Team Number:	apmcm2303758
Problem Chosen:	A

2023 APMCM summary sheet

By the rapid development of new technologies, advances in deep learning technology change the fields of image processing and object recognition. The YOLO v5 model, as a state-of-the-art real-time object detection technology, have exceptionally effective in various application domains. The aim of this paper is to use the YOLOv5 model for the rapid and accurate detection of apples, and enhances the efficiency of apple grading and quality control.

For **problem 1**, based on the feature extracted by the YOLOv5 model trained and valued by the specify test set and val set, the code generate some '.txt' files to store the features, the features contain the number of apples in the picture, the maturity of the apple and the confidence of the data. We build a mathematical model to filter those objects that are not apple and calculate the number of apples in the picture.

For **problem 2**, due to the labels and the detect labels both contain the coordinatesof the middle point of the apples in the pictures, so we just read the coordinate in the detect labels, filter those are not apples, denormalize the coordinate, which means times the coordinate with the width and height of the picture. Then draw a graph with all coordinates on it.

For **problem 3**, the maturity of apples are classified into 4 categories, we count the number of apples with its relevant categories and draw a histogram to show the distribution of all four categories.

For **problem 4**, to estimate the mass of an apple based on its area in an image, we assume each apple is spherical and appears circular in photos. The model outlines a square frame around the apple. If the model calculates the square's area as x, then the apple's volume V can be estimated. Since the actual size corresponding to the photo cannot be determined, an exact volume isn't obtainable. Therefore, we approximate the apple's mass using its estimated volume. Because of the estimation of the mass is too big, we set a 'k' parameter to lower the data.

For **problem 5**, in view of the fact that there are nearly 20,000 pictures in the attached file without labels, we do not plan to retrain yolov5 by using supervised learning methods. Our idea is to use classification algorithm to classify the fruits in Attachment 3 first to get all the apple images,

Contents

1. Introduction **1**
1.1 Problem background 1
1.2 Restatement of Problem 1
1.3 Our work 2
2. Notations **3**
3. Data Prepation **4**
3.1 Label all images 4
3.2 Normalize all images 5
3.3 Enhance image contrast 5
4. Models **7**
4.1 The Model I: Recognition of Apple Model Based on CNN 7
4.1.1 *Model Theory* 7
4.1.2 *Model Training Process* 9
4.2 The Model II: Classification of Fruit Model based on SVM 10
4.2.1 *Model Theory* 10
4.2.2 *Model Training Process* 12
5. Solutions **14**
5.1 The Solutions of Problem 1 14
5.2 The Solutions of Problem 2 15
5.3 The Solutions of Problem 3 16
5.4 The Solutions of Problem 4 18
5.5 The Solutions of Problem 5 19
6. Model Evaluation **22**
6.1 Performance Evaluation 22
6.2 Model Generalization and Anti-Interference Ability 24
6.3 Computational Efficiency 25
6.4 The Advantage and Limitations of Model 25
7. References **27**
8. Appendix **28**

I. Introduction

1.1 Problem background

China is the largest producer and exporter of apples in the world, producing 35 million tons of apples each year. However, the apple harvesting season faces a labor shortage problem, putting this traditional method into trouble. As a result, the technology of robots picking apples is gradually rising.

Figure 1 : A robot is picking apples[1]

However, robots picking technology also faces challenges in complex harvesting environments. The precise recognition of apples has already become the key in this field.

1.2 Restatement of Problem

- **Problem 1**

  Build a mathematical model for counting apples in each image, and draw the histogram to show the overall apple count in image.
- **Problem 2**

  Identify the position of each apples, and draw a two-dimensional diagram to represent the position of all apples.
- **Problem 3**

  Based on the model of counting apples, build a mathematical model to identify maturity of apples in each image. Draw a histogram to show the maturity distribution of all apples.
- **Problem 4**

  Calculate the two-dimensional area of each apples in images. And estimate the masses of the apples, displaying the mass distribution of all apples by histogram.

- **Problem 5**

  Train an apple recognition model based on the provide dataset, identifying the apples in Attachment 3. Use a histogram to represent apple images.

1.3 Our work

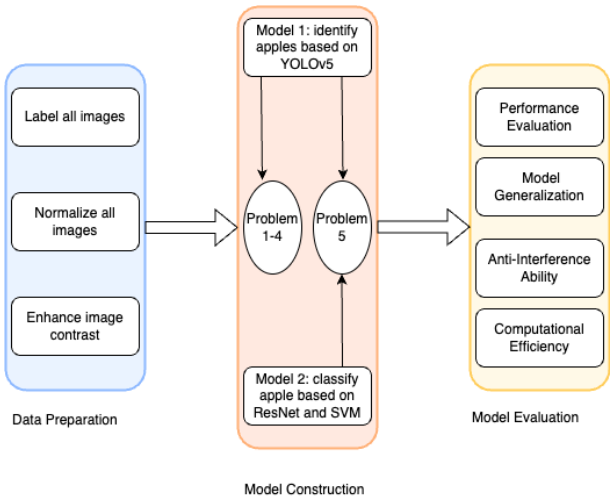

Figure 2 : The process of our works

The problem requires us to build a model to recognize apples based on provided dataset, and use this model to identify the number, maturity, position and mass of the provided figures. Therefore, we trained the model using Convolutional Neural Network method based on provided figures.

1. **Data Preparation**

   We selected some of the figures and use them to train the model. Dividing these images into training, validation, and testing sets. And label them for training input. In order to increase the performance of model, we also normalize the images to make all pixel values between 0 and 1. And the contrast of all the images is enhanced to makes the features of apples more significant.

2. **Model Construction**

   We using the prepared data to trained an image recognition model based on YOLOv5, using Convolutional Neural Networks to identify the number and position of apples in the figures, which also estimating maturity and mass of each apple. And we build classify model based on ResNet and SVM to solve Problem5.

3. **Model Evaluation**

   Evaluate the performance of model, and shows the advantages and limitations.

II. Notations

Notations	Definition
X	the single pixel value in the original image
$X_{\min}$	the min pixel value in the dataset
$X_{\max}$	the max pixel value in the dataset
S	The number of grid cells along one axis of the image
B	Number of bounding boxes each grid cell predicts in YOLO
C	Number of classes that the model is trained to detect
λ_{coord}	weight factor for box loss in the total loss function
λ_{class}	weight factor for classification loss in the total loss function
$\mathbb{I}_{ij}^{\text{obj}}$	Equal 1 if a object in jth bounding box in cell i, else 0
$\mathbb{I}_{ij}^{\text{noobj}}$	Equal 1 if no objects in jth bounding box in cell i, else 0
(x_i, y_i, w_i, h_i)	Actual bounding box parameters (center coordinates, width, height)
$(\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i)$	Predicted bounding box parameters (center coordinates, width, height)
C_i	Actual confidence score (IOU between the predicted bounding box and ground truth)
$\hat{C}_i$	Predicted confidence score for the bounding box
$p_i(c)$	Actual probability of class c in the bounding box in grid cell i
$\hat{p}_i(c)$	Predicted probability of class c in the bounding box in grid cell i
λ_{noobj}	Weight factor for no-object presence in the objectness loss
θ	The parameter vector of the model
α	The learning rate, controlling the step size in parameter updates
$\nabla_\theta L(\theta)$	The gradient of the loss function with respect to the parameter vector θ
$H(x)$	The desired output mapping of the network, often referred to as the target mapping.
$f(x)$	The learned residual mapping, representing the difference between the target output and

Table 1 Notations Table

III. Data Prepation

3.1 Label all images

The whole 200 figures dataset is divided to two parts. The first part contains 118 figures, where the number of apples is less than 12. And we will use the data of this part to generate recognition model.

For all figures in part 1, we divide these figures into training, validation, and testing sets. In order to make the training set large enough for a better performing model, we set the ratio of the test set to the other sets(including training and validation set) to be 1:9. And the ratio of the training set to validation set is 1:9.

Dataset type	Proportion
Training Set	81%
Validation Set	9%
Testing Set	10%

Table 2 The setting of dataset

We use labelImg[2] to manually mark up all the images of the first part for estimating the model.

Figure 3 : The data after labelling

For the second part, due to the limitations of manual annotation, we did not label the apples in these 82 images. But we leave the original data of this part for further testing of the model.

3.2 Normalize all images

Normalization transforms all figures data into similar ranges or distributions to facilitate model processing and training. This kind of image pre-processing can effectively improve the training efficiency and reduce bias, improving the generalization ability.

We use min-max normalization method for normalization, which scale the pixel values of the images to a range between 0 and 1. This method is achieved by using the Normalization equation.

$$\text{Normalization} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

In our dataset, the color space of all images is 8-bit, so we only need to divide the pixel values by 255(the max value for a pixel).

