

# Strawberry counting and ripeness detection

**Nour Boulahcen** (22312910), **Borja García-Quiroga** (22322676), and **Darragh Murray** (18320728)  
(boulahcen@tcd.ie) (garcaqub@tcd.ie) (dmurray4@tcd.ie)

*The University of Dublin, Trinity College, Ireland*

Automated harvesting is becoming increasingly important as the number of farmers declines and farming efficiency becomes increasingly critical. One of the most challenging tasks in this field is strawberry harvesting. These fruits can be difficult to spot for machines, and harvesting them at the wrong maturity level can significantly impact their quality for consumption. In this report, we discuss the current state of the art in counting and estimating the ripeness of strawberries and evaluate the performance of three different methods using the same dataset. Our goal is to identify the most effective methods to perform these tasks accurately, which can aid in developing effective robotic harvesting systems.

## 0 Introduction

With the decrease in the number of farmers that are not replaced (Ritchie [23]) and the improvements of robotics with new farming robots and automated farms as Cheng et al. or David et al. [4][24], we need more efficient computer vision methods to be able to make these robots effective, so that they can replace human work.

One of the most challenging tasks in automated harvesting is the harvest of strawberries. The first reason is that there are several on a small plant, and they are often hidden, making them difficult to find and pick (Mahmood et al. [16]). However, harvesting them too late when they are ripe will cause them to be overripe when they reach the consumer, and harvesting them too early will also make berries not proper for consumption. The sweet spot for harvesting them is when they are partially ripe (Chandy [3]). Therefore, automated strawberry harvesting needs accurate counting, localisation and ripeness estimation methods. In this report, we discuss the state of the art of algorithms for counting and estimating ripeness, and we will implement three of them and compare their performance using the same dataset.

## 1 Background

Current methods for counting and estimating the ripeness of strawberries were reviewed to give a good overview of the state of the art. The review on counting methods was not restricted to strawberry counting because the algorithms for object detection are usually generalisable to many kinds of objects. However, efforts were made to find specific techniques to improve these algorithms for strawberry counting and segmenting tasks, specifically.

On the other hand, for ripeness estimation, the focus was on strawberry ripeness estimation because, as mentioned earlier, this task is particular to this fruit.

Papers using different approaches, ranging from naive algorithms to machine learning and deep learning approaches, were researched and reviewed to compare the advantages and limits of each approach.

### 1.1 Review

This project has two key components: object counting and classification. Object counting and object classification are the two main components of this project. These are relevant tasks in computer vision with multiple applications and have been widely studied for many real-world applications. In this section, we break down different proposals made for these fields.

#### 1.1.1 Object counting

Object counting involves estimating the number of occurrences of objects in an image or video. It has received considerable attention since the early stages of computer vision. Most current techniques are split into three categories: 1) regression counting, 2) detection counting, and 3) clustering counting. Early approaches tend to focus on clustering-based methods due to computing power constraints. However, recent years, research has focused more on detection methods because of their broader applications and advantages. In this paper, we focus on the latter.

Li et al. [14] provide an extensive review on object counting substantiated by more than 50 articles published in the past decade. Focused on underwater computer vision, the authors analyse and compare the pros and cons of major counting methods and their applications. The review

also identifies current trends in the field and challenges to be solved.

Cholakkal et al. [5] proposed a new approach for density map estimation called the "image-level lower-count (ILC)" method. The approach used a pre-trained ResNet50 network and had two output branches: image classification and density branch. The image classification branch estimates the presence or absence of objects. In contrast, the density branch predicts the global object count and the spatial distribution of objects by constructing a density map. The network has a fully convolutional architecture, replacing pooling layers with 1x1 convolutions.

A similar architecture is proposed by Laradji et al. [17]. Their model extends what was previously introduced by Shelhamer, Long et al. [26] to perform object counting and localisation. Authors propose and implement a new loss function called the "localisation-based counting loss (LC)", and the proposed model is referred to as "LC-FCN". It is designed to handle point supervision, a weak form of annotation in which only the centre points of objects are labelled.

In Oñoro-Rubio and López-Sastre [6], the authors tackle object counting in images through regression neural networks for counting. They propose two novel convolutional neural networks (CNN): the Counting CNN (CCNN) and the Hydra CNN. Both methods use density map estimation to compute the regression, although they vary in the geometry awareness capabilities and training data.

Both Zhou et al. [29] and Ge et al. [9] propose using variants of the Faster R-CNN specifically for strawberry detection. The first describes the method and calls it "Improved Faster-RCNN", while the latter uses the "Mask-RCNN" algorithm. Faster R-CNN is a two-step object detection algorithm that generates regions of interest and then classifies or discards the objects within those regions. Each modification adds an improvement: the first one proposes specific modifications to make the system more robust for detecting strawberries in ground-level RGB images. The second adds an additional segmentation output to Faster R-CNN that allows extraction of the exact information from images to perform further analysis. A different deep-learning network is proposed by Wu et al. [28] for strawberry detection and counting, as well. The  $U^2$ -Net is used to segment strawberries and backgrounds in images.

YOLOv5 is used in Fan et al. [8] to detect—and, therefore, potentially count—strawberry instances. The YOLO series—an acronym for "You Only Look Once"—(Redmon et al. [19]) has been one of the most discussed and cited recognition algorithms in recent years. YOLO is a single-stage convolutional neural network (CNN) with fast and powerful detection capabilities. YOLOv5 (Jocher et al. [13]) is the fifth version of this series of algorithms and focuses on improving speed while guaranteeing accuracy.

