2015). Deep-plant: Plant identification with convolutional neural networks. *2015 IEEE International Conference on Image Processing (ICIP)*, 452–456.

Lei, H., Han, T., Zhou, F., Yu, Z., Qin, J., Elazab, A., & Lei, B. (2018). A deeply supervised residual network for HEp-2 cell classification via cross-modal transfer learning. *Pattern Recognition*, *79*, 290–302.

Li, Z., Zhang, X., Müller, H., & Zhang, S. (2018). Large-scale retrieval for medical image analytics: A comprehensive review. *Medical Image Analysis*, *43*, 66–84.

Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: A review. *Sensors (Switzerland)*, *18*(8), 1–29.

Liang, Q., Zhu, W., Long, J., Wang, Y., Sun, W., & Wu, W. (2018). A Real-Time Detection Framework for On-Tree Mango Based on SSD Network. *International Conference on Intelligent Robotics and Applications*, 423–436.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2117–2125.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision*, 2980–2988.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., … Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *European Conference on Computer Vision*, 740–755.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., … Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016).

Ssd: Single shot multibox detector. *European Conference on Computer Vision*, 21–37.

Lobell, D. B., Cassman, K. G., & Field, C. B. (2009). Crop yield gaps: their importance, magnitudes, and causes. *Annual Review of Environment and Resources*, *34*, 179–204.

Mahmood, A., Bennamoun, M., An, S., Sohel, F., Boussaid, F., Hovey, R., … Fisher, R. B. (2017). Deep learning for coral classification. In *Handbook of Neural Computation* (pp. 383–401). Elsevier.

Mazzini, D., Buzzelli, M., Pauy, D. Pietro, & Schettini, R. (2018). A CNN architecture for efficient semantic segmentation of street scenes. *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, 1–6.

Mehdizadeh, S., Behmanesh, J., & Khalili, K. (2017). Using MARS, SVM, GEP and empirical equations for estimation of monthly mean reference evapotranspiration. *Computers and Electronics in Agriculture*, *139*, 103–114.

Miles, J. (2014). R squared, adjusted R squared. *Wiley StatsRef: Statistics Reference Online*.

Milioto, A., Lottes, P., & Stachniss, C. (2017). Real-time blob-wise sugar beets vs weeds classification for monitoring fields using convolutional neural networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*, 41.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*, 1419.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John wiley & sons.

Morellos, A., Pantazi, X.-E., Moshou, D., Alexandridis, T., Whetton, R., Tziotzios, G., … Mouazen, A. M. (2016). Machine learning based prediction of soil total nitrogen, organic carbon and moisture content by using VIS-NIR spectroscopy. *Biosystems Engineering*, *152*, 104–116.

Morris, T. (2004). *Computer Vision and Image Processing (Cornerstones of Computing)*. Palgrave Macmillan Limited.

Mortensen, A. K., Dyrmann, M., Karstoft, H., Jørgensen, R. N., Gislum, R., & others. (2016). Semantic segmentation of mixed crops using deep convolutional neural network. *CIGR-AgEng Conference, 26-29 June 2016, Aarhus, Denmark. Abstracts and Full Papers*, 1–6.

Nahvi, B., Habibi, J., Mohammadi, K., Shamshirband, S., & Al Razgan, O. S. (2016). Using self-adaptive evolutionary algorithm to improve the performance of an extreme learning machine for estimating soil temperature. *Computers and Electronics in Agriculture*, *124*, 150–160.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 807–814.

Neubeck, A., & Van Gool, L. (2006). Efficient non-maximum suppression. *18th International Conference on Pattern Recognition (ICPR'06)*, *3*, 850–855.

Nguyen, V. N., Jenssen, R., & Roverso, D. (2018). Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning. *International Journal of Electrical Power & Energy Systems*, *99*, 107–120.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *ArXiv Preprint ArXiv:1811.03378*.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), 62–66.

Pai, A., Teng, Y.-C., Blair, J., Kallenberg, M., Dam, E. B., Sommer, S., … Nielsen, M. (2017). Characterization of Errors in Deep Learning-based Brain MRI Segmentation. In *Deep Learning for Medical Image Analysis* (pp. 223–242). Elsevier.

Pantazi, X. E., Moshou, D., Alexandridis, T., Whetton, R. L., & Mouazen, A. M. (2016). Wheat yield prediction using machine learning and advanced sensing techniques. *Computers and Electronics in Agriculture*, *121*, 57–65.

Patricio, D. I., & Rieder, R. (2018). Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. *Computers and Electronics in Agriculture*, *153*, 69–81.

Pereira, C. S., Morais, R., & Reis, M. J. C. S. (2017). Recent advances in image processing techniques for automated harvesting purposes: a review. *2017 Intelligent Systems Conference (IntelliSys)*, 566–575.

Potena, C., Nardi, D., & Pretto, A. (2016). Fast and accurate crop and weed identification with summarized train sets for precision agriculture. *International Conference on Intelligent Autonomous Systems*, 105–121.

Pound, M. P., Atkinson, J. A., Townsend, A. J., Wilson, M. H., Griffiths, M., Jackson, A. S., … others. (2017). Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *Gigascience*, *6*(10), gix083.

Pound, M. P., Atkinson, J. A., Wells, D. M., Pridmore, T. P., & French, A. P. (2017). Deep learning for multi-task plant phenotyping. *Proceedings of the IEEE International Conference on Computer Vision*, 2055–2063.

Qiao, Y., Truman, M., & Sukkarieh, S. (2019). Cattle segmentation and contour extraction based on Mask R-CNN for precision livestock farming. *Computers and Electronics in Agriculture*, *165*, 104958.

Rahman, M., Robson, A., & Bristow, M. (2018). Exploring the Potential of High Resolution WorldView-3 Imagery for Estimating Yield of Mango. *Remote Sensing*, *10*(12), 1866.

Rahnemoonfar, M., & Sheppard, C. (2017a). Deep count: fruit counting based on deep simulated learning. *Sensors*, *17*(4), 905.

Rahnemoonfar, M., & Sheppard, C. (2017b). Real-time yield estimation based on deep learning. *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, *10218*, 1021809.

Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, *29*(9), 2352–2449.

Rebetez, J., Satizábal, H. F., Mota, M., Noll, D., Büchi, L., Wendling, M., … Burgos, S. (2016). Augmenting a convolutional neural network with local histograms-A case study in crop classification from high-resolution UAV imagery. *ESANN*.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 91–99.

Reyes, A. K., Caicedo, J. C., & Camargo, J. E. (2015). Fine-tuning Deep Convolutional Networks for Plant Recognition. *CLEF (Working Notes)*, *1391*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234–241.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *ArXiv Preprint ArXiv:1609.04747*.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, *16*(8), 1222.

Sarig, Y. (1993). Robotics of fruit harvesting: A state-of-the-art review. *Journal of Agricultural Engineering Research*, *54*(4), 265–280.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *ArXiv Preprint ArXiv:1312.6229*.

Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 806–813.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*, *2016*.

Sonka, M., Hlavac, V., & Boyle, R. (2014). *Image processing, analysis, and machine vision*. Cengage Learning.

Sørensen, R. A., Rasmussen, J., Nielsen, J., & Jørgensen, R. N. (2017). Thistle detection using convolutional neural networks. *2017 EFITA WCCA CONGRESS*, 161.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stajnko, D., Rakun, J., & Blanke, M. (2009). Modelling apple fruit yield using image analysis for fruit colour, shape and texture. *European Journal of Horticultural Science*, *74*(6), 260.

Stein, M., Bargoti, S., & Underwood, J. (2016). Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors (Switzerland)*, *16*(11). https://doi.org/10.3390/s16111915

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Retrieved from https://books.google.co.il/books?id=bXzAlkODwa8C

Tao, Y., Zhou, J., Wang, K., & Shen, W. (2018). Rapid detection of fruits in orchard scene based on deep neural network. *2018 ASABE Annual International Meeting*, 1.

Tillett, R. D. (1991). Image analysis for agricultural processes: a review of potential opportunities. *Journal of Agricultural Engineering Research*, *50*(C), 247–258. https://doi.org/10.1016/S0021-8634(05)80018-6

Uijlings, J. R. R., Van De Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, *104*(2), 154–171.

van't Ooster, A., Bontsema, J., van Henten, E. J., & Hemming, S. (2014). Simulation of harvest operations in a static rose cultivation system. *Biosystems Engineering*, *120*, 34–46.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, *1*, 511–518.

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, *2018*.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., … Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1386–1393.

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). Cnn-rnn: A unified framework for multi-label image classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2285–2294.

Wang, X., & Schneider, J. (2014). Flexible Transfer Learning under Support and Model Shift. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 1898–1906). Curran Associates, Inc.