3.3 Enhance image contrast

In order to help the model better distinguish the features of the figures, the contrast of all image in training data is enhanced. And we use histogram equaliztion to enhance contrast. This method adjust the histogram of image and redistributes the brightness of the image. So that the histogram is evenly distributed throughout the range. The detail of enhance process is as follow:

- **Convert color space to YUV**

  The YUV color space separates the luminance information of an image from the chromaticity information, which greatly helps us to make adjustments. Therefore, the image will be converted from the BGR color space to the YUV color space at the first step.
- **Histogram Equalization**

  The histogram equalization is applied to the luminance channel, improving the global contrast of an image, especially when the data of the image is in a specific area. This method makes it more evenly distributed across all brightness by adjusting the histogram of the image.
- **Convert color space back to BGR**

The enhanced figure is converted from the YUV color space back to the BGR color space, so that the original colors of the image can be displayed normally.

- **Loop this method to other figures**

  Applying this method in other figures to enhance contrast of all dataset.

Figure 4 : The application of enhance contrast

IV. Models

4.1 The Model I: Recognition of Apple Model Based on CNN

To identify apples in images, we used the YOLOv5[3] model based on neural networks. In the following sections, I will explain in detail how the theory of model and how it was trained, and finally get the model to recognize apples.

4.1.1 *Model Theory*

- **Grid Division**

  YOLO first divides the input image into an $S \times S$ grid, which is to locate objects within the image. Each grid cell is for detecting objects whose center points fall within that cell.
- **Feature extraction**

  Using a convolutional neural network, features are extracted from each grid element.

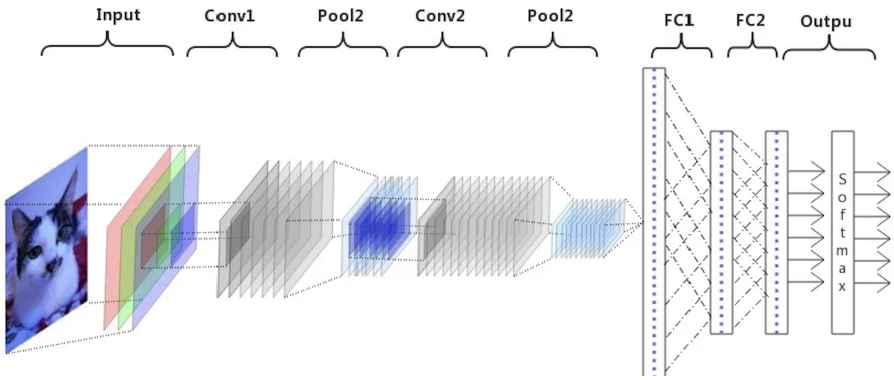

Figure 5 : The structure of CNN[4]

  These features are used for subsequent object detection and classification.
- **Category forecasting**

  Each grid element predicts the probability of C categories. These probabilities depend only on whether the mesh cell contains an object or not.
- **Output vectors**

  For each grid element, YOLO's output is a vector containing bounding box parameters, confidence, and category probability. The length of the vector is:

$$\text{Length of Output Vector} = B \times (5 + C) \tag{2}$$

- **Loss function**

  The loss function is the goal of optimization during training. In YOLO model, loss function include Box Loss, Objectness Loss, and Classification Loss.

$$\text{Total Loss} = \lambda_{\text{coord}} \cdot \text{Box Loss} + \text{Objectness Loss} + \lambda_{\text{class}} \cdot \text{Classification Loss} \tag{3}$$

  - **Box Loss**

    Box Loss focuses on the accuracy of the bounding box, including the center position and size of the box.

$$\text{Box Loss} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{I}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right] \tag{4}$$

  - **Objectness Loss**

    The objectness loss quantifies the confidence level at which an object is included within the predicted bounding box.

$$\text{Objectness Loss} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \left[\mathbb{I}_{ij}^{\text{obj}} \cdot (C_i - \hat{C}_i)^2 + \lambda_{\text{noobj}} \cdot \mathbb{I}_{ij}^{\text{noobj}} \cdot (C_i - \hat{C}_i)^2 \right] \tag{5}$$

  - **Classification Loss**

    Classification loss focuses on correctly classifying the objects within each bounding box.

$$\text{Classification Loss} = \sum_{i=0}^{S^2} \mathbb{I}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \tag{6}$$

- **Minimize the loss function**

  The gradient descent algorithm is used during YOLO training to minimize the loss function. By calculating the gradient of the loss function relative to the network parameters, the model parameters are gradually adjusted to reduce the total loss.

$$\theta_{\text{next}} = \theta_{\text{current}} - \alpha \nabla_{\theta} L(\theta) \tag{7}$$

  At the same time, the parameters of different parts of the model are adjusted to accurately predict the position of the object.

4.1.2 *Model Training Process*

- **Model Input**

  After entering the preprocessed data, image data is converted into tensors, which are multi-dimensional arrays that can be solved by pytorch. Moreover, image tensors are normalized, including subtracting the mean and dividing by the standard deviation.

- **Forward Propagation**

  The processed data is passed to the YOLOv5 model. The model's convolutional layers, activation functions, and other components start processing the data, extracting features, and making predictions.

- **Loss Calculation**

  Calculate the difference between the model output and the real annotated data. Classification Loss, Objectness Loss and Bounding Box Loss are combined to form the total loss.

- **Backward Propagation and Weight Update**

  The gradient of the loss function relative to the model parameters is calculated using the optimization algorithm update the weights according to the gradient to reduce the total loss.

- **Iteration and Validation**

  Repeat the training process several times until the model performance reaches a satisfactory level. Model performance is also evaluated regularly to monitor and prevent overfitting

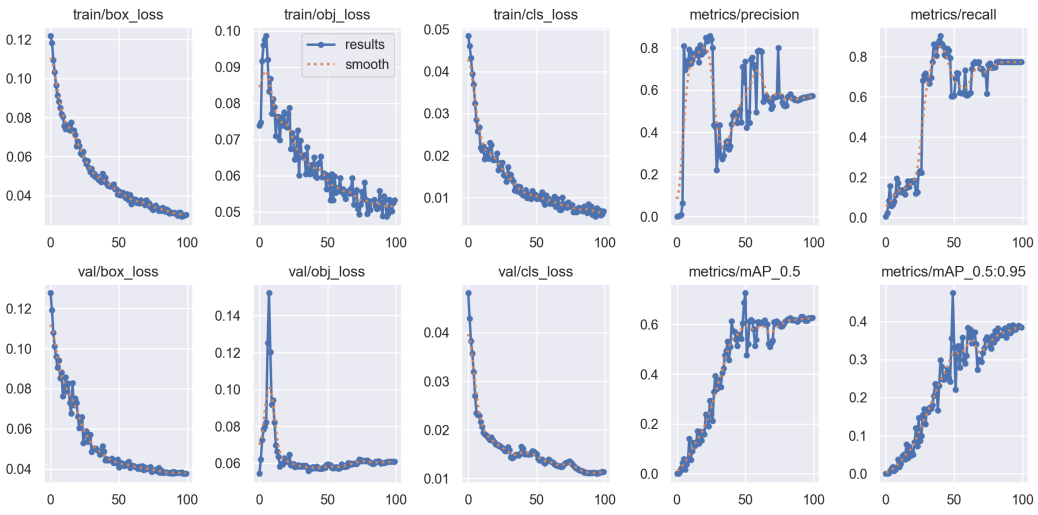

Figure 6 : The training process.

Listing 1: YOLOv5 Training Core Steps - Forward Propagation, Backpropagation, and Optimization

```
# Forward Propagation
with torch.cuda.amp.autocast(amp):
    pred = model(imgs) # Forward pass, get predictions
    loss, loss_items = compute_loss(pred, targets.to(device)) #
        Compute loss

# Backpropagation
scaler.scale(loss).backward() # Scale the loss and backpropagate

# Optimization
if ni - last_opt_step >= accumulate:
    scaler.unscale_(optimizer) # Unscale gradients
    torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=10.0)
        # Clip gradients
    scaler.step(optimizer) # Update model weights
    scaler.update() # Update scaler state
    optimizer.zero_grad() # Clear gradients
    if ema:
        ema.update(model) # Update Exponential Moving Average model
```

4.2 The Model II: Classification of Fruit Model based on SVM

4.2.1 *Model Theory*

Residual Network:

Residual Network[5] is a type of deep neural network architecture that was designed to address the challenges of training very deep neural networks. In this network, we use a technique called skip connections. The skip connection connects activations of a layer to further layers by skipping some layers in between. This forms a residual block. Resnets are made by stacking these residual blocks together.

The approach behind this network is instead of layers learning the underlying mapping, we allow the network to fit the residual mapping. So, instead of say H(x), initial mapping, let the network fit. So called shortcut connection.

The function as below, x is the input of function.

$$H(x) = f(x) + x \tag{8}$$

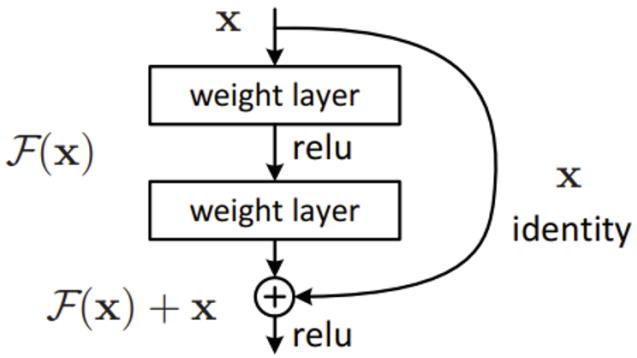

Figure 7 Residual Learning[6]

In summary, ResNet's ability to effectively train deep networks, capture complex features, and leverage pre-trained models for transfer learning makes it a strong choice for feature extraction in tasks like fruit image recognition.