### 1.1.2 Ripeness classification

Ripeness classification is the process by which the maturity level of a vegetable or fruit—in this paper, a straw-

berry—is determined. Although ripeness can be detected using various methods, this paper focuses exclusively on computer vision methods.

These methods classify instances among various classes that can grade from unripe to fully ripe and even rotten. Standard feature extraction techniques for ripeness classification are colour, texture, or shape analysis, as well as class-based detection or segmentation. Approaches have been made to different fruits and vegetables. However, features are highly particular to each species. This review focuses on strawberry ripeness classification.

The methods reviewed can be classified into three groups: 1) class-based detection, 2) machine learning classification, and minorly 3) algorithmic classification.

A wide variety of approaches can be found in systems belonging to the same group. For example, Fan et al. [8] (YOLO) and Ge et al. [9] (Mask R-CNN) produce systems that classify the ripeness level of each instance in detection time, returning for each instance the corresponding label. Both approaches classify ripeness into four categories. Conversely, Zhou et al. [29], although belonging to the Faster R-CNN group and closely related to Ge et al. [9], only collected ripe fruit by training the network with an all-ripe dataset.

Many Machine Learning classifiers have been proposed, varying broadly in algorithms and feature selection. For example, Anraeni et al. [1] and Indrabayu et al. [12] use the RGB features of the berry to train a 7-Nearest Neighbours (K-NN) and an SVM with Radial Basis Function (RBF), respectively. Both systems use three classes for classification: unripe, partially ripe, and ripe. Thakur et al. [27], instead, widen feature selection to include size and shape in addition to colour to train a Convolutional Neural Network.

Although using a Machine Learning classifier, Shao et al. [25] propose a much more sophisticated feature extraction algorithm. First, hyperspectral imaging was used to acquire strawberry images at three ripeness stages: ripe, mid-ripe and unripe. Then, the spectra were pre-processed by different methods, including X-loading weight, competitive adaptive reweighted sampling (CARS) and successive projections algorithm (SPA), which were applied to extract the effective wavelengths. Finally, training an SVM model with effective wavelengths obtained an outstanding performance.

Wu et al. [28] propose a diverting approach by determining the ripeness level based on the ratio of red pixels within the segmented strawberry mask.

## 1.2 Review conclusions

Experiments have been conducted with three very different approaches to implement the ripeness estimation: 1) A naïve approach with pixel counting [28], 2) a machine learning approach with feature extraction and KNN algorithm [1], and 3) a Deep Learning approach with YOLOv5 [8].

Because [28] and [1] approaches need inputs with only one strawberry, ways to perform instance segmentation were prioritized. Therefore, it was decided to implement

Mask RCNN. This algorithm can count the instances and segment the strawberries very precisely. Otherwise, when using YOLOv5, there is no need to find additional algorithms for counting and segmentation, as it does both things in one execution.

Initially, the Qin et al. [18] paper was assessed to implement instance segmentation, as only recently they published a new type of deep neural network known as 'highly accurate dichotomous image segmentation—or DIS, for short. We trained the a model for DIS using the given strawberry data and while the results were impressive, there was no built-in functionality to isolate and count the specific instances of strawberries, as the algorithm only removed the background of images while retaining objects of interest. Hence, Mask R-CNN proposed by [9] was our algorithm of choice for instance segmentation.

## 2 Data

### 2.1 Presentation

The same dataset of strawberry pictures provided by Trinity College Dublin was used to train and test these methods. This dataset contained 3000 high-resolution pictures of strawberry plants with several fruits and grayscale pictures with segmented instances. The dataset also contained each picture file containing the bounding box coordinates and the label (0: unripe, 1: partially ripe, 2: fully ripe) for each strawberry.

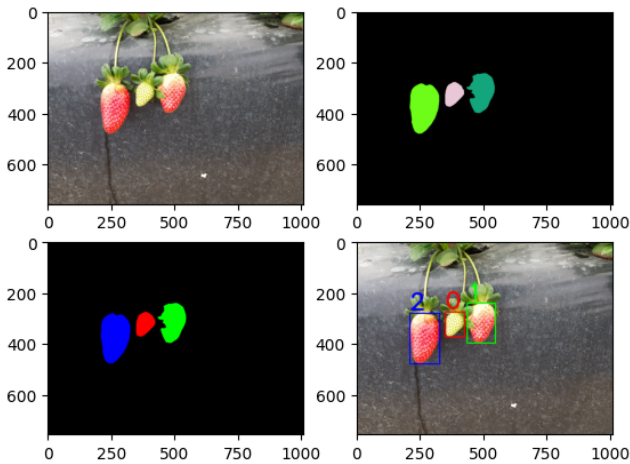


Fig. 1. Dataset summary: (a) Input image, (b) Instance segmentation, (c) Class segmentation, (d) Bounding boxes + classes.

In total, the dataset contained over 17000 individual strawberries.

### 2.2 Pretreatment

The dataset had to be pretreated to obtain additional inputs that could be easily given to preliminary models to assess the prediction quality they offered and train some of the final models. This pretreatment consisted of getting individual, masked and cropped pictures for each strawberry in the dataset.

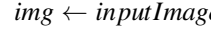

The process consisted of separating each instance segmented strawberry, multiplying it by the original image to obtain the masked strawberry, and finally obtaining the correct label for it by getting the most likely label based on bounding box proximity. Algorithm 1 shows the pseudocode.