Wang, Y.-Q. (2014). An analysis of the Viola-Jones face detection algorithm. *Image Processing On Line*, *4*, 128–148.

Wijewickrema, S. N. R., & Paplinski, A. P. (2005). Principal component analysis for the approximation of a fruit as an ellipse. *Full Papers/WSCG*.

Xinshao, W., & Cheng, C. (2015). Weed seeds classification based on PCANet deep learning baseline. *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 408–415.

XIONG, J., LIU, Z., TANG, L., LIN, R., BU, R., & PENG, H. (2018). Visual Detection Technology of Green Citrus under Natural Environment. *Transactions of the Chinese Society for Agricultural Machinery*, (4), 5.

Yalcin, H. (2017). Plant phenology recognition using deep learning: Deep-Pheno. *2017 6th International Conference on Agro-Geoinformatics*, 1–5.

Yao, G., Lei, T., & Zhong, J. (2019). A review of Convolutional-Neural-Network-based action recognition. *Pattern Recognition Letters*, *118*, 14–22.

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 3320–3328). Curran Associates, Inc.

Zhang, C., & Kovacs, J. M. (2012). The application of small unmanned aerial systems for precision agriculture: a review. *Precision Agriculture*, *13*(6), 693–712. https://doi.org/10.1007/s11119-012-9274-5

# 6. Summary, conclusions and future research

A system for detection and yield estimation of melons from top view UAV images of a melon field has been developed. The system includes three main stages: melon detection, geometric feature extraction, and individual melon yield estimation. The system provides an output that includes the estimated weight of each detected melon in the image along with localization and position (angle) of each melon. The system sums all the estimated melons weights and provides the overall weight yield of the field.

Initially, a hybrid method classical/CNN algorithm was suggested. A pipeline composed from three main stages was developed; it contains melon recognition based on Viola-Jones cascade and CNN methods, followed by feature extraction by using homogeneity cost function and yield estimation using linear regression based on the extracted features.

To train the cascade object detector, 3031 negative training images (of background) and 1933 positive training images (of melons) were created. CNN training was based on 1383 of the positive, and 1349 of the negative training images. All melons were randomly selected. Evaluation for melon detection was done on a total of 1056 melons from one image from 2016 season and one image from 2017 season.

The initial hybrid system achieved an average precision of 0.82 and F1 score of 0.85 for detection and more than 4% yield estimation error. Analyzing the drawbacks of the first method, a second improved method which included improved stages of object detection and image extraction was suggested.

The detection process of melons was based on a pre-defined RetinaNet deep convolutional neural network, it was trained using transfer learning to deal well with detecting small objects in high resolution images such as melons from top view UAV images. The detection process achieved an overall average precision score of 0.92, and a F1-score of 0.9 for different agriculture environments. The network was trained with four images including 4220 labeled melons from the 2018 season. During the training process, an augmentation technique was used in order to increase the variety of melons images. Evaluation for melon detection was done on four different images from different seasons and agriculture environments, with a total of 2,394 melons. Every melon which is detected by the network, is labeled using an anchor box. For each anchor box a feature extraction process that included two substages was applied. First, with an active contour method of Chan Vese an initial closed form of the melon was identified. Then, the closed form was presented with binary mask in order to allow full identification of the melon contour. in the second substage, a PCA method for ellipse fitting was perform on the binary mask. The melon contour features were extracted including the location and the position of the melon, minor and major axes size of the melon provided in terms of pixels. At the final stage, an individual melon yield estimation is provided. The weight of each melon is predicted using the feature size of the melon in a regression model trained to estimate the weight of a single melon. The weight estimation of a single melon measured by the MAPE index achieved 16% error. This error can be reduced to 12% with more accurate geometrical feature extraction. However, yield estimation of the sum of pf all melon weights in the field achieved deviation from the total actual yield estimation of only 3% underestimation. Hence, the system provides promising results.

In conclusion, both systems were based on the same main three stages. However, performance results were different since they included different methods. The second system outperformed the first system in every aspect, closing most of the gaps that were identified during the first system analysis. This was reflected in the first stage where the performance of the melon detection and localization yielded better result. The RetinaNet together with NMS reached 10% better results in terms of average precision as compared to the Viola-Jones and CNN configuration. A better localization allows to perform more accurate feature extraction using the Chan Vese active contour algorithm and PCA ellipse. This results in a better yield estimation result for the all crop.

To improve the existing system, future work should apply more advanced feature extraction algorithms. Since the detection algorithm performed well, the Chan Vese method should be replaced perhaps with a CNN which can provide local segmentation of the melon (e.g., the U-net (Ronneberger et al., 2015)). Another direction for implementation can be to create a system based on CNN which performs an instance segmentation, like Mask R-CNN (He et al., 2017), and on top of the segmentation add another layer which regresses the weight of the melon fruit. However, both methods suggested above require to label images at the pixel level, which is considered a tedious task and imposes a major limitation towards implementation.

# Bibliography

Agrawal, P., Girshick, R., & Malik, J. (2014). Analyzing the performance of multilayer neural networks for object recognition. *European Conference on Computer Vision*, 329–344.

Ali, I., Cawkwell, F., Dwyer, E., & Green, S. (2016). Modeling managed grassland biomass estimation by using multitemporal remote sensing data—A machine learning approach. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *10*(7), 3254–3264.

Amara, J., Bouaziz, B., & Algergawy, A. (2017). A Deep Learning-based Approach for Banana Leaf Diseases Classification. *BTW (Workshops)*, 79–88.

Anwar, S. M., Majid, M., Qayyum, A., Awais, M., Alnowami, M., & Khan, M. K. (2018). Medical image analysis using convolutional neural networks: a review. *Journal of Medical Systems*, *42*(11), 226.

Bac, C. W., van Henten, E. J., Hemming, J., & Edan, Y. (2014). Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *Journal of Field Robotics*, *31*(6), 888–911.

Bargoti, S., & Underwood, J. (2017a). Deep fruit detection in orchards. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 3626–3633.

Bargoti, S., & Underwood, J. P. (2017b). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, *34*(6), 1039–1060.

Bergerman, M., Billingsley, J., Reid, J., & van Henten, E. (2016). Robotics in agriculture and forestry. In *Springer handbook of robotics* (pp. 1463–1492). Springer.

Boureau, Y.-L., Ponce, J., & LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 111–118.

Bresilla, K., Perulli, G. D., Boini, A., Morandi, B., Grappadelli, L. C., & Manfrini, L. (2019). Single-shot convolution neural networks for real-time fruit detection within the tree. *Frontiers in Plant Science*, *10*.

Burrus, C. S., & Parks, T. W. (1985). *and Convolution Algorithms*. Citeseer.

Calixto, R. R., Neto, L. G. P., da Silveira Cavalcante, T., Aragão, M. F., & de Oliveira Silva, E. (2019). A computer vision model development for size and weight estimation of yellow melon in the Brazilian northeast. *Scientia Horticulturae*, 108521.

Carrio, A., Sampedro, C., Rodriguez-Ramos, A., & Campoy, P. (2017). A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*, *2017*.

Chan, T., & Vese, L. (1999). An active contour model without edges. *International Conference on Scale-Space Theories in Computer Vision*, 141–151.

Chen, S. W., Shivakumar, S. S., Dcunha, S., Das, J., Okon, E., Qu, C., & Kumar, V. (2017). Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robotics and Automation Letters*, *2*(2), 781–788.

Cheng, G., & Han, J. (2016). A survey on object detection in optical remote sensing images. *ISPRS Journal of Photogrammetry and Remote Sensing*, *117*, 11–28.

Cheng, H., Damerow, L., Sun, Y., & Blanke, M. (2017). Early yield prediction using image analysis of apple fruit and tree canopy features with neural networks. *Journal of Imaging*, *3*(1), 6.

Cohen, T., & Welling, M. (2016). Group equivariant convolutional networks. *International Conference on Machine Learning*, 2990–2999.

Dahl, G. E., Sainath, T. N., & Hinton, G. E. (2013). Improving deep neural networks for LVCSR using rectified linear units and dropout. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 8609–8613.

Dashuta, A., & Klapp, I. (2018). Melon Recognition in UAV Images to Estimate Yield of a Breeding Process. *Optics and Photonics for Energy and the Environment*, ET4A--2.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. *International Conference on Machine Learning*, 647–655.

Dyrmann, M., Jørgensen, R. N., & Midtiby, H. S. (2017). RoboWeedSupport-Detection of weed locations in leaf occluded cereal crops using a fully convolutional neural network. *Advances in Animal Biosciences*, *8*(2), 842–847.

Dyrmann, M., Karstoft, H., & Midtiby, H. S. (2016). Plant species classification using deep convolutional neural network. *Biosystems Engineering*, *151*, 72–80.