Support Vector Machine:

The goal of the support vector machine algorithm is to find a hyperplane in a N -dimensional space N - the number of features that clearly classifies data points.

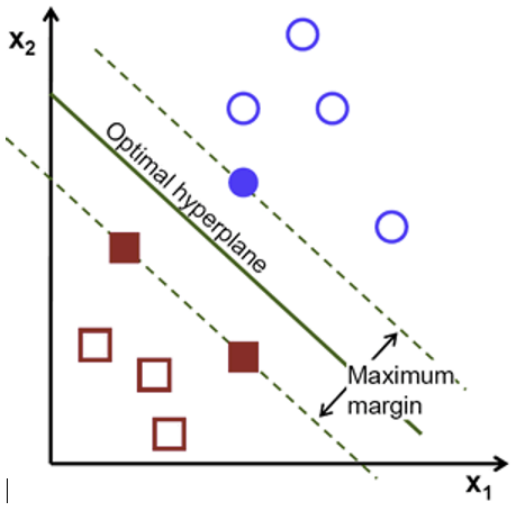

Figure 8 Optimal Hyperplane[7]

To separate the two classes of data points, there are many possible hyperplanes that could be chosen. The objective of SVM is to find a plane that has the maximum margin,

i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

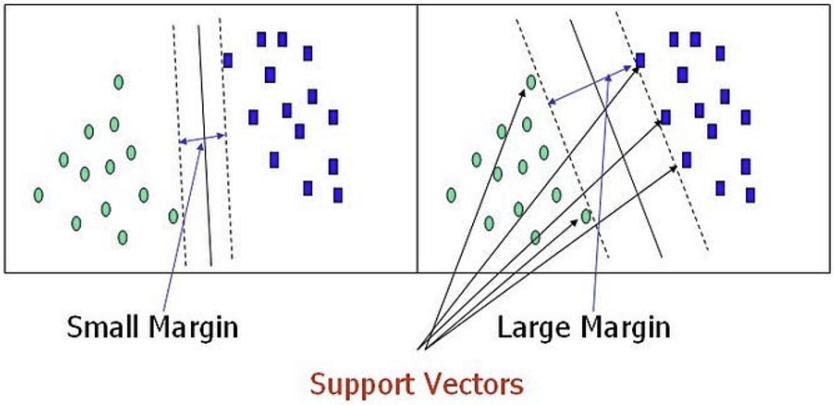

Figure 9 Support Vectors[7]

Considering our dataset has the characteristics are non-linear, relatively small and imbalanced (since in attachment2, the apple class images have over 10 thousand pictures, while other class only have about 2 thousand). Therefore, SVM provide a robust option for semi-supervised fruit image classification.

4.2.2 *Model Training Process*

- **Semi-Supervised Learning and Feature Extraction**

  The model initially employs semi-supervised learning, using a Residual Neural Network to extract features from images. ResNet solves the training problem of deeper networks by introducing ”residual learning”.
- **Support Vector Machine Training**

  The images in Attachment 2 are trained using Support Vector Machine. And the labels are combined with the image feature vectors to train the SVM model. SVM is a supervised learning algorithm, finding a hyperplane that maximizes the margin between different categories of data points.
- **Generating Pseudo Labels**

  The trained SVM model is applied to the unlabeled dataset Attachment 3 to generate pseudo-labels. This step is to maximize the use of unlabeled data and provide predictive labels to assist subsequent model training.

- **Merging Labeled and Pseudo-Labeled Samples**

  In order to enhance the data of model training, the pseudo-labels of labeled samples and unlabeled samples are merged to form a new training set.
- **Model Retraining**

  Retrain the SVM model with a new, pseudo-labeled dataset.
- **Reclassification and Result Saving**

  The newly unlabeled samples in Attachment 3 were classified using the retrained model, and the samples were saved to the corresponding folder based on the prediction results

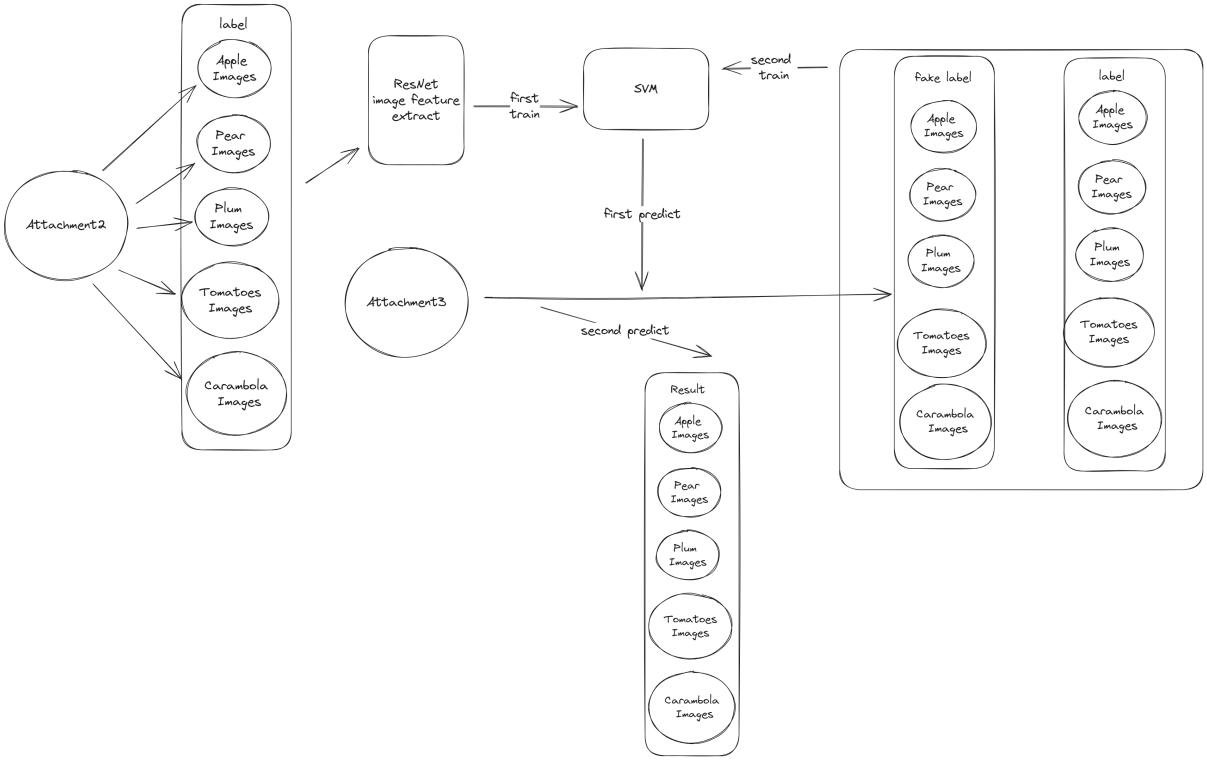

Figure 10 Training Process

V. Solutions

In this section, we put the above mathematical model into practice and solve several problems required problem.

5.1 The Solutions of Problem 1

We put the YOLOv5 model that trained by us to identify the apples. Based on this trained Neural network model, apples in each figures can be recognized, all the apples are marked with a square frame.

Figure 11 : Apples are marked

The model also counts the number of apples in each image. Based on this model, we draw a distribution chart of all the apples, depicting the number of apples in each image.

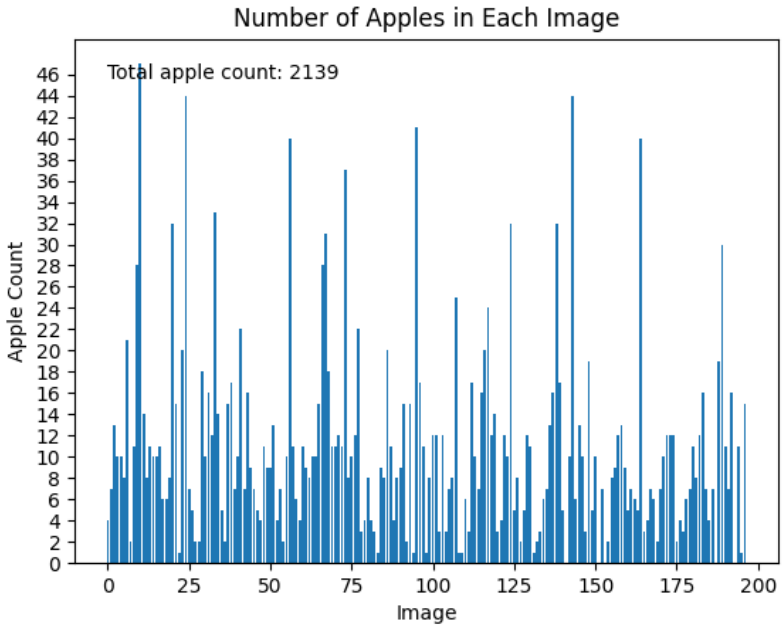

Figure 12 : Number of apple in each figure

5 . 2 The Solutions of Problem 2

The model divides all apples into four types: mature, immature, semi-mature, and Flowers and Other. We draw histograms of four levels of maturity.