---

#### Algorithm 1 Dataset pretreatment pseudocode

---

```

 ← inputImage
 ← inputSegmentation
while subMsk = getSubMask(msk) do
  boundingBox ← getBoundingBox(subMsk)
  minDist ← ∞
  bestBB ← None
  while candBB = getCandidateBB() do
    dist ← euclidian(boundingBox, candidateBB)
    if dist < minDist then
      minDist ← dist
      bestBB ← candBB
    end if
  end while
  maskedImg ← img × msk
  cropped ← crop(maskedImg, boundingBox)
  label ← getLabel(candBB)
  storeImage(subMsk, label)
end while

```

---

## 3 Image Segmentation and counting with Mask RCNN

Ge et al. [9] show a method of detection and instance segmentation of strawberries using a Deep Convolutional Neural Network (DCNN). Their paper used only 310 images of strawberries and reached 0.68 accuracies for the bounding box overlap between the raw detected and ground truth. So they decided to add a refinement method using the width and height ratio (WHR) of the output mask to detect occlusion, and they were able to reach a 0.87 accuracy for the bounding box overlap.



Fig. 2. Mask-RCNN Architecture used for strawberry instance segmentation. (Ge et al. [9])

RCNN stands for Region-Based Convolutional Neural Network. This approach uses bounding boxes across the object regions to apply the CNN on each Region of Interest (ROI) to classify multiple image regions in the proposed class, as shown in Figure 3.

Another version of RCNN was proposed: Faster RCNN, a new two-stage architecture. First, the Region Proposal Network (RPN) proposes multiple objects available in a given image. Then, the Fast RCNN extracts the features using RoI Pool (Region of Interest Pooling) on each object

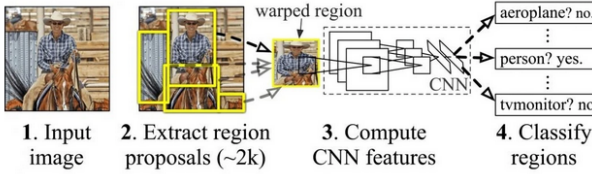


Fig. 3. Concept of RCNN. (Girshick et al. [10])

given by the previous stage to perform the classification and the bounding box regression.

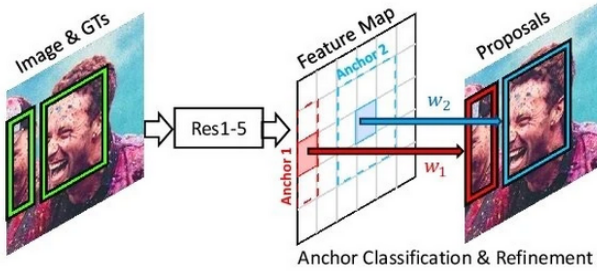


Fig. 4. RPN Stage (Ren et al. [22])

Mask RCNN is the state-of-the-art Neural Network for image and instance segmentation. The significant addition of Mask-RCNN is the pixel-to-pixel alignment. It uses the same architecture as Faster-RCNN with the two stages, but for the second stage, in parallel to the box and class prediction, Mask R-CNN has a third branch outputting a binary mask for each ROI.

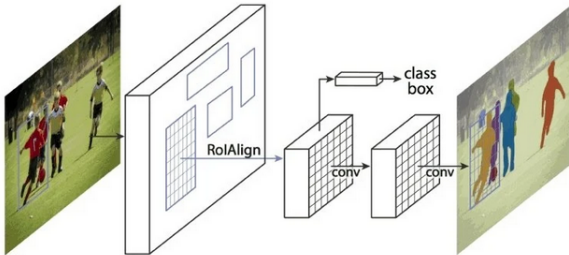


Fig. 5. Mask R-CNN framework for instance segmentation (He et al. [11])

The advantage of Mask-RCNN is that it can be pre-trained (often with COCO dataset), so it does not need a vast training set and time to reach high accuracy.

We used Mask-RCNN for the segmentation task because we wanted to use all the images as one class to gain accuracy. Also, this model needs less work for the labelling if we already have a trained model for the Ripeness estimation because we need to draw the mask on each strawberry without labelling them, so we do not need any knowledge in agriculture to do it.

Each instance of the objects in the segmentation dataset was separated into separate images. All of them were put into a single tensor of dimension  $Width \times Height \times N$ ,  $N$

being the number of instances in that image. The bounding boxes are obtained from the masks, as well. As we only want to separate instances, all labels are set to one. All the input images and outputs—consisting of masks, bounding boxes and labels—were converted into tensors using PyTorch before training the model.

Guided by the Side [[7]], these tensor features have been given to a PyTorch ResNet50 Mask R-CNN pre-trained using the COCOv1 ([15]) dataset for training. In addition, Adam optimiser has been set during the training and a penalty parameter of  $l_r = 10^{-5}$ .

#### 4 Ripeness estimation with Pixel counting

Wu et al. [28] use  $U^2$ -Net algorithm to segment the images to extract only the strawberries. Then they use a two-pass algorithm to get individual strawberries. We will not focus on this method for our implementation but rather on their ripeness estimation method. Their method is a very naive approach based on the ratio of the red pixel in each image to determine a threshold to classify the ripeness. Moreover, in their paper, they only use two classes. Therefore, they are not focusing on the accuracy of their technique for the ripeness estimation. However, we still wanted to test how a simple algorithm like this could give results to compare it to the other one as a Baseline.