Dyrmann, M., Mortensen, A. K., Midtiby, H. S., & Jørgensen, R. N. (2016). Pixel-wise classification of weeds and crops in images by using a fully convolutional neural network. *Proceedings of the International Conference on Agricultural Engineering, Aarhus, Denmark*, 26–29.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Farjon, G., Krikeb, O., Hillel, A. B., & Alchanatis, V. (2019). Detection and counting of flowers on apple trees for better chemical thinning decisions. *Precision Agriculture*.

Faust, O., Hagiwara, Y., Hong, T. J., Lih, O. S., & Acharya, U. R. (2018). Deep learning for healthcare applications based on physiological signals: A review. *Computer Methods and Programs in Biomedicine*, *161*, 1–13.

Feng, Y., Peng, Y., Cui, N., Gong, D., & Zhang, K. (2017). Modeling reference evapotranspiration using extreme learning machine and generalized regression neural network only with temperature data. *Computers and Electronics in*

*Agriculture*, *136*, 71–78.

Floyd, F., & Sabins, J. R. (1987). Remote sensing principles and interpretation. *2nd, Edition*, 1–12.

Girshick, R. (2015). Fast r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 1440–1448.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 580–587.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2016). Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 142–158. https://doi.org/10.1109/TPAMI.2015.2437384

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., & Lewis, K. (2015). Sensors and systems for fruit detection and localization: A review. *Computers and Electronics in Agriculture*, *116*, 8–19.

Gonzalo, M. J., Brewer, M. T., Anderson, C., Sullivan, D., Gray, S., & van der Knaap, E. (2009). Tomato fruit shape analysis using morphometric and morphology attributes implemented in Tomato Analyzer software program. *Journal of the American Society for Horticultural Science*, *134*(1), 77–87.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Grinblat, G. L., Uzal, L. C., Larese, M. G., & Granitto, P. M. (2016). Deep learning for plant identification using vein morphological patterns. *Computers and Electronics in Agriculture*, *127*, 418–424.

Groover, M. P. (2007). *Automation, production systems, and computer-integrated manufacturing*. Prentice Hall Press.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., & Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, *187*, 27–48.

Gustafsson, H., Claesson, I., & Nordholm, S. (2001). *Signal noise reduction by spectral subtraction using linear convolution and casual filtering*. Google Patents.

Habaragamuwa, H., Ogawa, Y., Suzuki, T., Shiigi, T., Ono, M., & Kondo, N. (2018). Detecting greenhouse strawberries (mature and immature), using deep convolutional neural network. *Engineering in Agriculture, Environment and Food*, *11*(3), 127–138.

Hall, D., McCool, C., Dayoub, F., Sunderhauf, N., & Upcroft, B. (2015). Evaluation of features for leaf classification in challenging conditions. *2015 IEEE Winter Conference on Applications of Computer Vision*, 797–804.

Han, J., & Moraga, C. (1995). The influence of the sigmoid function parameters on the speed of backpropagation learning. *International Workshop on Artificial Neural Networks*, 195–201.

Hansen, M. F., Smith, M. L., Smith, L. N., Salter, M. G., Baxter, E. M., Farish, M., & Grieve, B. (2018). Towards on-farm pig face recognition using convolutional neural networks. *Computers in Industry*, *98*, 145–152.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, 2961–2969.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hemming, C. (2003). Using neural networks in linguistic resources. *Department of Languages, University College of Skövde, Swedish National Graduate School of Language Technology*.

Huang, T. S. (1996). Computer Vision: Evolution and Promise. *University of Illinois*, 1–3. Retrieved from http://cds.cern.ch/record/400313/files/p21.pdf

Huh, M., Agrawal, P., & Efros, A. A. (2016). What makes ImageNet good for transfer learning? *ArXiv Preprint ArXiv:1608.08614*.

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, (3), 31–44.

Janocha, K., & Czarnecki, W. M. (2017). On loss functions for deep neural networks in classification. *ArXiv Preprint ArXiv:1702.05659*.

Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A., & Klapp, I. (2019). Estimating melon yield for breeding processes by machine-vision processing of UAV images. In *Precision agriculture'19* (pp. 1386–1393). Wageningen Academic Publishers.

Kamilaris, A., & Prenafeta-Boldú., F. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, *147*(April), 70–90. https://doi.org/10.1016/j.compag.2018.02.016

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). A review of the use of convolutional neural networks in agriculture. *The Journal of Agricultural Science*, *156*(3), 312–322. https://doi.org/10.1017/S0021859618000436

Kapach, K., Barnea, E., Mairon, R., Edan, Y., & Ben-Shahar, O. (2012). Computer vision for fruit harvesting robots--state of the art and challenges ahead. *International Journal of Computational Vision and Robotics*, *3*(1/2), 4–34.

Ke, Q., Liu, J., Bennamoun, M., An, S., Sohel, F., & Boussaid, F. (2018). Computer Vision for Human--Machine Interaction. In *Computer Vision for Assistive Healthcare* (pp. 127–145). Elsevier.

Kestur, R., Meduri, A., & Narasipura, O. (2019). MangoNet: A deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard. *Engineering Applications of Artificial Intelligence*, *77*, 59–69.

Kim, S., & Casper, R. (2013). Applications of convolution in image processing with

MATLAB. *University of Washington*, 1–20.

Klette, R. (2014). *Concise computer vision*. Springer.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019a). Deep learning--Method overview and review of use for fruit detection and yield estimation. *Computers and Electronics in Agriculture*, *162*, 219–234.

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019b). Deep learning for real-time fruit detection and orchard fruit load estimation: benchmarking of 'MangoYOLO.' *Precision Agriculture*, (0123456789).

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C. (2019c). Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO.' *Precision Agriculture*, 1–29.

Kon, M. A., & Plaskota, L. (2000). Information complexity of neural networks. *Neural Networks*, *13*(3), 365–375.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Krogh, A., & Hertz, J. A. (1992). A simple weight decay can improve generalization. *Advances in Neural Information Processing Systems*, 950–957.

Kumar, A., Kaur, A., & Kumar, M. (2019). Face detection techniques: a review. *Artificial Intelligence Review*, *52*(2), 927–948.

Kung, H. Y., Kuo, T. H., Chen, C. H., & Tsai, P. Y. (2016). Accuracy analysis mechanism for agriculture data using the ensemble neural network method. *Sustainability (Switzerland)*, *8*(8), 1–11. https://doi.org/10.3390/su8080735

Kussul, N., Lavreniuk, M., Skakun, S., & Shelestov, A. (2017). Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, *14*(5), 778–782.

Labelimg, graphical image annotation tool. (2018). Retrieved from https://github.com/tzutalin/labelImg

Lamb, N., & Chuah, M. C. (2018). A Strawberry Detection System Using Convolutional Neural Networks. *2018 IEEE International Conference on Big Data (Big Data)*, 2515–2520.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., Jackel, L. D., Bottou, L., Cortes, C., Denker, J. S., Drucker, H., & Vapnik, V. (1995). Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural Networks: The Statistical Mechanics Perspective*, *261*, 276.

Lee, D. S. (1988). Neural network models and their application to handwritten digit recognition. *IEEE 1988 International Conference on Neural Networks*, 63–70.

Lee, S. H., Chan, C. S., Wilkin, P., & Remagnino, P. (2015). Deep-plant: Plant identification with convolutional neural networks. *2015 IEEE International Conference on Image Processing (ICIP)*, 452–456.

Lei, H., Han, T., Zhou, F., Yu, Z., Qin, J., Elazab, A., & Lei, B. (2018). A deeply supervised residual network for HEp-2 cell classification via cross-modal transfer learning. *Pattern Recognition*, *79*, 290–302.

Li, Z., Zhang, X., Müller, H., & Zhang, S. (2018). Large-scale retrieval for medical image analytics: A comprehensive review. *Medical Image Analysis*, *43*, 66–84.

Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: A review. *Sensors (Switzerland)*, *18*(8), 1–29.

Liang, Q., Zhu, W., Long, J., Wang, Y., Sun, W., & Wu, W. (2018). A Real-Time Detection Framework for On-Tree Mango Based on SSD Network. *International Conference on Intelligent Robotics and Applications*, 423–436.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2117–2125.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision*, 2980–2988.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *European Conference on Computer Vision*, 740–755.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. *European Conference on Computer Vision*, 21–37.

Lobell, D. B., Cassman, K. G., & Field, C. B. (2009). Crop yield gaps: their importance, magnitudes, and causes. *Annual Review of Environment and Resources*, *34*, 179–204.

Mahmood, A., Bennamoun, M., An, S., Sohel, F., Boussaid, F., Hovey, R., & Fisher, R. B. (2017). Deep learning for coral classification. In *Handbook of Neural Computation* (pp. 383–401). Elsevier.

Mazzini, D., Buzzelli, M., Pauy, D. Pietro, & Schettini, R. (2018). A CNN architecture for efficient semantic segmentation of street scenes. *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, 1–6.