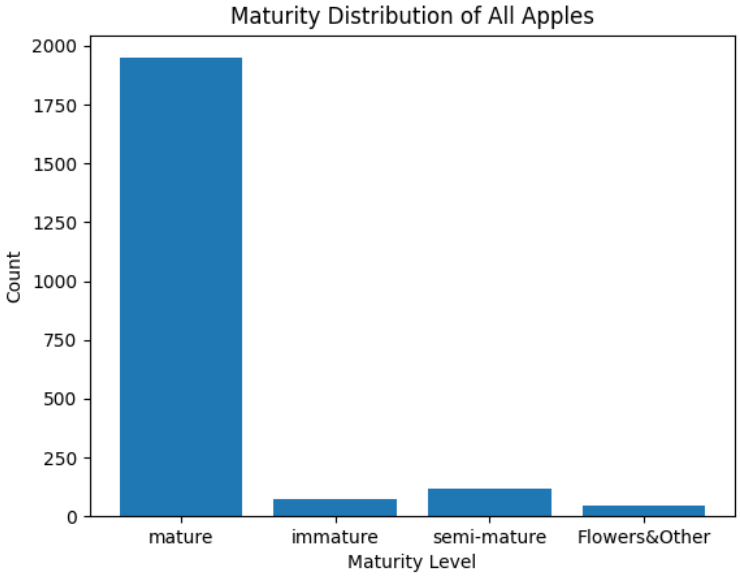

Figure 13 : Maturity Distribution of All Apples

It is clear that ripe apples account for the majority in these two hundred pictures.

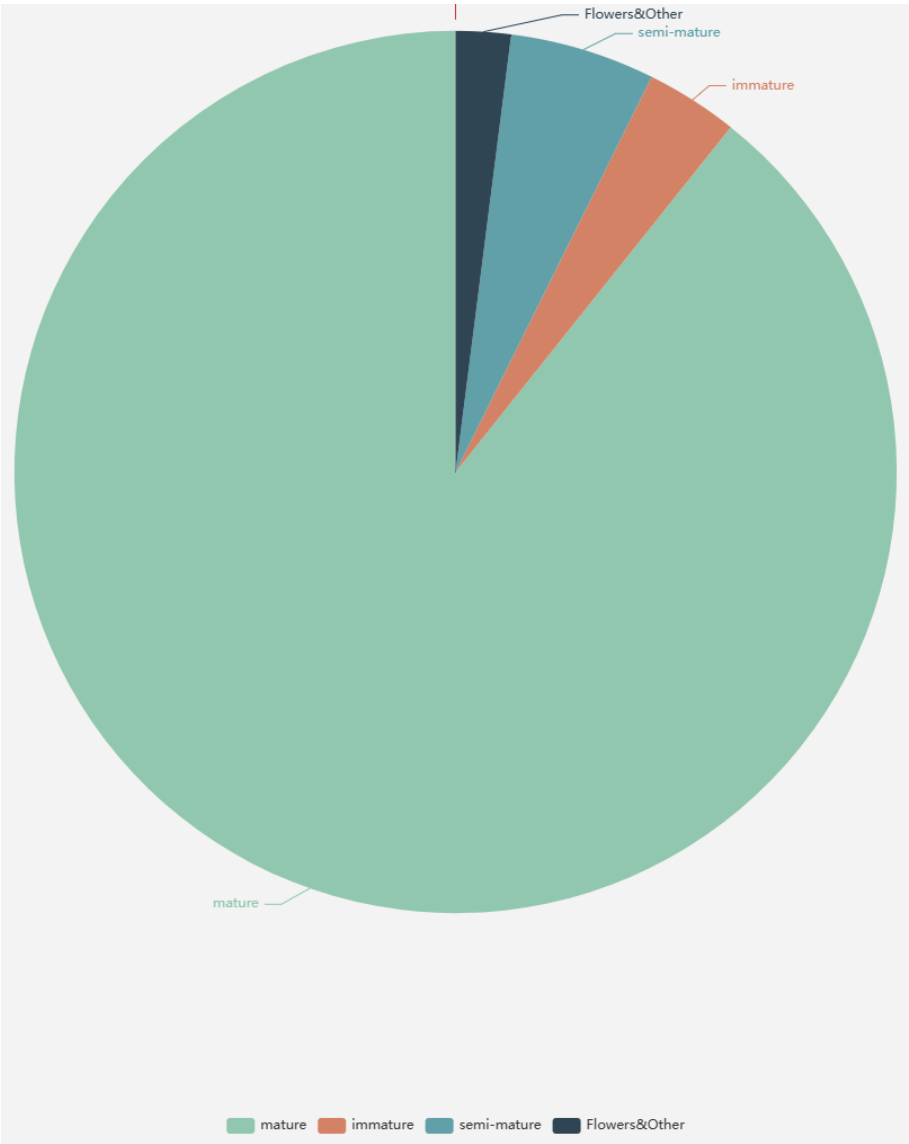

Figure 14 : Pie chart of apple maturity levels

5.3 The Solutions of Problem 3

The centroid of the apple is considered as its position in the image.

The centroid of an object in figure:

$$x_{\text{centroid}} = \frac{\sum x_i \times w_i}{\sum w_i} \tag{9}$$

$$y_{\text{centroid}} = \frac{\sum y_i \times w_i}{\sum w_i} \tag{10}$$

Plot the positions of all apples in the dataset as a coordinate plot:

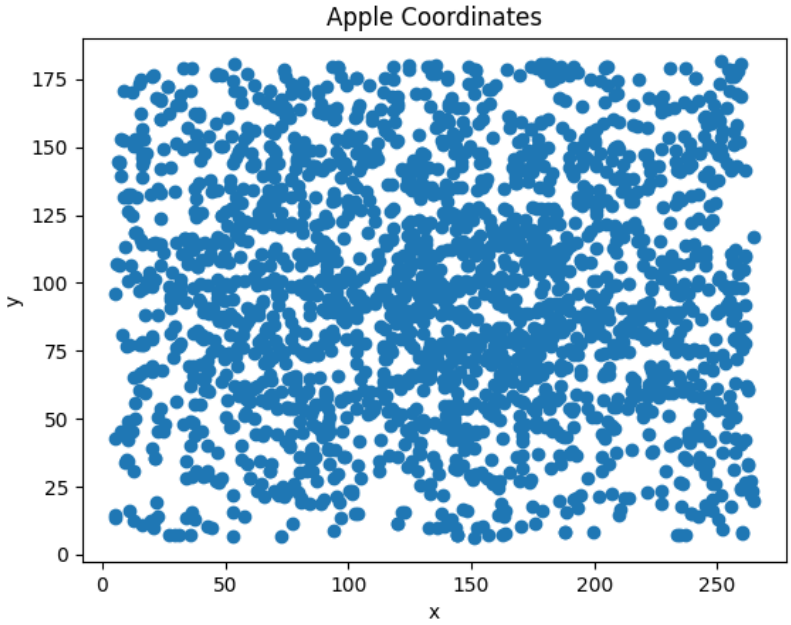

Figure 15 : Apple Coordinates

However, the x-axis and y-axis of this chart are normalized pixel values. Reverting this normalization, we can obtain the following coordinate plot:

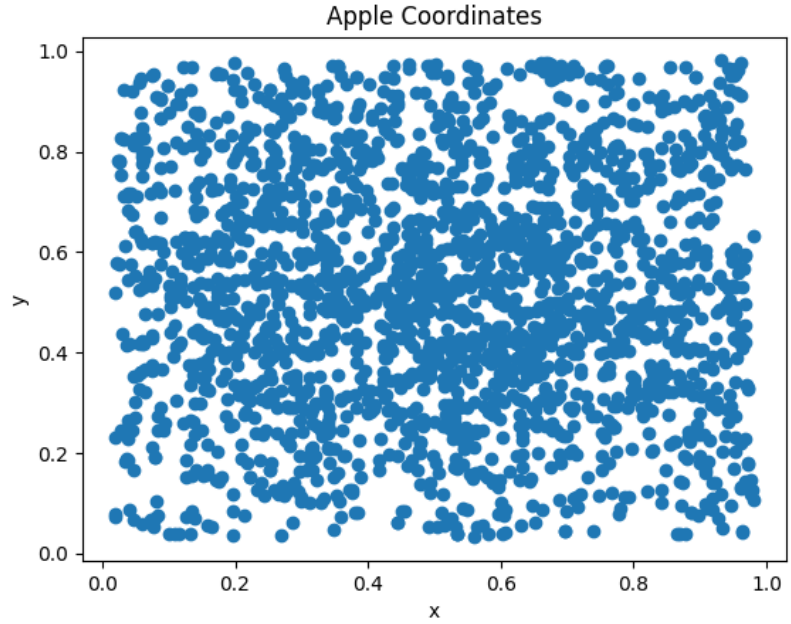

Figure 16 : Apple Coordinates

We also drew the following heat map based on the location of Apple.