Wu et al. do not discuss how to implement the pixel counting method but broadly expose the main reasoning and ideas behind their method. Therefore, it was necessary to design a detailed implementation that would match the principles and main logic discussed in the paper. All the design choices were backed by the statements in the paper and mathematic principles to obtain normalised features that could be generalised.

The algorithm implemented takes all the pixel values in the R channel of the image and multiplies them by the segmentation mask obtained from the Mask R-CNN module. This way, only the red pixels belonging to the detected strawberries are obtained. Then, these pixels are divided by the number of non-black pixels in the mask. In other words, the algorithm obtains the strawberry's average value of the red channel. These values are then given to a trained classifier to return the class. The pseudocode for this algorithm is shown in Algorithm 2. Because the calculation were too long, we used a reduced dataset with 157 images of individual strawberries with 125 training images.

---

#### Algorithm 2 Pixel counting pseudocode

---

```


img ← inputImage
msk ← inputSegmentation
R, G, B ← separateChannels(img)
rBerry ← R × msk
redSum ← sum(R)
mskSum ← sum(msk)
avgR ← rSum/maskSum

```

---



## 5 Ripeness estimation with Feature extraction and KNN

In their paper, Anraeni et al. [1] are experimenting with a ripeness identification method for strawberries using what they call 'RGB feature extraction' and a K-NN algorithm to classify input images into one of four categories/classes: ripe, unripe, raw and other. Their dataset was minimal (30 images for training and 20 for testing), but they achieved an accuracy of 85% using the twenty testing images. However, all error was due to the inability of the K-NN algorithm to correctly predict the ripeness of a strawberry when it is only partially ripe. With more test images present and the removal of the 'other' class, the accuracy would, in reality, be lower. Because we are using a different method for the segmentation and isolation of strawberries, the right side of the graphic below is what we focused on in this paper.

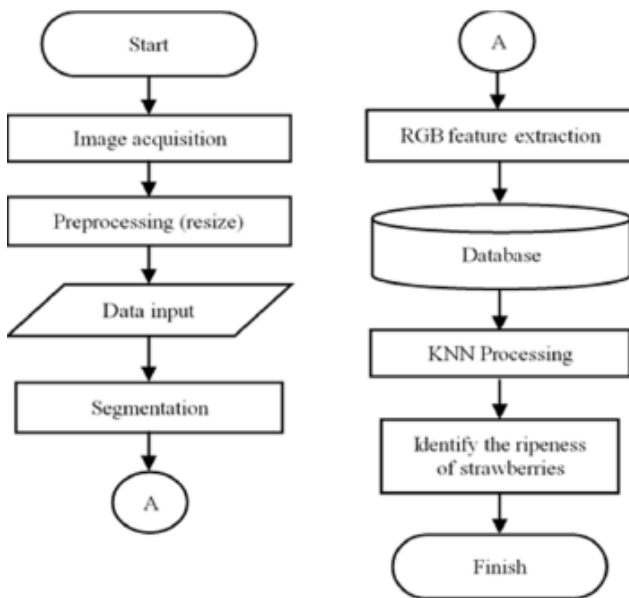


Fig. 6. Flowchart of the system. (Anraeni et al. [1])

The ripeness detection method used by [1] used the K-Nearest Neighbours algorithm to select the K 'closest' images from the training set and then determine the most popular class amongst the selected images in order to predict the class of the input image. Entire images were not compared; instead, a process called 'RGB feature extraction' was used to obtain a single three-value vector (the average centroid) for each training image. These vector values were stored in a database with the class of the image they came from. Any input images would have their own centroid calculated and its distance to the stored images computed via a Euclidean distance function that takes the stored centroid of a training and the computed centroid of the input image.

As mentioned earlier, we used the Mask-RCNN algorithm to first process any input images into a group of single isolated strawberries, each of which was also classified by the Mask-RCNN algorithm. Then we fed each labelled and isolated strawberry to the rest of the application for feature extraction and K-NN algorithm ripeness detection.

Finally, the K-NN predicted label for each strawberry was overlaid onto the original image, as was its bounding box.

The K- Nearest Neighbours algorithm model is relatively simple compared to deep learning models. First, using the Mask-RCNN algorithm, we created a folder of a couple of dozen isolated strawberry images labelled (in terms of ripeness) by the last number in the file name. Then, each isolated image was fed to our training function, which calculated the centroid of that image (using the process described in the paper) and then stored that centroid along with the class of the training image in a database. To calculate the centroid for any one image, an initial centroid consisting of the range of R, G, B values within an image must be generated. In our case, we created an initial centroid with two classes, one consisting of the lowest R, G, B values in the image, and the other consisting of the highest R, G, B values inside the image.

R			G			B		
52	62	88	47	80	57	25	30	28
133	180	225	71	110	150	32	61	93
125	166	233	120	115	227	124	60	229

Class	Component Centroid		
	R	G	B
1	52	47	25
2	233	227	229

Fig. 7. Initial centroid for some input image. Anraeni et al. [1])

Then, for each pixel in the input image, it's R, G, B values are turned into a vector, and the Euclidean distance between that RGB vector the each of the classes in the initial component centroid is determined and recorded.