Mehdizadeh, S., Behmanesh, J., & Khalili, K. (2017). Using MARS, SVM, GEP and empirical equations for estimation of monthly mean reference evapotranspiration. *Computers and Electronics in Agriculture*, *139*, 103–114.

Miles, J. (2014). R squared, adjusted R squared. *Wiley StatsRef: Statistics Reference Online*.

Milioto, A., Lottes, P., & Stachniss, C. (2017). Real-time blob-wise sugar beets vs weeds classification for monitoring fields using convolutional neural networks. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*, 41.

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*, 1419.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John Wiley & Sons.

Morellos, A., Pantazi, X.-E., Moshou, D., Alexandridis, T., Whetton, R., Tziotzios, G., & Mouazen, A. M. (2016). Machine learning based prediction of soil total nitrogen, organic carbon and moisture content by using VIS-NIR spectroscopy. *Biosystems Engineering*, *152*, 104–116.

Morris, T. (2004). *Computer Vision and Image Processing (Cornerstones of Computing)*. Palgrave Macmillan Limited.

Mortensen, A. K., Dyrmann, M., Karstoft, H., Jørgensen, R. N., & Gislum, R. (2016). Semantic segmentation of mixed crops using deep convolutional neural network. *CIGR-AgEng Conference, 26-29 June 2016, Aarhus, Denmark. Abstracts and Full Papers*, 1–6.

Nahvi, B., Habibi, J., Mohammadi, K., Shamshirband, S., & Al Razgan, O. S. (2016). Using self-adaptive evolutionary algorithm to improve the performance of an extreme learning machine for estimating soil temperature. *Computers and Electronics in Agriculture*, *124*, 150–160.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 807–814.

Neubeck, A., & Van Gool, L. (2006). Efficient non-maximum suppression. *18th International Conference on Pattern Recognition (ICPR'06)*, *3*, 850–855.

Nguyen, V. N., Jenssen, R., & Roverso, D. (2018). Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning. *International Journal of Electrical Power & Energy Systems*, *99*, 107–120.

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *ArXiv Preprint ArXiv:1811.03378*.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), 62–66.

Pai, A., Teng, Y.-C., Blair, J., Kallenberg, M., Dam, E. B., Sommer, S., & Nielsen, M. (2017). Characterization of Errors in Deep Learning-based Brain MRI Segmentation. In *Deep Learning for Medical Image Analysis* (pp. 223–242). Elsevier.

Pantazi, X. E., Moshou, D., Alexandridis, T., Whetton, R. L., & Mouazen, A. M. (2016). Wheat yield prediction using machine learning and advanced sensing techniques. *Computers and Electronics in Agriculture*, *121*, 57–65.

Patricio, D. I., & Rieder, R. (2018). Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. *Computers and Electronics in Agriculture*, *153*, 69–81.

Pereira, C. S., Morais, R., & Reis, M. J. C. S. (2017). Recent advances in image processing techniques for automated harvesting purposes: a review. *2017 Intelligent Systems Conference (IntelliSys)*, 566–575.

Potena, C., Nardi, D., & Pretto, A. (2016). Fast and accurate crop and weed identification with summarized train sets for precision agriculture. *International Conference on Intelligent Autonomous Systems*, 105–121.

Pound, M. P., Atkinson, J. A., Townsend, A. J., Wilson, M. H., Griffiths, M., Jackson, A. S., … others. (2017). Deep machine learning provides state-of-the-art performance in image-based plant phenotyping. *Gigascience*, *6*(10), gix083.

Pound, M. P., Atkinson, J. A., Wells, D. M., Pridmore, T. P., & French, A. P. (2017). Deep learning for multi-task plant phenotyping. *Proceedings of the IEEE International Conference on Computer Vision*, 2055–2063.

Qiao, Y., Truman, M., & Sukkarieh, S. (2019). Cattle segmentation and contour extraction based on Mask R-CNN for precision livestock farming. *Computers and Electronics in Agriculture*, *165*, 104958.

Rahman, M., Robson, A., & Bristow, M. (2018). Exploring the Potential of High Resolution WorldView-3 Imagery for Estimating Yield of Mango. *Remote Sensing*, *10*(12), 1866.

Rahnemoonfar, M., & Sheppard, C. (2017a). Deep count: fruit counting based on deep simulated learning. *Sensors*, *17*(4), 905.

Rahnemoonfar, M., & Sheppard, C. (2017b). Real-time yield estimation based on deep learning. *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, *10218*, 1021809.

Rawat, W., & Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, *29*(9), 2352–2449.

Rebetez, J., Satizábal, H. F., Mota, M., Noll, D., Büchi, L., Wendling, M., & Burgos, S. (2016). Augmenting a convolutional neural network with local histograms-A case study in crop classification from high-resolution UAV imagery. *in ESANN*.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer*

*Vision and Pattern Recognition*, 779–788.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 91–99.

Reyes, A. K., Caicedo, J. C., & Camargo, J. E. (2015). Fine-tuning Deep Convolutional Networks for Plant Recognition. *CLEF (Working Notes)*, *1391*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234–241.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *ArXiv Preprint ArXiv:1609.04747*.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, *16*(8), 1222.

Sarig, Y. (1993). Robotics of fruit harvesting: A state-of-the-art review. *Journal of Agricultural Engineering Research*, *54*(4), 265–280.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *ArXiv Preprint ArXiv:1312.6229*.

Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: an astounding baseline for recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 806–813.

Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*, *2016*.

Sonka, M., Hlavac, V., & Boyle, R. (2014). *Image processing, analysis, and machine vision*. Cengage Learning.

Sørensen, R. A., Rasmussen, J., Nielsen, J., & Jørgensen, R. N. (2017). Thistle detection using convolutional neural networks. *2017 EFITA WCCA CONGRESS*, 161.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stajnko, D., Rakun, J., & Blanke, M. (2009). Modelling apple fruit yield using image analysis for fruit colour, shape and texture. *European Journal of Horticultural Science*, *74*(6), 260.

Stein, M., Bargoti, S., & Underwood, J. (2016). Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors*

*(Switzerland)*, *16*(11). https://doi.org/10.3390/s16111915

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., … Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Retrieved from https://books.google.co.il/books?id=bXzAlkODwa8C

Tao, Y., Zhou, J., Wang, K., & Shen, W. (2018). Rapid detection of fruits in orchard scene based on deep neural network. *2018 ASABE Annual International Meeting*, 1.

Tillett, R. D. (1991). Image analysis for agricultural processes: a review of potential opportunities. *Journal of Agricultural Engineering Research*, *50*(C), 247–258. https://doi.org/10.1016/S0021-8634(05)80018-6

Uijlings, J. R. R., Van De Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective search for object recognition. *International Journal of Computer Vision*, *104*(2), 154–171.

van't Ooster, A., Bontsema, J., van Henten, E. J., & Hemming, S. (2014). Simulation of harvest operations in a static rose cultivation system. *Biosystems Engineering*, *120*, 34–46.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, *1*, 511–518.

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, *2018*.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., & Wu, Y. (2014). Learning fine-grained image similarity with deep ranking. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1386–1393.

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). Cnn-rnn: A unified framework for multi-label image classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2285–2294.

Wang, X., & Schneider, J. (2014). Flexible Transfer Learning under Support and Model Shift. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 1898–1906). Curran Associates, Inc.

Wang, Y.-Q. (2014). An analysis of the Viola-Jones face detection algorithm. *Image Processing On Line*, *4*, 128–148.

Wijewickrema, S. N. R., & Paplinski, A. P. (2005). Principal component analysis for the approximation of a fruit as an ellipse. *Full Papers/WSCG*.

Xinshao, W., & Cheng, C. (2015). Weed seeds classification based on PCANet deep learning baseline. *2015 Asia-Pacific Signal and Information Processing*

*Association Annual Summit and Conference (APSIPA)*, 408–415.

XIONG, J., LIU, Z., TANG, L., LIN, R., BU, R., & PENG, H. (2018). Visual Detection Technology of Green Citrus under Natural Environment. *Transactions of the Chinese Society for Agricultural Machinery*, (4), 5.

Yalcin, H. (2017). Plant phenology recognition using deep learning: Deep-Pheno. *2017 6th International Conference on Agro-Geoinformatics*, 1–5.

Yao, G., Lei, T., & Zhong, J. (2019). A review of Convolutional-Neural-Network-based action recognition. *Pattern Recognition Letters*, *118*, 14–22.

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 3320–3328). Curran Associates, Inc.