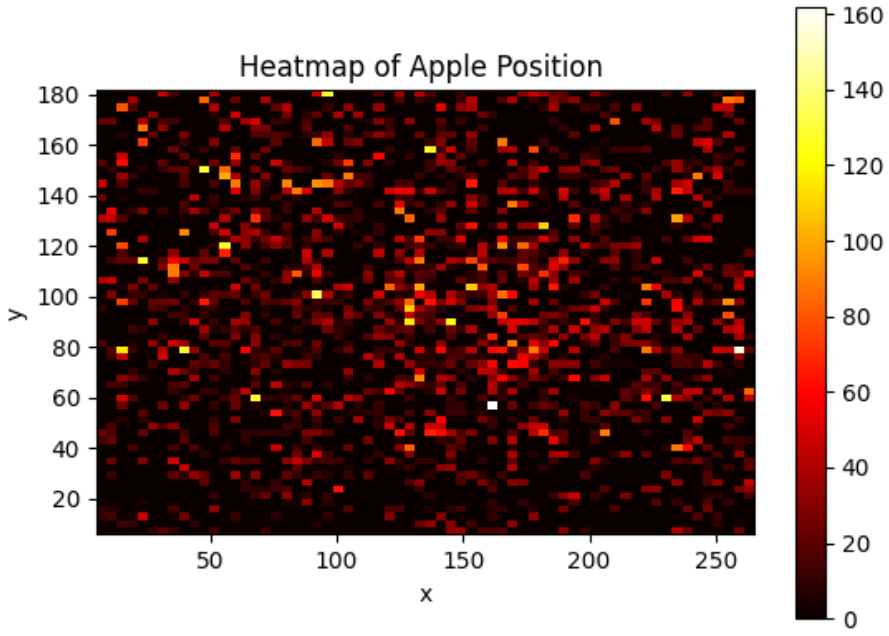

Figure 17 : Heatmap of apple positions

5.4 The Solutions of Problem 4

Estimating the mass of an apple based on the area it occupies in the image. We assume all apples are spherical and the apple in the photo is circular. The model marks a square frame around the apple. Given that the model provides data that can be used to calculate the area of the square frame as x, then the volume V of the apple can be estimated as

$$V = \frac{\pi x \sqrt{\frac{\pi x}{4}}}{3} \tag{11}$$

Since it's impossible to determine the actual size corresponding to the photo, an exact volume can't be obtained. Therefore, we use the estimated volume of the apple to approximate its mass

So we can draw the figure of apples mass distribution.

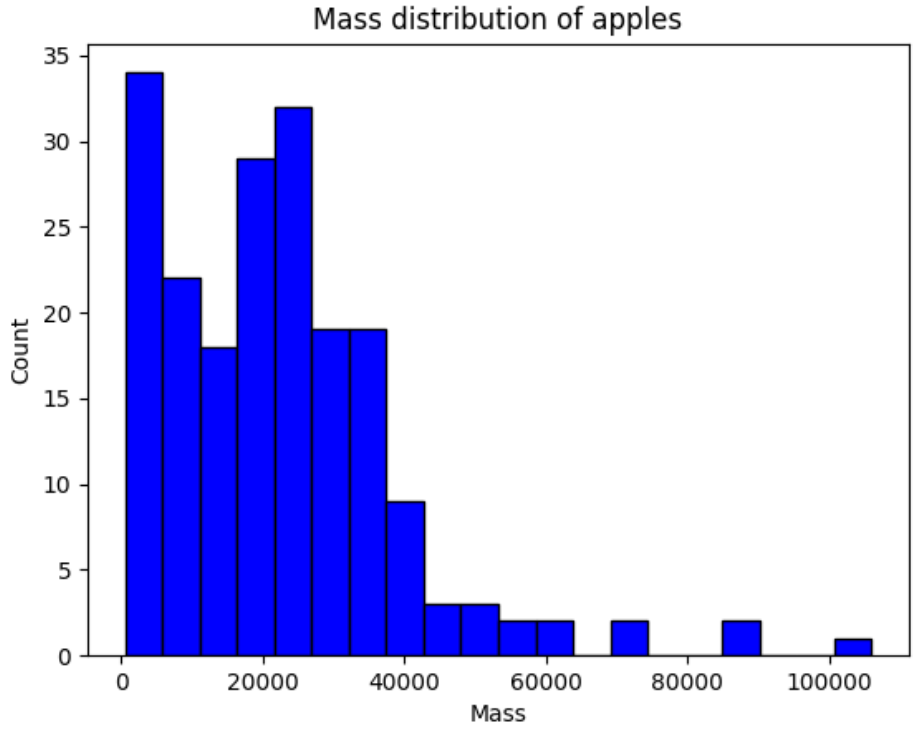

Figure 18 : Mass distribution of apples

5.5 The Solutions of Problem 5

To solve problem 5, we intend to directly use the trained YOLOv5 model to identify the fruits in Annex 3 at the beginning. However, the recognition effect was poor, and the model would identify tomatoes and other fruits as apples of different ripeness, resulting in errors in the final statistics.

The following figure shows the identification result of YOLOv5:

Figure 19 : The Shortcomings of Using Only the YOLO Model

The trained model pair has a poor experimental effect on the whole. It can recognize Tomatoes and Plum as apples, but it can accurately identify the number and maturity of apples in the picture of apples.

The reason of failure of YOLO model

1. The yolov5 model trained in the previous questions is trained by the pictures in Attachment 1, which are mainly the photos of apples still on the tree in the farm, while the pictures in
2. The yolov5 model trained before is trained to answer the first four questions, and has no function ofdistinguishing Tomatoes and other four kinds of fruits except apples.

Therefore, the previously trained model cannot be applied to question 5.

In view of the fact that there are nearly 20,000 pictures in the attached file without labels, we do not plan to retrain yolov5 by using supervised learning methods. Our idea is to use classification algorithm to classify the fruits in Annex 3. Semi-supervised learning algorithm, SVM model for classification task and ResNet residual neural network for image feature extraction method are adopted.

Compared with other classification algorithms, SVM has the characteristics of strong generalization ability and good performance in high-dimensional space, so we choose it for classification task.

For ResNet, the reason why we choose it for image feature extraction is that it introduces residual learning to solve the problem of gradient disappearance, and can efficiently shorten the training time and achieve good accuracy while training deeper networks. After comparison, considering the limited training time and computing resources, after comparison, ResNet50 is strong enough, and the use of deeper ResNet100 may not bring significant performance improvement, we finally choose ResNet-50, and only includes the convolutional layer and the full connection layer, excluding the pooling layer, exactly 50 layers.

Training results: From the 20,705 fruit pictures, 11,240 Apple pictures, 2,078 Carambola pictures, 2,924 Pear pictures, 2,297 Plum pictures and 2,166 Tomatoes pictures were classified The classification effect diagram is as follows:

Finally, the trained yolov5 model is used to identify the number of apples in the classified apple folder. The results are as follows:

It can be seen that even if there is a human hand in the picture, the number of apples in the picture can still be accurately identified, so on the whole, the cost of the problem requirements are met, and the completion effect is excellent.

Figure 20 : Identify the apple in Attachment 3

Number of apples square statistics chart:

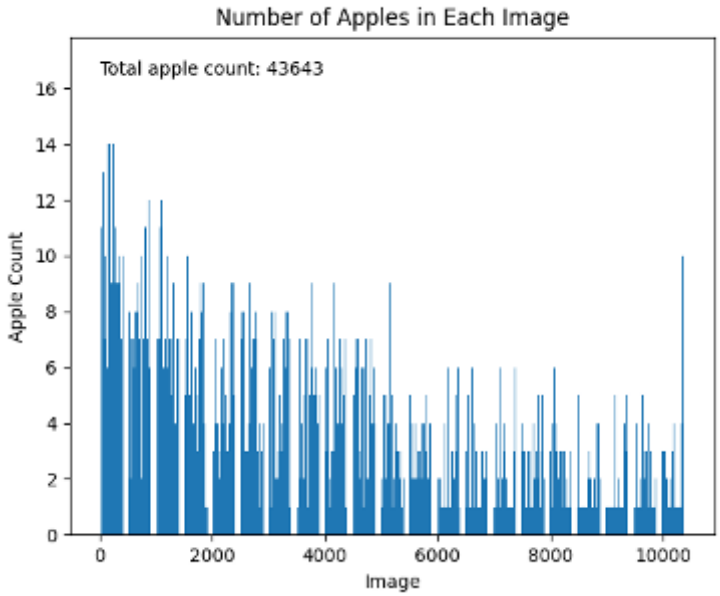

Figure 21 : Number of Apples in Each Image

In conclusion, for all the pictures in Attachment 3, the total number of apples is 43,643.

VI. Model Evaluation

6.1 Performance Evaluation

We use the F1-Confidence Curve to show the F1 scores of the model under different Confidence thresholds. The F1-Confidence Curve indicate the relationship between the F1 score for different categories Mature, Immature, and Semi-mature and the confidence threshold. The F1 score is the harmonic mean of precision and recall and is used to represent the overall performance of a model.