Pixel's order	Component			Euclidean Distance	
	R	G	B	k1	k2
1	52	47	25	0	326.77
2	133	71	32	84.77	270.45
3	125	120	124	143.03	184.76
4	62	80	30	34.84	300.75
5	180	110	61	147.13	211.47
6	166	115	60	137.27	213.52
7	88	57	28	37.48	300.54
8	225	150	93	212.51	156.48
9	233	227	229	326.77	0

Fig. 8. Calculating distance from each pixel to each initial component centroid class. Anraeni et al. [1])

Each pixel was assigned to the class it was closest to, and then all the pixels assigned to class 1 were summed and averaged to create a new class 1, and all the pixels assigned to class 2 were summed and averaged to create a new class 2. Finally, these two classes were then themselves summed and averaged to create a final centroid feature. We used the Python key-value store Pickle to store each training image's final centroid feature and their associated ripeness label.

The last stage of the algorithm was implemented by creating a function that would take a single image and a K-value, compute its distance to each training image centroid and keep track of the closest K (centroids, ripeness



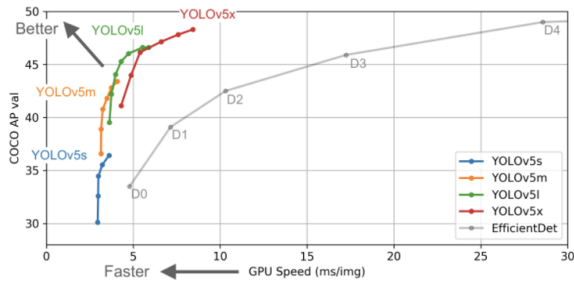


Fig. 11. Evaluation metrics of YOLOv5 sub-versions on the COCO dataset. (YOLOv5 [13])

We organised our dataset into the original images and text files containing the bounding boxes and the ripeness class of its matching image. We then experimented by training custom models with the different pre-trained models that YOLOv5 offered—e.g. YOLOv5s6, YOLOv5x6, or else—for short amounts of epochs until we found one that was most suitable for our needs. Figure 11 shows that some models are more accurate than others, with a trade-off between real time computation cost and accuracy. That is, as the accuracy of the model increases, so too does the time to evaluate an input image, as does the time required to train the model.

Initially, we tried to train a custom model using YOLOv5x6, the most accurate and largest pre-trained model. However, the time required for training was too long, so we downgraded to YOLOv5l6, which took 6 hours to complete 75 epochs of training. As recommended by the YOLOv5 team, the training set has also used a validation set during training. To start the training process, only the train.py script provided by the creators needed to be run after setting up the environment.

The scripts to take an input image, use a custom model, and generate the output were also provided in the repository. Therefore, the code needed to be slightly modified to run correctly with our data.

## 7 Results

### 7.1 Evaluation Mask RCNN

Segmentation evaluation metrics are challenging to get and interpret, as they are not as straightforward to get as metrics for other types of computer vision techniques. Accuracy metrics of segmentation problems are typically calculated as the proportion of correctly classified pixels in the images.

For this paper, pixel accuracy is the metric chosen. This metric has been particularly calculated as the fraction obtained by dividing the intersection of predictions and targets, between the union of predictions and targets.

The implementation of Mask RCNN discussed in this paper obtains a pixel accuracy of 88.20% when predicting a subset of a 10% of unseen images. This metric is truly outstanding taking into account how this metric is computed.

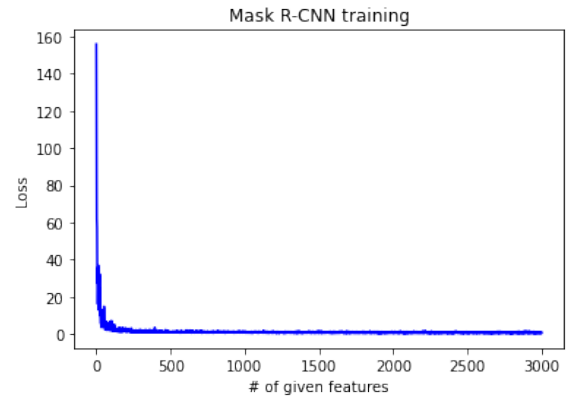


Fig. 12. Mask RCNN training loss evolution as the training set is incremented.



Fig. 13. Test image results using Mask RCNN segmentation.

### 7.2 Evaluation Pixel Counting

First, the results of tuning the thresholds can be seen in Figure 14. The optimal threshold values were 0.1 and 0.7, with an accuracy of 0.875 (Table 1). We also trained Several Classifier on the same data to see if we could improve the accuracy (Figure 15), and we found that we were able to reach an accuracy of 0.91 on the testing set with just 125 images in the training set.

### 7.3 Evaluation Feature extraction and KNN

For the RGB Feature extraction combined with KNN, we start by verifying that the number of neighbours given

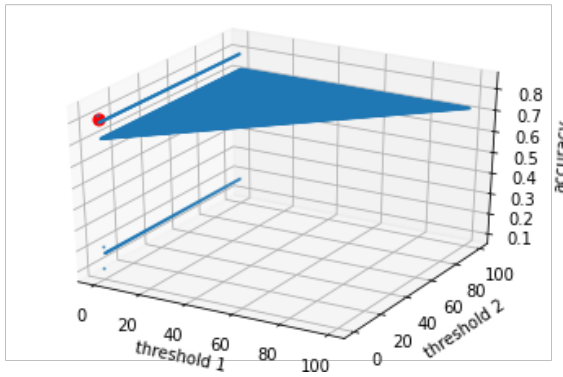


Fig. 14. Thresholds tuning for the Pixel counting method.