Zhang, C., & Kovacs, J. M. (2012). The application of small unmanned aerial systems for precision agriculture: a review. *Precision Agriculture*, *13*(6), 693–712. https://doi.org/10.1007/s11119-012-9274-5

# 7. Appendices

## 7.1 Estimating melon yield for breeding processes by machine-vision processing of UAV images

Conference: ECPA, Montpellier, France, 2019

Author's: Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A. and Klapp, I

Cited as: Kalantar, A., Dashuta, A., Edan, Y., Dafna, A., Gur, A., & Klapp, I. (2019). Estimating melon yield for breeding processes by machine-vision processing of UAV images. In Precision agriculture’19 (pp. 1386-1393). Wageningen Academic Publishers.

# Estimating Melon Yield for Breeding Processes by Machine-Vision Processing of UAV Images

A. Kalantar[1,2], A. Dashuta[1,3], Y. Edan[2], A. Dafna[4], A. Gur[4,], I. Klapp[1]

[1] *Institute of Agricultural Engineering, Agricultural Research Organization (ARO),Volcani Center, Rishon-LeZion, Israel*
[2] *Department of Industrial Engineering & Management, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel*
[3] *School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel*
[4] *Newe Ya'ar, ARO, Volcani Center, Ramat Ishay, Israel*

iftach@volcani.agri.gov.il

## Abstract

Monitoring plants, for yield estimation in melon breeding, is a highly labor-intensive task. An algorithmic pipeline for detection and yield estimation of melons from top-view images of a melon's field is presented. The pipeline developed at the individual melon level includes three main stages: melon recognition, feature extraction, and yield estimation. For each region of interest classified as a melon, the melon features were extracted by fitting an ellipse to the melon contour. A regression model that ties the ellipse features to the melon's weight is presented. The modified $R^2$ value of the regression model was 0.94. Comparing yield estimation to ground truth, the average estimation error was 16%. The yield accuracy is highly dependent on the ellipse estimation accuracy, with promising results of only 4% error for the best ellipse-fitted melons.

**Keywords:** Precision agriculture, machine learning, CNN, active contour, melon, breeding, phenotyping, yield estimation.

## Introduction

Melon breeding requires a detailed account of accumulated yield and general yield distribution, in addition to melon size and location, which are essential for connecting yield to treatment/melon genetics. Manual plants monitoring for yield estimation is a highly labor-intensive task, thus expensive (Gongal et al., 2015). Hence, automation of the process is beneficial. One proposed method for achieving an automated qualitative yield analysis is to apply computer-vision and machine-learning methods to color images of a top view of a melon field. Machine-learning is extensively used for fruit detection and recognition, using different recognition methods such as K nearest neighbors, K means with color information (Anisha et.al, 2013),(Qureshi et.al, 2017), Faster R-CNN (Bargoti & Underwood,2017). Detection of almonds, mangoes and apples resulted in precision in the range of 0.7 to 0.9 (Bargoti & Underwood,2017). Machine-learning techniques was also

used to estimating yield loss such as a system that detects fruits on the orchard ground (Choi et.al, 2013).
This work proposes a framework for detection and yield estimation of melons from color images, acquired from a digital camera mounted on a drone. Object detection under the highly variable outdoor conditions along with extraction of melon features related to yield, such as size, are detailed in the described algorithmic pipeline along with the results. The focus is on yield detection based on individual melons.

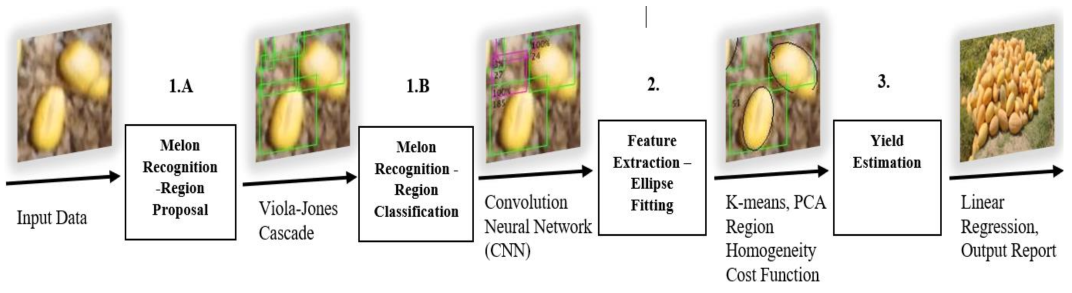
## Materials and methods

### *Data acquisition*

Data were acquired at an experimental agricultural research farm in Newe Ya'ar, Izrael valley, northern Israel. Latitude/Longitude 32.718492/35.181951 respectively. Foliage coverage was reduced to improve image detection by stopping irrigation 1 week before the acquisition. The acquisition was performed at midday on 17 Aug 2017, at fruit ripening stage. RGB images were acquired from a Sony ILCE-5000 camera mounted on a drone (Quad-Copter) hovering about 15 m above the field, with the camera facing vertically downward. An area of 280 m x 15 m was acquired. Before starting the image acquisition, 30 melons with different shapes, sizes and colors were randomly tagged in the field and used for ground-truth data. These melons were analyzed after image acquisition with a "Tomato Analyzer" tool (Gonzalo et al., 2009). The analysis provided a dataset of the ground-truth features of each melon, Additional similar data acquired in the following year (2018) were used to develop the statistical model to tie extracted section geometry to melon weight. The images were saved in .jpg format with a resolution of 3064 × 5456 pixels. Overall there were thousands of melons with high diversity. The variation in the dataset was intended to ensure the modeling of a robust and accurate detection algorithm that would generalize well under outdoor environmental conditions.

### *Data preparation*

The algorithms included two machine-vision phases: a cascade object detector (COD) (Viola & Jones, 2001), and a neural network classifier (NNC). To train the COD, 3031 negative training images (of background) and 1933 positive training images (of melons) were created. NNC training was based on 1383 positive (melon) and 1349 negative (background) training images. All melons were randomly selected. Augmentation techniques were used to increase robustness, resulting in a training set with 44,256 positive images and 43,168 negative images. Augmentation was achieved by applying blurring, a rotation operation, and adding Gaussian noise, resulting in 31 additional new images for each original training image.



Input Data
1.A
Melon Recognition -Region Proposal
Viola-Jones Cascade
1.B
Melon Recognition - Region Classification
Convolution Neural Network (CNN)
2.
Feature Extraction – Ellipse Fitting
K-means, PCA Region Homogeneity Cost Function
3.
Yield Estimation
Linear Regression, Output Report

Figure 31. Algorithm pipeline for automated yield tracking

*Algorithm*

The proposed algorithm pipeline for automated yield prediction from RGB images of a melon field includes the following three sequential main stages: 1) melon recognition, 2) feature extraction, 3) yield estimation. Each stage uses results from the previous stage. The input for the system are RGB images of a melon field, and the output is a report that includes each melon's location and weight. The algorithm pipeline is illustrated in Figure 1.

*Analysis*

Detection performance was evaluated using recall and precision indicators. A fruit detection was considered to be a true positive if the predicted and ground-truth bounding box had an intersection over union (IoU) greater than a fixed threshold (Bargoti & Underwood, 2017b). Two different thresholds were examined. The first was 0.5, used in PASCAL Visual Object Classes (VOC) challenges (Everingham et al., 2009), resulted in relatively small fruit size with respect to the image resolution, errors in detecting melons were received, causing them to be registered as false negative. To reduce the false negative misclassification, a smaller threshold of 0.2 was applied, which equates to a 58% overlap along each axis of the object; this was considered sufficient for the fruit-mapping application (Krizhevsky et al., 2012).

**Algorithm**

*Melon recognition implementation details*

The melons make up only a small portion of typical melon field images. Therefore, to reduce computational effort, the recognition process was split into two sub stages: detection of proposed "candidate" regions, followed by region classification. First, regions of interest (ROI) are derived, the proposal model runs in a sliding window over the entire input image and detects possible melon candidates. Since the detection of candidate regions requires an exhaustive search over the entire image, it is assigned to have low time complexity. It is also designed to have high recall, at the expense of low precision. Second, the candidates detected by the candidate region proposal model are classified by an algorithm, which is more computationally expensive, but compensates for the region proposal model's low precision (Dashuta & Klapp, 2018).

1A) *Candidate region proposal*

The Viola–Jones face detector was chosen for the candidate region detection model: it proposes ROI suspected of containing a melon. The Viola–Jones detector, a COD, is an ensemble of a number of weak classifiers. A weak classifier is constructed by simply thresholding one co-ordinate of some feature vector of the concurrent window (Dalal & Triggs, 2005). The feature type selected for the detector is the histogram of oriented gradients (HOG) Error! Reference source not found., which can capture the elliptical nature of the melon. The COD is arranged in stages with increasing complexity. The role of each stage is to decide whether the concurrent window is certainly NOT an object. If a stage decides that the concurrent window is not an object, the remaining stages are not evaluated. Hence, only true object windows trigger the entire cascade of stages. This mechanism ensures low time complexity of the model.