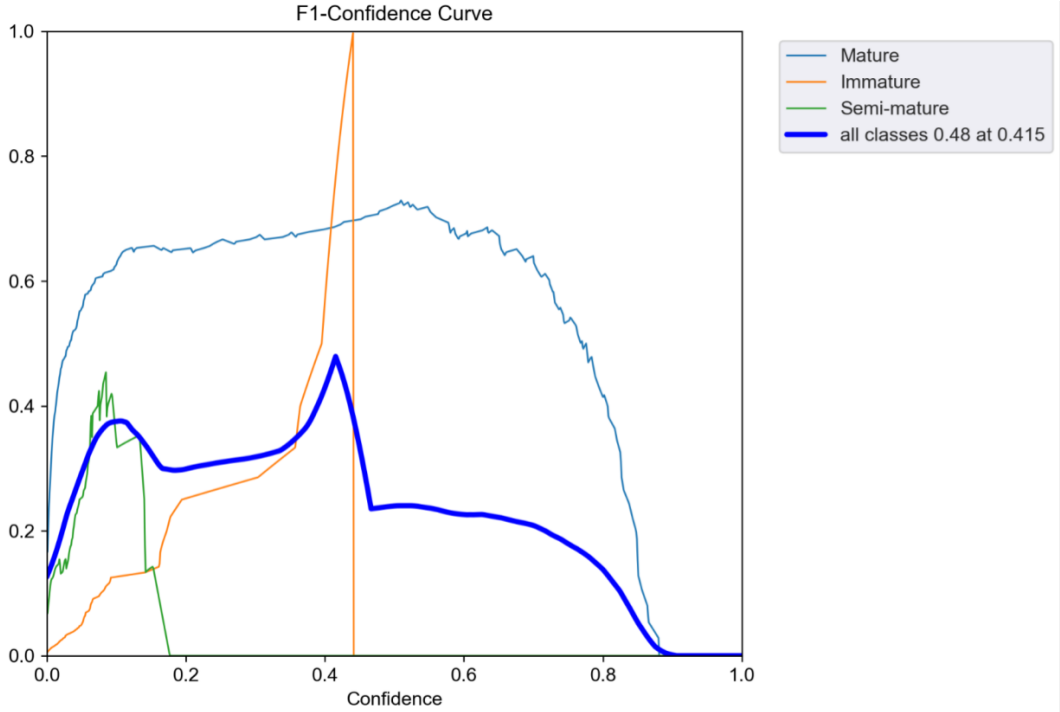

Figure 22 : F1-Confidence Curve

- **Mature**:

  The F1 score for this category increases with the confidence threshold, reaches a peak, and then begins to decline. This suggests there is an optimal confidence threshold at which the model's predictions for the mature category are both precise and have a high recall rate.

- **Immature**:

  For the immature category, the F1 score is higher at lower confidence thresholds and then rapidly decreases, indicating that the model performs well in predicting this category at lower confidence levels.

- **Semi-mature**:

  The F1 score for the semi-mature category has several peaks at low confidence thresholds, then gradually decreases and stabilizes. These peaks may indicate that the model has high confidence in certain specific samples or features.

- **All classes**:

  The combined F1 score for all categories reaches 0.48 at a confidence threshold of about 0.415. This may be the optimal confidence threshold for the best overall performance of the model across all categories.

We also use the Precision-Recall Curve, which can demonstrate the relationship between the model's precision and recall.

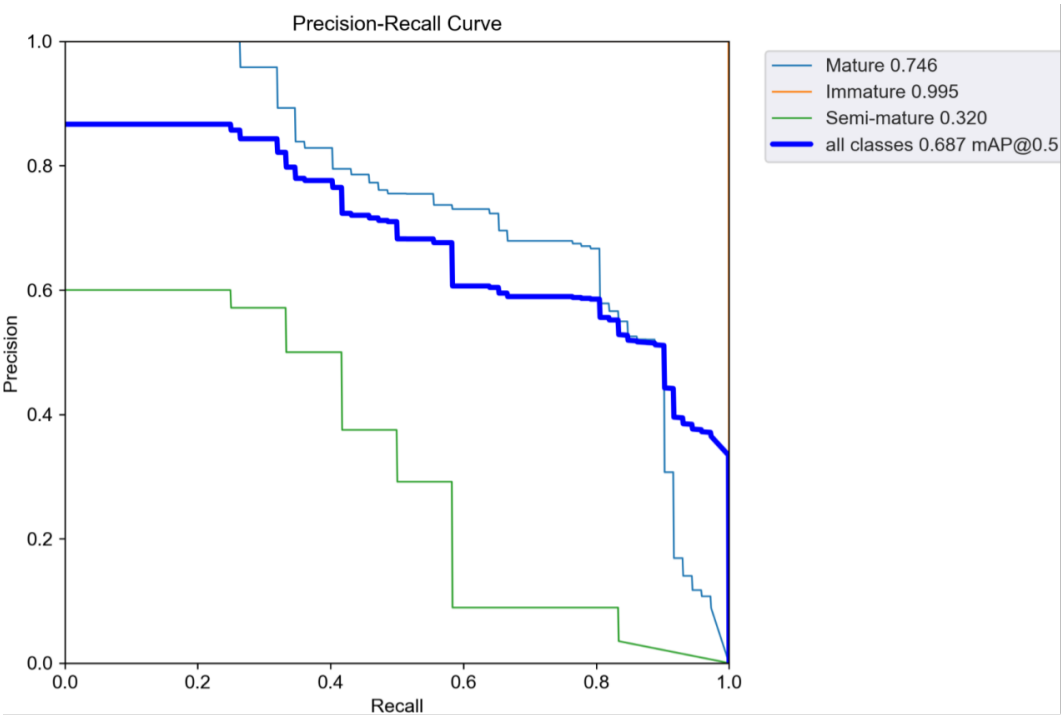

Figure 23 : Precision-Recall Curve

- **Mature**:

  It has a maximum Average Precision (max AP) of 0.746, indicating relatively high precision at certain recall levels, but the precision decreases as recall increases. This suggests that the model can detect samples of the mature category with high precision but may produce some false positives when attempting to detect more samples.

- **Immature**:

  It has a maximum Average Precision of 0.995, with high precision across most levels of recall and very little decrease in precision as recall increases. This indicates that the model's predictions for the immature category are both accurate and reliable.

- **Semi-mature**:

  It has a maximum Average Precision of 0.320, with low precision across the range of recall, which may mean the model has a higher number of false positives when predicting the semi-mature category.

- **All classes**:

  Overall, the model has a mean average precision of 0.687 at a confidence threshold of 0.5, which indicate that the model have a good overall performance.

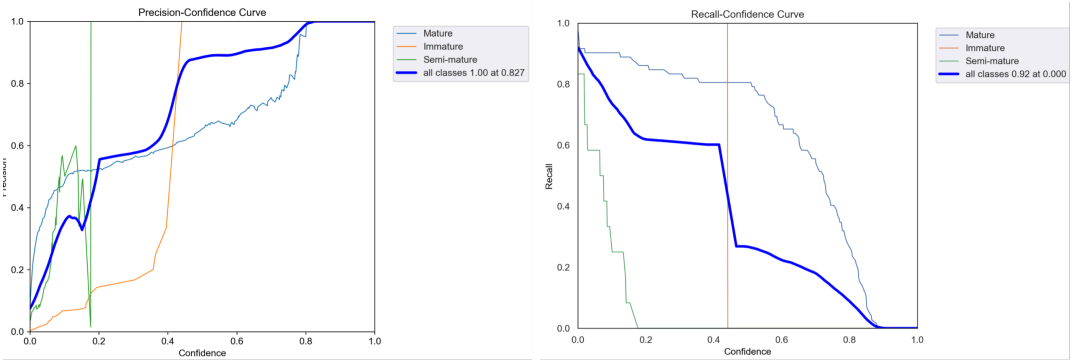

Figure 24 : The Precision-Confidence Curve and Recall-Confidence Curve

6.2 Model Generalization and Anti-Interference Ability

Through our evaluation, we have found that when a large number of apples in an image and apples are closely packed, the model's prediction accuracy and efficiency are low. But in models with fewer apples, the accuracy of the model is very high. Therefore, the generalization ability of model is low.

For Anti-Interference ability, even if there is an object blocking the apple on the picture, the model can still identify and mark the apple, which shows that the model has strong anti-interference ability

Figure 25 : Anti-Interference ability

6.3 Computational Efficiency

The average training time is about two hours in successful model training. The training set contains 100-200 images, and a total of 100 epochs are performed to optimize the weights. The average processing prediction time of each image is in milliseconds (20ms-30ms).

Therefore, Using this model requires very minimal computational resources.

6.4 The Advantage and Limitations of Model

- **Advantages**
  1. **Performance Metric**
     F1 scores at different confidence thresholds indicate a well-tuned model. For the mature apple category, the F1 score peaks before declining, indicating a good balance between precision and recall. The model performs well when immature classes have low confidence, while semi-mature classes show higher confidence in specific samples
  2. **Computational Efficiency**
     The model requires less computing resources and can be easily run even on a personal computer
  3. **Model accuracy**
     The model is very accurate in identifying apples, especially mature apples.