Threshold Training accuracy	0.952
Threshold Testing accuracy	0.875
Decision Tree Training accuracy	0.96
Decision Tree Testing accuracy	0.90625

Table 1. Pixel counting accuracy metrics

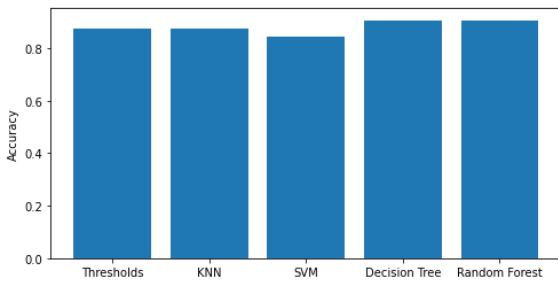


Fig. 15. Comparison of the accuracy for different classifier with the pixel ratio method.

in the paper was optimal. We found that with more data,  $k = 15$  gave the highest accuracy Table 2.

KNN Training accuracy	0.968
KNN Testing accuracy	0.966

Table 2. KNN accuracy metrics.

We also compared the accuracy of different classifiers such as SVM, Decision Tree or a random forest, but we found similar results.

### 7.4 Evaluation YOLOv5

We are still underfitting, as shown in Figure 20 and Figure 21. We could continue the training to reach higher precision/recall. Unfortunately, the training time for this algorithm was superior to 6 hours, and we could not finish the training (66/100). The optimal number of epochs in the paper is 300 [8]. However, we are still reaching a maximum precision of 0.99.

### 7.5 Comparison Counting

If we compare the two Ripeness Methods based on single strawberry image input, we notice that even if the results are very close, The RGB Feature extraction has a better accuracy overall.



Fig. 16. Input and output images to the Pixel Counting/Naive algorithm.

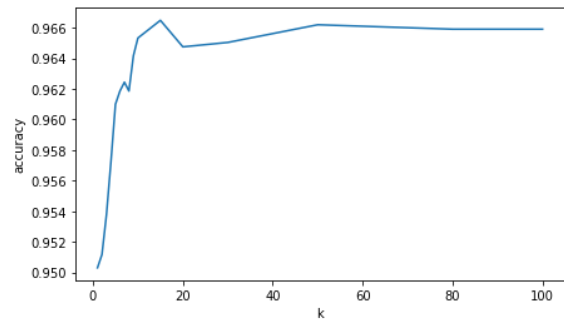


Fig. 17. Tuning of the KNN classifier for the RGB Feature extraction method.

### 7.6 Comparison Ripeness Estimation

We can see our implementation has even better results by comparing our results to others in the literature, as seen in Table 3. This is probably due to the size of the dataset we are using and the quality of the segmentation mask for the Mask R-CNN. Also, in some cases, when training some



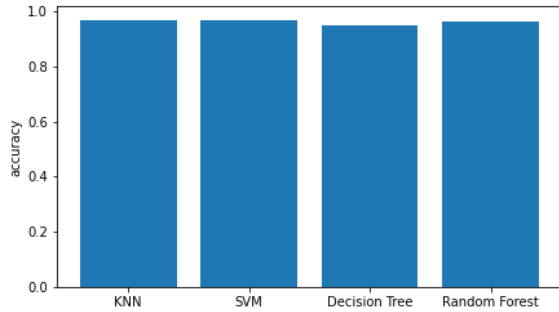


Fig. 18. Comparison of the accuracy for different classifier with the RGB Feature extraction method.



Fig. 19. Input and output images to the KNN algorithm.

models, we could only use some fraction of the dataset. The results could reach higher scores if that had been possible.

## 8 Limitations

Each of our algorithms has its own set of limitations. For example, the pixel counting algorithm is flawed in that if it is presented with an image containing no strawberries at

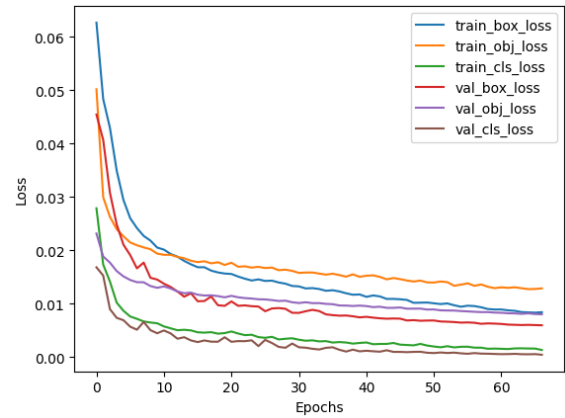


Fig. 20. YOLOv5 training and validation loss as epochs grow.

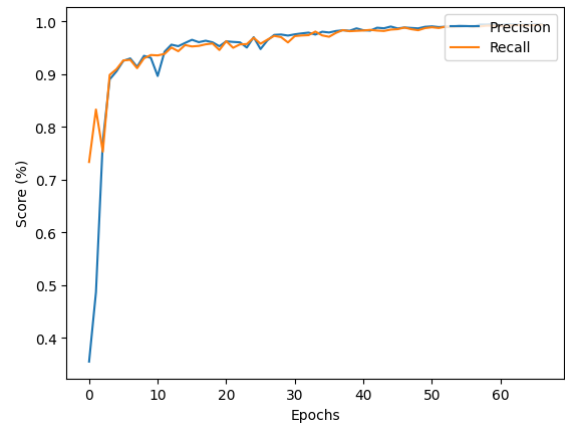


Fig. 21. YOLOv5 precision and recall scores as epochs grow.