1B) *Region classification*

A convolutional neural network (CNN) Error! Reference source not found. was used for the region classification model. The model was trained using the well-known 'transfer-learning' methodology Error! Reference source not found.. The learning process was initiated with a network that was pre-trained on the CIFAR-10 dataset (Krizhevsky et al., 2012), a 10 category 32 x 32 pixels color image dataset. In this work, the final fully connected layer, which is essentially an image classifier based on a 64-dimensional feature vector produced by the hidden layers, was changed from a layer with output size of 10 labels (the original dataset has 10 classes) to one with an output size of 2 labels—melon and background. The training data were expanded using augmentation techniques (Wang et al., 2014). Results revealed that the algorithm achieves 85.01% precision and 89.8% recall compared to the human identification used for the ground-truth measurement.

2) *Feature extraction*

This stage works on patches of the image, that have been detected as ROI and classified as melon. Each ROI assumed to contain only one melon. For each melon, the exact location in the field and the features that are related to melon size were derived. These features help estimate the melon weight. The pre-knowledge that the shape of each individual melon can be well approximated by a spheroid (Heinzen et al., 1998) implies that from a top view, the melon contour will be recognize as an ellipse. This enables translating the feature-extraction problem into an ellipse-fitting problem. Once the ellipse with the best fit to each detected melon is derived, the feature extraction is trivial. The ellipse partitions the patch into two regions: inner region (inside the ellipse) and outer region (in the proposed candidate region, outside the ellipse), using the following decision rule: pixels in the inner region are classified as 'melon' and pixels in the outer region are classified as 'background'. Every possible ellipse can be parametrized by 5 parameters:

semi-major axis ($c$), semi-minor axis ($a$), centroid x co-ordinate ($x_0$), centroid y co-ordinate ($y_0$) and angle of tilt ($\theta$). Finally, the contour of every melon in the field can be approximated as an ellipse:

$$\frac{\left[(x-x_0)\cos(\theta)-(y-y_0)\sin(\theta)\right]^2}{a^2}+\frac{\left[(x-x_0)\sin(\theta)+(y-y_0)\cos(\theta)\right]^2}{c^2}=1 \quad (1)$$

Every point $\mathbf{x}=(a,c,x_0,y_0,\theta)\in\mathbb{R}^5$ in the parameter space corresponds to a single ellipse (single solution) in the given ROI. To derive these parameters, first an initial ellipse that fits the contour is derived, then parametrization is conducted, followed by solution optimization using a cost function minimization problem (Dashuta & Klapp, 2018).

To estimate the yield, a model that ties the melon's geometry to its weight was determined. A spheroid model was applied using a 3D shape with same-sized width and depth axes (Figure 2).

Assuming that a typical melon has a spheroid shape, the parameters derived above from the fitted ellipse in the 2D image of the melon, in particular the sizes of the minor and major axes, should be correlated to the semi-height and semi-width of the melon. Using these two parameters, the melon weight is predicted. The regression model was built using information from 30 randomly selected individual melons which were both imaged by the UAV and measured for their weight and geometry in the laboratory. The best derived regression model was:

$$\mathrm{W}=0.1096653+0.003397929\cdot \mathrm{c}\cdot a^2 \quad (2)$$

Where *W* is the melon weight, *a* is the ellipse semi-height and *c* is the ellipse semi-width.

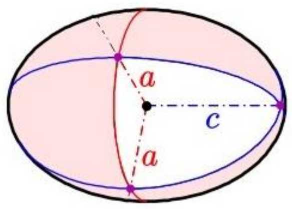


Figure 2. Ellipsoid model

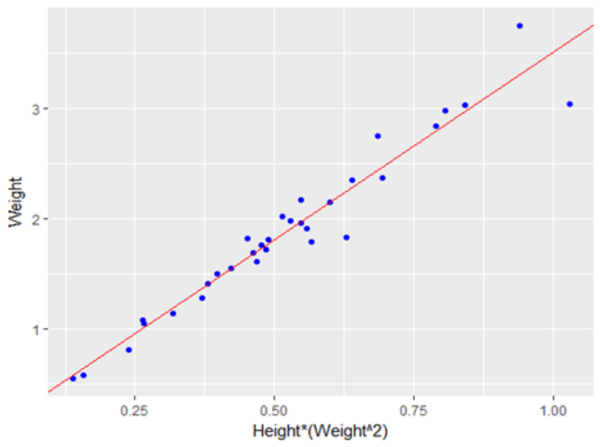


Figure 3. Weight and spheroid parameter regression results on training set analysis of regression

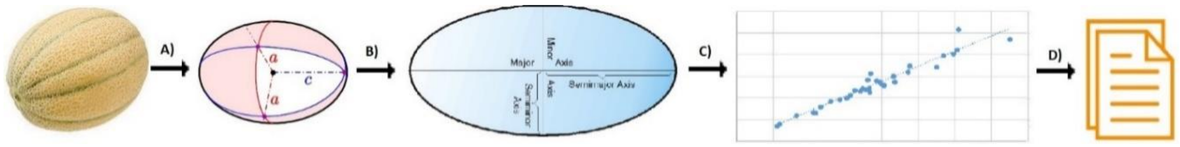


Figure 4. Yield estimation process. Stage (A) represents a melon by spheroid shape – use pre-knowledge of melon shape; stage (B) converts from 3D to 2D– the width and depth are minor axes and length is the major axis; stage (C) uses linear regression to predict yield of each melon; stage (D) generates a report for each melon location and yield estimation for the entire field

Following this model, fidelity to data points to the model was estimated using the adjusted $R^2$ figure of merit (Montgomery, 1997), resulting in a $R^2_{Adj}$ of 0.94, and implying a strong fit for the specific dataset (for n = 30, Figure 3). Another dataset of 135 melons obtained in the following year (July 2018) was used for cross-validation (Hawkins, 2004). The overall yield estimation from the ellipsoid parameter is presented in Figure 4.

The above method to fit an ellipse contour to the melon image was suggested. Feature extraction of such a contour is an ellipsoid section. Thus, assigning the ellipse parameters (Eq. 1) in the proposed regression (Eq. 2) results in an estimation of melon weight from post-processing of the UAV images.

The parameters in Eq. (1) are given in pixels. Pixels were converted into centimeters by calculating the ground sample distance (GSD). The GSD was estimated from the images of a signboard that was placed next to the 30 randomly selected individual melons. A calibration ratio (1/GSD), was derived by dividing the number of pixels in the signboard image (along an edge), by the edge size in centimeters. This ratio was applied to the features of the melon marked in the image and located near the signboard. For all other melons, the averages of all of the ratios were calculated and applied.

## Results

Precision vs. recall of the experimental results is presented in Figure 5, revealing that the precision is retained through almost all of the recalled range. The detection algorithm was able to detect only 87% of the tagged fruit used for the calibration. The GSD metric was used to convert the pixel units into centimeters with a derived ratio of approximately 0.4 cm per pixel. Using this ratio, the actual yield was forecast. The result overestimated the actual yield by 16%. The error was due to the ellipse error and the error in the GSD estimation. The UAV's distance from the melon effect the image GSD. Wrong GSD resulted in estimation error in melon's size. By applying a more accurate GSD estimation based on the size of the local known targets for the 5 best recognized melons (accurate contour detection), the error was reduced to 4%.

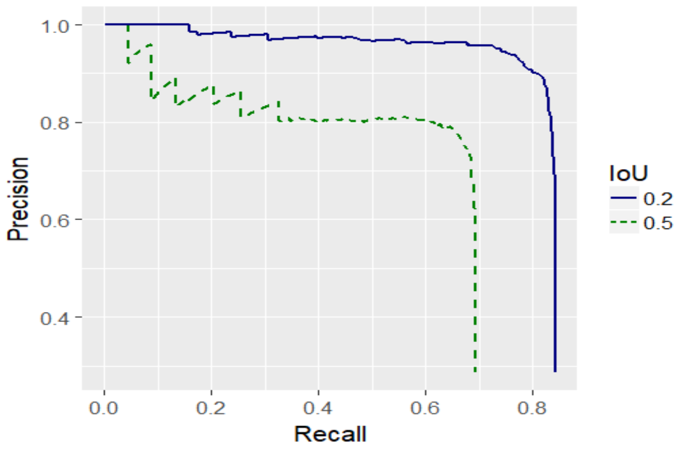


Figure 5. Experimental results precision vs. recall curve, average precision for melon detection using different IoU thresholds

## Discussion

Yield accuracy strongly depends on the ellipse estimation accuracy. The weight estimation error for the best fitted melons was only 4%. Improvement of melon recognition and the fitting model can potentially result in an applicable method for estimation of melon yield from aerial surveys. Ongoing work is focused on improving recognition by using improved object-detection techniques, such as a faster R-CNN object detector. In addition, several models for improving the ellipse-fitting problem, such as the Chan–Vese model for active contours, are under examination. Since individual melons in the same field ripen at different rates, harvesting cannot be done in a single pass. Therefore, the field workers must sample every melon in the field to determine their ripeness. An automated yield analysis system could guide the field workers directly to the ripe melons and thus save time.