4. **Anti-Interference Capability**

   The model demonstrates strong anti-interference abilities. It can still identify and mark apples even when they are partially obstructed in the image.

- **Limitations**

1. **Generalization Ability**

   The accuracy and efficiency of the model decrease when processing images containing a large number of closely packed apples.

2. **Fewer Dataset**

   This results in models with insufficient predictive capabilities when encountering complex situations.

VII. References

[1] 郭玉侠. (2021). 快看！摘苹果机器人来洛川啦～ https://new.qq.com/rain/a/20211105A07YSV00

[2] Tzutalin. (2022). labelImg. https://github.com/HumanSignal/labelImg

[3] Ultralytics. (2020). YOLOv5. GitHub repository. https://github.com/ultralytics/yolov5

[4] 西岩寺往事. (2023). 卷积神经网络（CNN）基础及经典模型介绍. https://zhuanlan.zhihu.com/p/344562609

[5] 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 (CVPR), 770-778.

[6] Reconcile. (2023). ResNet（深度残差网络）原理及代码实现（基于Pytorch）. https://zhuanlan.zhihu.com/p/589762877

[7] Rohith Gandhi. (2018). Support Vector Machine — Introduction to Machine Learning Algorithms. https://towardsdatascience.com

VIII. Appendix

- **The process of YOLOv5 training:**

epoch	train/box_loss	train/obj_loss	train/cls_loss	metrics/precision	metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.95	val/box_loss	val/obj_loss	val/cls_loss	x/lr0	x/lr1	x/lr2
0	0.121840	0.073831	0.048430	0.000588	0.004630	0.000393	0.000039	0.127750	0.054386	0.048142	0.095500	0.000500	0.000500
1	0.118210	0.074731	0.046059	0.002196	0.023148	0.001555	0.000317	0.119160	0.061984	0.042925	0.090089	0.001089	0.001089
2	0.109520	0.091679	0.043227	0.005386	0.083333	0.007085	0.001636	0.107950	0.072730	0.038260	0.084666	0.001666	0.001666
3	0.103330	0.096066	0.039378	0.009237	0.157410	0.022384	0.006978	0.101060	0.078247	0.035684	0.079232	0.002232	0.002232
4	0.096610	0.097473	0.036869	0.063469	0.056633	0.042704	0.012478	0.096065	0.079765	0.031944	0.073785	0.002785	0.002785
5	0.091420	0.098578	0.032373	0.808470	0.064815	0.059138	0.017769	0.090616	0.082310	0.027046	0.068327	0.003327	0.003327
6	0.088136	0.092050	0.028505	0.696630	0.078704	0.017087	0.007121	0.094140	0.125300	0.023508	0.062856	0.003857	0.003857
7	0.084933	0.083229	0.025776	0.738030	0.111110	0.038633	0.011224	0.085328	0.152260	0.023126	0.057374	0.004374	0.004374
8	0.081905	0.086790	0.026678	0.719020	0.194440	0.057879	0.015913	0.088138	0.120230	0.022939	0.051880	0.004880	0.004880
9	0.080540	0.077110	0.021912	0.791890	0.172730	0.140880	0.037078	0.076180	0.091648	0.021666	0.046374	0.005374	0.005374
10	0.075753	0.081223	0.021270	0.725330	0.134260	0.055446	0.017772	0.080715	0.094045	0.020634	0.040857	0.005856	0.005856
11	0.074072	0.078926	0.022261	0.741810	0.125000	0.091484	0.025604	0.085727	0.082207	0.019518	0.035327	0.006327	0.006327
12	0.076169	0.070961	0.019155	0.773700	0.135760	0.125790	0.047012	0.078701	0.069707	0.019074	0.029785	0.006785	0.006785
13	0.074045	0.076340	0.019561	0.764070	0.120370	0.112540	0.050189	0.082643	0.066328	0.018967	0.024232	0.007232	0.007232
14	0.073474	0.076323	0.021418	0.777740	0.115740	0.125050	0.037910	0.072870	0.063455	0.018696	0.018666	0.007666	0.007666
15	0.077486	0.069891	0.019320	0.794040	0.148150	0.172020	0.076373	0.067589	0.058154	0.018299	0.013089	0.008089	0.008089
16	0.073274	0.076137	0.023043	0.730550	0.180560	0.121500	0.040495	0.082851	0.059748	0.017936	0.008416	0.008416	0.008416
17	0.073524	0.073822	0.022550	0.812730	0.152780	0.169880	0.071058	0.071936	0.059437	0.018522	0.008416	0.008416	0.008416
18	0.071185	0.077503	0.019009	0.768470	0.180560	0.134430	0.046103	0.075134	0.062921	0.017827	0.008317	0.008317	0.008317
19	0.065321	0.073759	0.018939	0.794400	0.181220	0.167240	0.059491	0.072843	0.061842	0.017454	0.008218	0.008218	0.008218
20	0.067788	0.073367	0.019263	0.782480	0.175930	0.158590	0.050062	0.066060	0.061113	0.017036	0.008119	0.008119	0.008119
21	0.065573	0.077054	0.020521	0.848900	0.184280	0.238470	0.082272	0.060303	0.060533	0.017025	0.008020	0.008020	0.008020
22	0.061708	0.078775	0.016506	0.832590	0.119320	0.231840	0.119630	0.063587	0.064799	0.016793	0.007921	0.007921	0.007921
23	0.060936	0.067384	0.018220	0.844190	0.125000	0.190400	0.072744	0.065884	0.059075	0.016749	0.007822	0.007822	0.007822
24	0.062536	0.071822	0.016818	0.856290	0.222220	0.294330	0.146090	0.052904	0.058898	0.016485	0.007723	0.007723	0.007723
25	0.058312	0.070564	0.018280	0.840450	0.231480	0.240720	0.086274	0.056896	0.059689	0.016276	0.007624	0.007624	0.007624
26	0.056246	0.064533	0.015372	0.799700	0.222220	0.212740	0.098881	0.058933	0.058123	0.016038	0.007525	0.007525	0.007525
27	0.057796	0.065222	0.015136	0.434210	0.681840	0.329460	0.155040	0.053529	0.058158	0.015649	0.007426	0.007426	0.007426
28	0.054142	0.072499	0.013788	0.425290	0.706920	0.392320	0.170040	0.056021	0.058891	0.015702	0.007327	0.007327	0.007327
29	0.052054	0.060030	0.017284	0.220450	0.717590	0.339930	0.130000	0.057001	0.058361	0.016451	0.007228	0.007228	0.007228
30	0.053674	0.069660	0.015191	0.294090	0.693350	0.345690	0.168740	0.048575	0.058737	0.015836	0.007129	0.007129	0.007129
31	0.050574	0.065428	0.015524	0.433590	0.693660	0.376930	0.160970	0.050235	0.058278	0.014439	0.007030	0.007030	0.007030
32	0.051180	0.065059	0.014328	0.289640	0.666670	0.349470	0.172820	0.050354	0.058779	0.014170	0.006931	0.006931	0.006931
33	0.049480	0.065721	0.013841	0.272200	0.694440	0.405010	0.171330	0.050351	0.058200	0.014360	0.006832	0.006832	0.006832
34	0.049965	0.060454	0.011720	0.289000	0.772970	0.422600	0.180340	0.049603	0.058663	0.014754	0.006733	0.006733	0.006733
35	0.047953	0.067854	0.012663	0.329260	0.861110	0.476260	0.204450	0.048852	0.058093	0.014897	0.006634	0.006634	0.006634
36	0.047719	0.060760	0.011490	0.354800	0.861110	0.494650	0.235970	0.047065	0.055918	0.014861	0.006535	0.006535	0.006535
37	0.046870	0.060444	0.012679	0.359800	0.805560	0.530880	0.235230	0.048024	0.057590	0.014787	0.006436	0.006436	0.006436
38	0.051333	0.062978	0.012592	0.318570	0.856480	0.456880	0.166270	0.051430	0.057100	0.015158	0.006337	0.006337	0.006337
39	0.047831	0.060835	0.010971	0.334580	0.879630	0.509660	0.231530	0.048713	0.058128	0.016585	0.006238	0.006238	0.006238
40	0.049174	0.065033	0.010610	0.438730	0.902780	0.612370	0.298780	0.044335	0.057787	0.016152	0.006139	0.006139	0.006139
41	0.046217	0.059862	0.011273	0.480240	0.864660	0.574110	0.281240	0.045058	0.057311	0.014999	0.006040	0.006040	0.006040
42	0.045082	0.059454	0.011802	0.492310	0.852710	0.570210	0.247940	0.044371	0.058320	0.015002	0.005941	0.005941	0.005941
43	0.044826	0.063195	0.009958	0.477060	0.808630	0.540460	0.247150	0.044525	0.057321	0.015028	0.005842	0.005842	0.005842
44	0.044596	0.063654	0.010712	0.445360	0.801370	0.513810	0.263590	0.044291	0.057027	0.014982	0.005743	0.005743	0.005743
45	0.045010	0.065345	0.010807	0.490430	0.839020	0.559490	0.275120	0.045211	0.057013	0.015032	0.005644	0.005644	0.005644
46	0.045888	0.059651	0.009737	0.508050	0.829880	0.549900	0.251820	0.043029	0.057110	0.015032	0.005545	0.005545	0.005545
47	0.043209	0.060725	0.011629	0.447060	0.794210	0.542430	0.242020	0.043452	0.057698	0.015620	0.005446	0.005446	0.005446
48	0.043288	0.060503	0.010922	0.710940	0.601850	0.604100	0.354910	0.042996	0.057441	0.015547	0.005347	0.005347	0.005347
49	0.040580	0.056266	0.010729	0.636390	0.601850	0.688230	0.474590	0.044679	0.059196	0.015331	0.005248	0.005248	0.005248
50	0.042075	0.059853	0.009694	0.735820	0.606480	0.726380	0.331340	0.040865	0.059338	0.015022	0.005149	0.005149	0.005149
51	0.040158	0.060116	0.010316	0.422340	0.689810	0.476620	0.221210	0.042777	0.058949	0.014142	0.005050	0.005050	0.005050
52	0.040881	0.053251	0.009859	0.494410	0.719570	0.520020	0.313610	0.042892	0.059029	0.013799	0.004951	0.004951	0.004951
53	0.041599	0.060480	0.009684	0.445320	0.717590	0.614110	0.336080	0.042414	0.057592	0.013784	0.004852	0.004852	0.004852
54	0.039934	0.053286	0.011246	0.746120	0.619530	0.618070	0.278580	0.041477	0.057273	0.013932	0.004753	0.004753	0.004753
55	0.038701	0.059844	0.009194	0.753860	0.617340	0.613300	0.329590	0.042456	0.057471	0.013378	0.004654	0.004654	0.004654
56	0.040720	0.055352	0.008309	0.720080	0.617360	0.581210	0.357560	0.042285	0.058059	0.013137	0.004555	0.004555	0.004555
57	0.037653	0.056697	0.010571	0.701440	0.615740	0.513410	0.298200	0.043647	0.057679	0.013351	0.004456	0.004456	0.004456
58	0.040300	0.055815	0.009846	0.494160	0.732890	0.538290	0.289380	0.040499	0.057031	0.013650	0.004357	0.004357	0.004357
59	0.035517	0.059362	0.009710	0.784260	0.606980	0.609430	0.310620	0.041213	0.057490	0.013964	0.004258	0.004258	0.004258
60	0.038341	0.055604	0.008602	0.788710	0.609880	0.610500	0.382720	0.041558	0.059438	0.013706	0.004159	0.004159	0.004159
61	0.036879	0.056359	0.009183	0.786310	0.615640	0.603880	0.359060	0.041357	0.058659	0.013440	0.004060	0.004060	0.004060
62	0.038012	0.054734	0.008605	0.781080	0.620370	0.607260	0.378380	0.041008	0.057990	0.013263	0.003961	0.003961	0.003961
63	0.036194	0.053625	0.010243	0.545710	0.737560	0.606610	0.342690	0.040847	0.058812	0.012927	0.003862	0.003862	0.003862
64	0.037116	0.055648	0.007813	0.553710	0.764760	0.615900	0.367600	0.040017	0.058246	0.012718	0.003763	0.003763	0.003763
65	0.036083	0.055604	0.009748	0.572490	0.773150	0.600850	0.371120	0.041108	0.060134	0.012674	0.003664	0.003664	0.003664
66	0.037664	0.055072	0.008417	0.560970	0.768520	0.601390	0.340130	0.040080	0.059166	0.012564	0.003565	0.003565	0.003565
67	0.036199	0.056454	0.007079	0.558330	0.768520	0.522400	0.273410	0.041378	0.059861	0.012591	0.003466	0.003466	0.003466
68	0.036897	0.059348	0.007089	0.545860	0.773150	0.510450	0.299980	0.039496	0.059434	0.012412	0.003367	0.003367	0.003367
69	0.035139	0.052027	0.009998	0.511630	0.740740	0.526280	0.297740	0.040770	0.059936	0.012458	0.003268	0.003268	0.003268
70	0.037538	0.054986	0.007834	0.524840	0.740740	0.534880	0.294350	0.039841	0.059807	0.012884	0.003169	0.003169	0.003169