Method	Litterature	Our Model
Mask RCNN	0.68 accuracy	0.88 accuracy
Pixel Ratio	-	0.875 accuracy
RGB Feature	0.85 accuracy	0.97 accuracy
YOLOv5	>0.9 accuracy	0.99 precision

Table 3. Result comparison between the implementation proposed and existing implementations in the literature.

all, the output will still be either ripe, unripe or partially ripe depending on the colour content of the image. Our implementation of the K-NN algorithm suffers from the same flaw, but with some adjustments it could be capable of identifying when there is no strawberry present in the image. To do this, we would have to re-train the database with background images that contain no strawberries and a new class: 'other'.

Another major limitation is that we are training our models strawberries from one farm, and in quite specific lighting conditions, so our model is probably very specific to the dataset we were given. However, if the use-case is automated harvesting, it's not a limitation since we want the algorithm to be specific for the farm we are working for.

Also, we did not analyse the combination of the Mask-RCNN and the ripeness estimator, so perhaps there is a propagation of an error that is strongly decreasing the accuracy for the ripeness estimation.



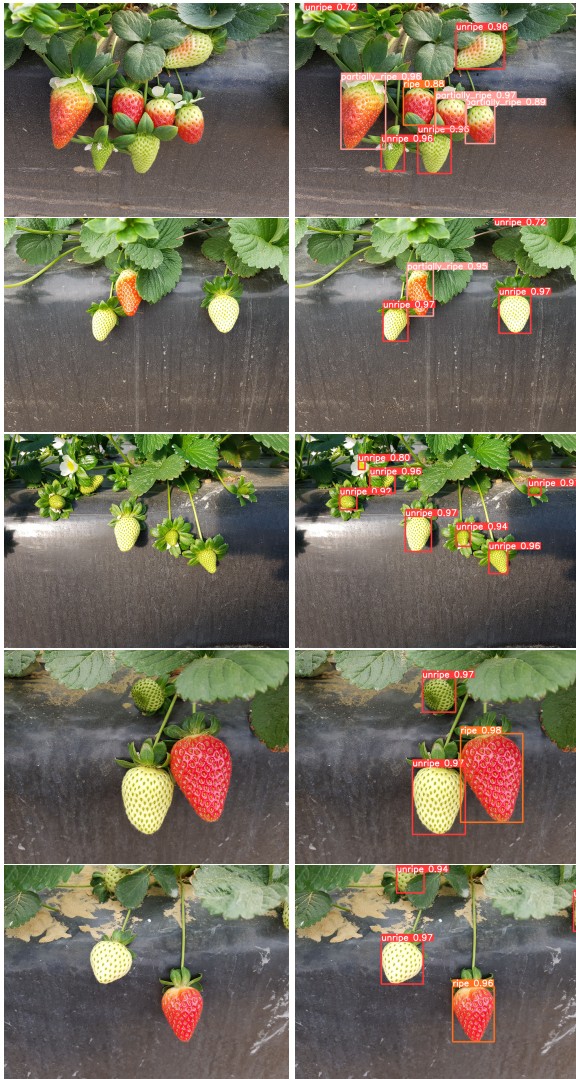


Fig. 22. Input and output images to the YOLOv5 algorithm.

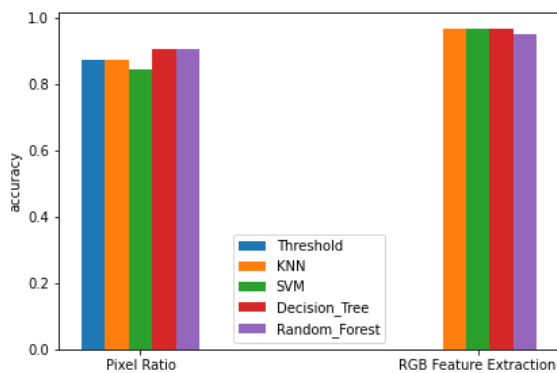


Fig. 23. Comparison of the accuracy for different classifier on the Pixel Ratio and RGB Feature extraction methods.

## 9 Conclusions

To conclude, all three methods gave outstanding overall results: 88.20% overlapping accuracy for the Mask RCNN and the results for YOLOv5 where a precision of 0.99 for both the ripeness estimation and the bounding box over-

lapping. Moreover, for the ripeness, we have an accuracy of 0.875 using the Thresholds and 0.906 using a Decision Tree with only a 157 images dataset. Finally, for the RGB Feature extraction, we have an accuracy of 0.966 on all the images. Still, depending on the resources and dataset, some methods could be more appropriate. Having a large dataset with time for the training, not wanting a segmentation but just a bounding box with the position of each strawberry, then YOLOv5 would be the best solution because of its accuracy.

On the other hand, Mask-RCNN can be trained on a much smaller dataset and still give good results because it is already pre-trained. Also, the training time is much faster, and since we have other methods to estimate the ripeness of the strawberries, we just need one class for the strawberry. Therefore, the dataset is much easier to prepare. For the ripeness estimation, both techniques give similar results with higher accuracy for the RGB feature extraction. However, it is also because it was trained on all the images, whereas the Pixel Ratio was trained on only 125 images.

## 10 Future work

In the future, several aspects of this research can be deepened and broadened to improve the performance of automated strawberry harvesting systems.

One key area of improvement is the development of more advanced imaging and feature obtention, such as hyperspectral imaging, as discussed by Shao et al. [25]. Another interesting approach discussed during the development of the experiments was the possibility of executing detection systems at image obtention time. This way, the resulting ripeness evaluation images could have much better quality.