## Conclusions

The above pipeline includes three main stages for yield detection based on calculation derived at the single melon level: melon recognition, feature extraction, and yield estimation. This scheme show promising capabilities. First, to recognize individual melon's couture. Second upon correct feature extraction predict melon weight with high accuracy. To improve the performances of the proposed pipe line the ongoing research focused on improving of learning process and possibly the active couture schema.

**Acknowledgements**
This work was partially supported by BARD Program number IS-4911-16 and by the Rabbi W. Gunther Plaut Chair in Manufacturing Engineering at Ben-Gurion University of the Negev.

## References

Anisha. S, Divya. G, Shanu. S 2013. "A Survey of Computer Vision Methods for Counting Fruits and Yield Prediction", International Journal of Computer Science Engineering (IJCSE), 2, 6, 2319-7323

Bargoti.S and Underwood.J 2017a "Deep Fruit Detection in Orchards" arXiv:1610.03677v2 [cs.RO] Last revisit 18 Sep 2017

Bargoti S., and Underwood, J. 2017b "Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards," Journal of Field Robotics 34(6), 1039-1060.

Chan T.F. and Vese L.A. 2011 "Active Contours without Edges," IEEE Transactions on Image Processing 10(2), 266-277.

Choi.D, Suk-Lee.W, Ehsani.R 2013 "Detecting and counting citrus fruit on the ground using machine vision" 2013 ASABE Annual International Meeting , Kansas City, Missouri, USA July 21 – 24, 2013 Paper Number: 131591603

Dalal N., and Triggs B. 2005 "Histograms of Oriented Gradients for Human Detection," In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1 (Institute of Electrical and Electronics Engineers,), 886-893.

Dashuta A., and Klapp I. 2018. "Melon Recognition in UAV Images to Estimate Yield of a Breeding Process," Light, Energy and the Environment (E2, FTS, HISE, SOLAR, SSL), OSA Technical Digest (Optical Society of America), paper ET4A.2. https://www.osapublishing.org/abstract.cfm?URI=EE-2018-ET4A.2 .

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. 2010 "The pascal visual object classes (voc) challenge". International journal of computer vision, 88(2), 303-338.

Gongal A., Amatya S., Karkee M., Zhang Q., and Lewis K. 2015 "Sensors and systems for fruit detection and localization: A review". In: Computers and Electronics in Agriculture 116, 8–19.

Gonzalo M.J., Brewer M., Anderson C., Sullivan D., Gray S., Knaap E. 2009, "Tomato Fruit Shape Analysis Using Morphometric and Morphology Attributes Implemented in Tomato Analyzer Software Program," Journal of the American Society for Horticultural Science 134(1) 77-87.

Hawkins D.M. 2004 "The Problem of Overfitting," Journal of Chemical Information and Modeling 44(1), 1-12.

Heinzen A., Shimmel C., Groppe R. and Davidson E.A., "Apparatus for Removing Rind from Spheroidal Fruits and Vegetables," U.S. Patent 5,806,414, 15 Sep 1998.

Krizhevsky A., Sutskever I., and Hinton G.E. 2012 “Imagenet Classification with Deep Convolutional Neural Networks,” in Advances in Neural Information Processing Systems 25, (Lake Tahoe, CA, USA). 1097-1105.

Li Z., Song Y., Mcloughlin I., and Dai L. 2016 "Compact Convolutional Neural Network Transfer Learning for Small-Scale Image Classification," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2737-2741.

Montgomery D.C. 1997. Chapter 13 in Design and Analysis of Experiments (fourth ed.), Arizona, USA: John Wiley & Sons,

Qureshi. W. S, Payne.A, Walsh. K. B, Linker.R, Cohen.O, Dailey. M. N, 2016 " Machine vision for counting fruit on mango tree canopies", Precision Agric 8, 224. https://doi.org/10.1007/s11119-016-9458-5

Viola P. and Jones M.J. 2001 “Rapid object detection using a boosted cascade of simple features,” In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1 (Institute of Electrical and Electronics Engineers), 511-518.

Wang J., Song Y., Leung T., Rosenberg C., Philbin J., Chen B. 2014 "Learning Fine-grained Image Similarity with Deep Ranking," In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1386-1393.

## 7.2 Code documentation

The first system can be found at the next link: Estimating open-field melon yield by machine-vision processing of UAV images

The system was implemented using MATLAB 2017b version

It includes the next folders:

- Model steps visualization – output images for each step in the system
- Results and analysis - include results for mAP, regression, final report and Weight analysis
- System code – include all function for each stage of the system
- tagged dataset – all the labeled images for training and testing

the system should be executed with the function runme.m which call to the function ProcessSingleImagefinal.m and start all the imaging process.

Before execution please set the image path at the parameter 'image_name' (line 33)

All details and description for each sub function can be found inside the source code.

The second system can be found at the next link : A deep learning system for yield estimation of melons using UAV images.

The system was implemented using python and will work only if the pc has GPU

It includes the next folders:

- Data – includes all the tagged data which were used for training and validation for each year. For each image attached a CSV file with melons annotations
- Extra – include images and film from the field test day
- Melon_detection_using_RetinaNet_Train_RetinaNet – include the files for training the RetinaNet network (detailed explanation provided below)
- Project_pipeline – include all the pipeline files (detailed explanation provided below)
- Results and analysis – include two folders:
    - Detection Retinanet results – in this folder we stored all the result for the tested images, also includes a script for generate mAP curve with the final result.
    - Final image result with ellipse fitted – include the result for all tested images after ellipse fitting stage
    - results of weight – include a script for building the regression model, also presented analysis for weight estimation for all the 135 selected melons

## Training the RetinaNet network

In order to train the RetinaNet network, please follow the steps below:

Installation

1) copy the Melon_detection_using_RetinaNet_Train_RetinaNet folder.

2) Please make sure `tensorflow` and 'Numpy' is installed as per your systems requirements. Otherwise run the next line using anaconda sell:

Numpy: `pip install numpy --user`

Tensorflow: use the link - https://www.tensorflow.org/install

3) Using anaconda shell, navigate to the 'Melon_detection_using_RetinaNet_Train_RetinaNet' folder and execute ` pip install . --user`.

4) If you have some issues with the installation, please use stackoverflow.

Training

For training on a [custom dataset], a CSV file can be used as a way to pass the data.

* See below for more details on the format of these CSV files.

To train using your CSV, run:

1) Running directly from the anaconda shell the line:

```
keras_retinanet/bin/train.py --weights snapshots/resnet50_coco_best_v2.1.0.h5 csv train_annotations.csv labels.csv --val-annotations val_annotations.csv
```

Where you need to update the next pass parameters:

* weights: Path to the weights for initializing training

* csv indicates retinanet is trained on a custom data set

* train_annotations.csv is path to training annotations

* labels.csv are the labels in the format class_name, class_id with 0 reserved for background class

* val_annotations is path to validation annotations

### Annotations format

The CSV file with annotations should contain one annotation per line.

Images with multiple bounding boxes should use one row per bounding box.

Note that indexing for pixel values starts at 0.

The expected format of each line is:

```
path/to/image.jpg,x1,y1,x2,y2,class_name
```

### Labels format

The class name to ID mapping file should contain one mapping per line.