Continued on next page

epoch	train/box_loss	train/obj_loss	train/cls_loss	metrics/precision	metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.95	val/box_loss	val/obj_loss	val/cls_loss	x/lr0	x/lr1	x/lr2
71	0.035689	0.056022	0.008408	0.567600	0.731210	0.610080	0.317420	0.040371	0.059808	0.013236	0.003070	0.003070	0.003070
72	0.036143	0.050855	0.007669	0.566640	0.712070	0.602190	0.330360	0.040171	0.060719	0.013304	0.002971	0.002971	0.002971
73	0.034006	0.049396	0.008912	0.567700	0.716140	0.613490	0.351790	0.040910	0.061984	0.013421	0.002872	0.002872	0.002872
74	0.034535	0.052025	0.007876	0.800220	0.616640	0.606060	0.358810	0.038479	0.061931	0.013539	0.002773	0.002773	0.002773
75	0.032744	0.055572	0.008096	0.573060	0.758720	0.594830	0.353030	0.039352	0.061148	0.013271	0.002674	0.002674	0.002674
76	0.034182	0.053466	0.008248	0.571150	0.768140	0.592590	0.335380	0.039270	0.061360	0.013029	0.002575	0.002575	0.002575
77	0.034890	0.058161	0.006533	0.540470	0.745370	0.592930	0.328570	0.039753	0.061337	0.012632	0.002476	0.002476	0.002476
78	0.033498	0.053385	0.007207	0.528790	0.745370	0.602380	0.362150	0.038719	0.061122	0.012432	0.002377	0.002377	0.002377
79	0.032253	0.049484	0.006605	0.524140	0.747470	0.611080	0.345390	0.040464	0.061638	0.012169	0.002278	0.002278	0.002278
80	0.033244	0.052998	0.006644	0.526640	0.746500	0.614650	0.374960	0.038061	0.060793	0.011993	0.002179	0.002179	0.002179
81	0.032232	0.050525	0.007595	0.568790	0.773150	0.620610	0.353670	0.039166	0.060411	0.011800	0.002080	0.002080	0.002080
82	0.033864	0.052885	0.007464	0.572680	0.773150	0.621400	0.381730	0.038064	0.060022	0.011661	0.001981	0.001981	0.001981
83	0.033146	0.052943	0.006895	0.581110	0.776430	0.625800	0.364200	0.038076	0.059701	0.011560	0.001882	0.001882	0.001882
84	0.031654	0.053424	0.006729	0.561690	0.773150	0.617010	0.363050	0.038255	0.059843	0.011456	0.001783	0.001783	0.001783
85	0.032962	0.051674	0.007930	0.561160	0.773150	0.617760	0.365080	0.038312	0.060895	0.011245	0.001684	0.001684	0.001684
86	0.032353	0.050767	0.007452	0.553580	0.773150	0.617990	0.378070	0.038088	0.060913	0.011193	0.001585	0.001585	0.001585
87	0.030879	0.051153	0.006384	0.554360	0.773150	0.615030	0.381690	0.037963	0.061076	0.011221	0.001486	0.001486	0.001486
88	0.031391	0.055855	0.008487	0.551280	0.773150	0.611590	0.357720	0.038401	0.060858	0.011163	0.001387	0.001387	0.001387
89	0.031178	0.052482	0.007177	0.554080	0.773150	0.617590	0.369930	0.037652	0.059876	0.011178	0.001288	0.001288	0.001288
90	0.031203	0.048922	0.007928	0.555870	0.773150	0.622220	0.387310	0.038517	0.060087	0.011170	0.001189	0.001189	0.001189
91	0.030829	0.054351	0.005930	0.562440	0.773150	0.630880	0.369340	0.037949	0.060240	0.011181	0.001090	0.001090	0.001090
92	0.030838	0.053359	0.006825	0.561600	0.773150	0.631230	0.374170	0.038073	0.060400	0.011184	0.000991	0.000991	0.000991
93	0.029671	0.048725	0.005502	0.564930	0.773150	0.622670	0.390090	0.038109	0.060618	0.011178	0.000892	0.000892	0.000892
94	0.031193	0.049433	0.007465	0.564810	0.773150	0.615120	0.378440	0.038077	0.060814	0.011020	0.000793	0.000793	0.000793
95	0.031223	0.053378	0.006367	0.567130	0.773150	0.614630	0.378370	0.038068	0.060811	0.011134	0.000694	0.000694	0.000694
96	0.029229	0.050365	0.007001	0.568020	0.773150	0.624840	0.384410	0.038259	0.060876	0.011244	0.000595	0.000595	0.000595
97	0.029822	0.050161	0.005828	0.570240	0.773150	0.624990	0.389590	0.037760	0.060861	0.011321	0.000496	0.000496	0.000496
98	0.029657	0.052525	0.005995	0.571410	0.773150	0.624960	0.386310	0.037740	0.060848	0.011403	0.000397	0.000397	0.000397
99	0.030148	0.053251	0.006765	0.571810	0.773150	0.626680	0.384920	0.037749	0.060797	0.011415	0.000298	0.000298	0.000298