The addition of videos could be a pivotal expansion to these systems, as having several images of each capture could allow the models to grasp different angles and spatial information. These two concepts could improve both detection and ripeness classification.

Furthermore, models could be trained and tested with larger datasets, including varying lighting conditions and day hours. The resulting systems could benefit from a larger, more representative and diverse dataset that could improve the algorithm's performance and generalisability.

Finally, further research could investigate the integration of these systems in an authentic environment for real-world testing.

## 11 REFERENCES

- [1] Siska Anraeni, Dolly Indra, Desrial Adirahmadi, Suwito Pomalingo, Sugiarti, and St. Hajrah Mansyur. Strawberry ripeness identification using feature extraction of rgb and k-nearest neighbor. pages 395–398, 2021.
- [2] Alexey Bochkovskiy, Chien Yao Wang, and Hong Yuan Mark Liao. Yolov4. *CVPR Workshop on The Future of Datasets in Vision*, 2020.

- [3] Abraham Chandy. Rgb analysis for finding the different stages of maturity of fruits in farming. *Journal of Innovative Image Processing*, 1, 2019.
- [4] Chao Cheng, Jun Fu, Hang Su, and Luquan Ren. Recent advancements in agriculture robots: Benefits and challenges. *Machines*, 11, 2023.
- [5] Hisham Cholakkal, Guolei Sun, Fahad Shahbaz Khan, and Ling Shao. Object counting and instance segmentation with image-level supervision. volume 2019-June, 2019.
- [6] Roberto J Oñoro-Rubio Daniel and López-Sastre. Towards perspective-free object counting with deep learning. pages 615–629. Springer International Publishing, 2016.
- [7] Sagi eppel. Train mask r-cnn net for object detection in 60 lines of code, 6 2022.
- [8] Youchen Fan, Shuya Zhang, Kai Feng, Kechang Qian, Yitong Wang, and Shangzhi Qin. Strawberry maturity recognition algorithm combining dark channel enhancement and yolov5. *Sensors*, 22, 2022.
- [9] Yuanyue Ge, Ya Xiong, and Pål J From. Instance segmentation and localization of strawberries in farm conditions for automatic fruit harvesting. *IFAC-PapersOnLine*, 52:294–299, 2019. 6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture AGRI-CONTROL 2019.
- [10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. pages 580–587, 2014.
- [11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2017.
- [12] Indrabayu Indrabayu, Nurhikma Arifin, and Intan Sari Areni. Strawberry ripeness classification system based on skin tone color using multi-class support vector machine. pages 191–195, 2019.
- [13] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - yolov5 sota realtime instance segmentation, 11 2022.
- [14] Daoliang Li, Zheng Miao, Fang Peng, Liang Wang, Yinfeng Hao, Zhenhu Wang, Tao Chen, Hui Li, and Yingying Zheng. Automatic counting methods in aquaculture: A review. *Journal of the World Aquaculture Society*, 52, 2021.
- [15] Tsung-Yi Lin, Michael Maire, Serge J Belongie, Lubomir D Bourdev, Ross B Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. *CoRR*, abs/1405.0312, 2014.
- [16] Tahir Mahmood, Farooq Anwar, Mateen Abbas, and Nazamid Saari. Effect of maturity on phenolics (phenolic acids and flavonoids) profile of strawberry cultivars and mulberry species from pakistan. *Int J Mol Sci*, 13:4591–4607, 4 2012.
- [17] Negar, Pinheiro Pedro O., Vazquez David, Schmidt Mark Laradji Issam H., and Rostamzadeh. Where are the blobs: Counting by localization with point supervision. pages 560–576. Springer International Publishing, 2018.
- [18] Xuebin Qin, Hang Dai, Xiaobin Hu, Deng-Ping Fan, Ling Shao, and Luc Van Gool. Highly accurate dichotomous image segmentation, 2022.
- [19] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. pages 779–788, 2016.
- [20] Joseph Redmon and Ali Farhadi. Yolov2. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua, 2016.
- [21] Joseph Redmon, Ali Farhadi, and Coco Ap. Yolov3. *Nutrition Reviews*, 36, 2018.
- [22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015.
- [23] Hannah Ritchie. Employment in agriculture: Data sources and definitions. *Our World in Data via the World bank*, 2 2022. <https://ourworldindata.org/agri-employment-sources>.
- [24] David Christian Rose and Mondira Bhattacharya. Adoption of autonomous robots in the soft fruit sector: Grower perspectives in the uk. *Smart Agricultural Technology*, 3:100118, 2023.
- [25] Yuanyuan Shao, Yongxian Wang, Guantao Xuan, Zongmei Gao, Zhichao Hu, Chong Gao, and Kaili Wang. Assessment of strawberry ripeness using hyperspectral imaging. *Analytical Letters*, 54:1547–1560, 2021.
- [26] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, 2017.
- [27] Rucha Thakur, Gaurav Suryawanshi, Hardik Patel, and Janhavi Sangoi. An innovative approach for fruit ripeness classification. pages 550–554, 2020.
- [28] Huajie Wu, Yunlai Cheng, Ruiqi Zeng, and Li Li. Strawberry image segmentation based on u2-net and maturity calculation. pages 74–78, 2022.
- [29] Chengquan Zhou, Jun Hu, Zhifu Xu, Jibo Yue, Hongbao Ye, and Guijun Yang. A novel greenhouse-based system for the detection and plumpness assessment of strawberry using an improved deep learning technique. *Frontiers in Plant Science*, 11, 2020.