Each line should use the following format:

```
class_name,id
```

For example - in my project the csv for training is 'train_annotations.csv',

validation annotations are in val_annoations.csv

and labels are in labels.csv

### Evaluating Results

To calculate mean average precision on the validation set, please run

```
keras_retinanet/bin/evaluate.py csv val_annotations.csv labels.csv snapshots/resnet50_csv_01_inference.h5 --convert-model
```

Here we pass the val_annotations, labels and path to the trained weights

### Running Inference on Images

To run inference on the trained model, first step is to convert the trained model to a format that can be used by inference. The command for this is:

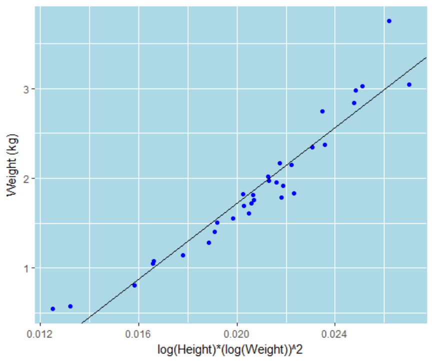
```
```

keras_retinanet/bin/convert_model.py snapshots/resnet50_csv_08.h5 snapshots/resnet50_csv_01_inference.h5
```

Here first path is the path to the trained model and the second would be the path to the converted inference model

We will use the 'resnet50_csv_01_inference.h5 Inference' for our testing images and for the production system

Also, for testing the Network, the file Melon_detect.com can be used

## **Project_pipeline**

In order to run the system, please execute the main.py file. This file includes several functions, each function proposed described below. Also, documented separately inside the code.
Before running the system, please create a folder with the name = 'images to process' and put all the images inside the folder. please set the parameter of *imgdir* and *imgSplitdir* to the correct path where the images are located.
The result of the system will be located at: '../images to process/image results'.
It will include a image with all the tagged melons and a CSV file with the location and weight estimation of each melon.

- Main.py – the main function
- chane_vese.py – function which perform a chane_vese method
- create_ellipse - artium.py – this function is not used at the current system
- create_ellipse.py – function that fit ellipse for each proposed bounded box. We use this function at this system
- draw box.py – function that plot on top of the images all detected bounded boxes
- eval.py – function which generate evaluation report for detection stage
- gradientDescentEllipseFit.py –function which execute the gradient descent, this function is not used at the current system
- non_max_suppression.py – function that composed the sub images after detection process into one big image and perform the NMS algorithm.
- split2.py – function that split the image into grid of 10 by 10 sub images

In addition to the presented function above, few additional function were used in order to create ground true labels for training
1_gt.py, 2.py, composed_coordinates_csv.py, composed_coordinates_xml.py, convert csv to txt.py
```

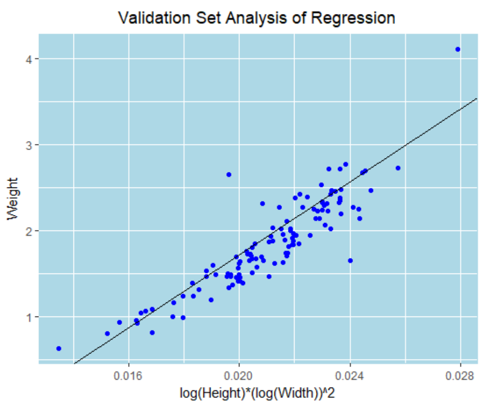
## 7.3 Additional regression for yield estimation

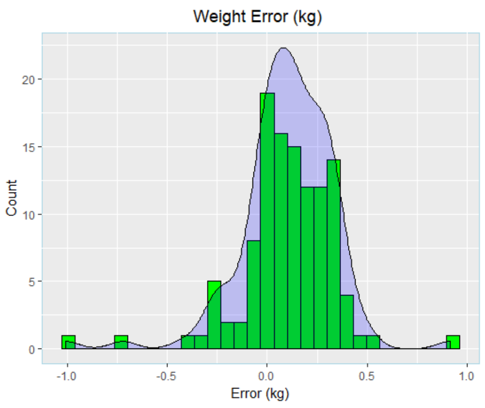
In addition to the regression models that were presented at section 4 and 5, we tried another regression which tried to estimate melon weight using logarithmic transformation of the max height (2*c) and max width (2*a) of each melon. The suggested model gained a correlation score of $R^2_{Adj}$ = 0.91. The resulted weight estimation regression model was:

$$Weight = -2.497493 + 210.8525 \cdot \log(c) \cdot (\log(a))^2$$



The model was tested using 116 randomly selected melons from images taken in the last season. The mean absolute percentage error (MAPE) index for individual melon estimation was 10% with an overweight overall yield estimation error of 5.5%.





Since the additional suggested model results were less better than the regression without the logarithmic transformation, we choose to not continue with this model.

אוניברסיטת בן גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת תעשיה וניהול

# הערכת יבול של שדה מלונים באמצעות תמונות אוויר שצולמו בעזרת כטב"ם תוך שימוש בלמידה עמוקה

אהרון קלנטר

ספטמבר, 2019

אלול, התשע"ט

מנחים :

פרופ' יעל אידן, אוניברסיטת בן-גוריון בנגב

דר' יפתח קלפ, מינהל המחקר החקלאי, מרכז וולקני

אוניברסיטת בן גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת תעשיה וניהול

# הערכת יבול של שדה מלונים באמצעות תמונות אוויר שצולמו בעזרת כטב"ם תוך שימוש בלמידה עמוקה

שם: אהרון קלנטר

מנחים אקדמיים : פרופ' יעל אידן

דר' יפתח קלפ

ספטמבר, 2019

אלול, התשע"ט

חתימת המחבר: ................. [illegible] ................................. תאריך: ......24.09.19...........

אישור המנחה: .................. Yael Edan ................................ תאריך: ......24.09.19..........

אישור המנחה: ............ [illegible] ............................. תאריך: ......24.09.19..........

אישור יו"ר ועדת תואר שני מחלקתית: .......... [illegible] ......... תאריך: .......25.09.19..........

# תקציר

תזה זו עוסקת בפיתוח מערכת לאומדן תנובת שדה מלונים בסביבה חקלאית באמצעות תמונות צבע שצולמו ממצלמה דיגיטלית המותקנת על כלי טייס בלתי מאויש (כטב"מ). המערכת מקבלת כקלט תמונת RGB אווירית של שדה מלונים, והפלט הוא דו"ח הכולל את המיקומו של כל מלון ומשקלו המשוער.
אחת המניעים ליישום מערכת הנ"ל קשורה לעובדה שאומדן תנובת השדה לפני הקטיף נחשב כמשימה עתירת עבודה, וזאת מכוון שנדרש לבצע הערכת יבול מפורטת ומדויקת עבור כל מלון בנפרד, תוך התחייסות למידות המדויקות של גודלו ומיקומו בשדה.
בעיית זיהוי והערכת תנובת השדה בסביבות חקלאיות באמצעות שיטות לראייה ממוחשבת נחקרו במשך עשרות שנים. בשנים האחרונות, חלה התקדמות משמעותית בתחום הלמידה העמוקה אשר הציגה ביצועים מרשימים בפתרון בעיות רבות של איתור אובייקטים. יחד עם זאת, למרות המחקר הרחב בתחום זה, מעט מאוד יישומים חקלאיים ממוסחרים משתמשים במודלים מבוססי ראייה ממוחשבת, וזאת מכיוון שזיהוי אובייקטים בעולם האמיתי נחשב למשימה קשה במיוחד, ובפרט בסביבה חקלאית מורכבת ולא מובנית. בנוסף, חלק גדול מהעבודות שפורסמו עד כה, התמקדו בשיפור הדיוק של האלגוריתמים לזיהוי מדויק יותר של כמות הפירות שבתמונה. בעבודה זו אנו מציגים מערכת שתנבא לא רק את כמות הפירות בתמונה אלא גם תשערך את המשקל בפועל של כל פרי.

במחקר זו אנו מציגים שתי מערכות שונות שפותחו להשגת היעדים שהוזכרו לעיל. בשתי המערכות שלושה שלבים: זיהוי המלון, חילוץ תכונות המתארות את גודל המלון, שערוך משקל המלון על סמך תכונות אלו.

המערכת הראשונה מסתמכת על שיטות לראייה ממוחשבת קלאסית יחד עם רשתות נוירונים עצביות "רזות" הדורשות מאמץ חישובי מופחת. המערכת השנייה פותחה במטרה לתת מענה טוב יותר לחלקים פחות יעילים מהמערכת הראשונית, פיתוח המערכת המתקדמת בוצע על בסיס אלגוריתמים מעולם הלמידה העמוקה, ובפרט, רשתות נוירונים עצביות עמוקות. במסגרת התזה בוצע ניסוי יעודי שמטרתו לאסוף תמונות של שדה מלונים שבעזרתם בוצע תהליך למידה בעבור שני המערכות. תוצאות המערכת המתקדמת מראות זיהוי גבוה ורמה מבטיחה של אומדן היבול.

התרומה המרכזית של תזה זו הינה פיתוח מערכת לאומדן תנובת שדה חקלאי בעזרת תצלומי אויר, המערכת מצליחה לבצע זיהוי מדויק של אובייקטים בגודל קטן, כמו גם, ביצוע שערוך משקלו של כל מלון בודד בנוסף לספירת המלונים המסורתית. המחקר מספק עדויות אמפיריות לכך שניתן להשיג אוטומציה של משימה חלקאית הנ"ל באמצעות מערכת הנדסית מבוססת ראייה ממוחשבת. עם כוונון עדין נוסף לחלק מהאלגורתמים במערכת, המערכת יכולה להיות בעלת כושר יצרני.

**מילות מפתח:** ראיה ממוחשבת, חקלאות מדייקת, זיהוי פירות, רשת נוירונים קונבולוציונית עמוקות, הערכת יבול, הערכת משקל, מלונים