Estimation of root crop yield: Integration of computer vision for quantity counting, quality assessment, and geospatial data mapping in Chinese Yam harvesting

（根菜類の収量推定：長芋の収穫における数量カウント、品質評価、および地理空間データマッピングのためのコンピュータビジョンの統合）

Hokkaido University　Graduate School of Agriculture

Environmental Resources　Doctor Course

Cao Tianzhi

ACKNOWLEDGEMENTS

Throughout the long journey of my doctoral studies, I have been fortunate to receive help, support, and guidance from many individuals. I would like to express my deepest gratitude to all of them.

First and foremost, I would like to extend my special thanks to my supervisor, Professor Hiroshi OKAMOTO. Professor has not only guided me into the depths of academic research with his exceptional academic prowess, but his rigorous approach to research and his patient and meticulous guidance have helped me overcome numerous challenges in my research. Professor OKAMOTO's wisdom and dedication will continue to inspire me to move forward.

I would also like to thank Professors Noboru NOGUCHI and Kazunobu ISHI for their valuable advice and support for my research work. Their expertise and unique insights have been invaluable to my study.

Additionally, I am deeply grateful to our laboratory secretary for her assistance. Her help in managing day-to-day administration and logistics allowed me to focus more on my research work.

Lastly, I wish to express my gratitude to my family, especially my parents and loved ones. It is their selfless love and support on my journey to pursue my academic dreams that have given me the strength and courage to persevere to the end.

Thank you to everyone who has helped and supported me. Your encouragement and assistance have made this academic journey all the more enriching and meaningful.

ABSTRACT

Yield estimation of root crops has emerged as one of the fastest-growing fields in precision

agriculture in recent years. Chinese Yam (Dioscorea polystachya), widely cultivated in Japan and also called 'Nagaimo', represents a typical root crop within the Dioscoreaceae genus. This research introduced a framework aimed at providing farmers with yield estimates derived from videos of yams. It utilized geospatial information collected during harvesting to guide yield estimation across farmland. The framework included accurate yield counting utilizing a deep learning model for yam detection and a multi-tracking model for tracking the detected yams. Modifications were implemented in the original algorithm to boost efficiency without requiring retraining. Moreover, leveraging Shuffle-Net and transfer learning, a lightweight deep learning model was developed to detect defects of a yam and grade a yam by its size. Simultaneously, correlation analysis between GNSS location data and yam quantity distribution was conducted, mapping geospatial data. Leveraging image processing through cameras and computers enabled cost-effective collection of resultant data without impacting root crops, given its non-contact nature. This assists farm managers in improving production management and planning.

Counting yams by object detection and multi-object tracking:

Accurate counting is crucial for determining root crop yield. To address this, a method for precisely counting harvested Chinese Yams in the field was devised. To optimize the detector's recognition capabilities, the attention module CBAM was integrated with the neck part of the YOLOv5s network, thereby enhancing the network's feature extraction ability CIoU Loss was used instead of GIoU Loss as the target bounding box regression loss function to speed up the bounding box regression rate while improving positioning accuracy. The structure of the Deep-SORT appearance feature extraction network was adjusted and retrained on the yam re-identification dataset to reduce the identity switching caused by target overlap. The enhanced

YOLOv5s detector was integrated with Deep-SORT, establishing a virtual detection line within the video to tally the quantity of yams. Experimental results indicated a notable improvement in the accuracy of yam counting.

Defect detection and size grading of yam in harvest:

The goal was to be able to detect defective yams and to grade a yam by its size through video dataset. Specifically, based on Shuffle-Net and transfer learning, a lightweight deep learning model (CDD Net) was constructed to detect surface and shape defects of yam. Methods based on minimum bounding rectangle (MBR) fitting and convex polygon approximation were also proposed. The experimental results demonstrated that the proposed CDD Net achieved detection accuracy of 98.94% for two categories (normal and defective) and 92.92% for multi-category classification (normal, curve, fork root, break). Additionally, the dimensional accuracy rates for Minimum Bounding Rectangle (MBR) fitting and convex polygon approximation were recorded at 92.8% and 95.1%, respectively. This experimentation presented a practical approach for defect detection and size grading.

Geospatial mapping of yam count and its visualization:

The integration of geospatial information into crop analysis is pivotal for intelligent harvesting, offering farmers comprehensive insights to optimize their resources. In this context, GNSS points were recorded while capturing videos of the yam rows, supporting these functionalities. The synchronization of GNSS capturing with video recording allowed for the correlation of GNSS coordinates with video frames during the counting process. The recorded counts, along with their respective GNSS coordinates, were utilized to visualize the data. Subsequently, a map was generated, encompassing yield information on a point-by-point basis.

In conclusion, YOLOv5s deep learning model was served as the detector, coupled with the Deep-SORT target tracking method, to accurately count Chinese Yams. Additionally, a size grading method based on (MBR) fitting and convex polygon approximation was proposed. Overcoming the limitations of manual labor, approach utilizes computer vision and deep learning technologies for precise detection and grading. This innovation provides robust support for estimating harvest yield not only for Chinese Yams but also for other root crops in future agricultural practices.

Table of Contents

List of figures

list of tables

Chapter 1 Introduction

1.1 Research Background

Hokkaido's agriculture, a vital sector in Japan, is currently grappling with significant challenges due to a labor shortage. This issue is compounded by an aging agricultural population, with an increasing proportion of farmers aged 65 and over. As the number of people working on farms decreases, there arises a pressing need for more stable and efficient cultivation techniques. Despite existing stable cultivation practices, many farmers in Hokkaido observe variations in yield and quality but struggle to utilize this information effectively for future cultivation. This is largely due to a lack of understanding of the factors contributing to these variations.

One of the key challenges in addressing this issue is the need for accurate data collection and analysis to identify the causes of yield variation. While technologies for recording management work history and using remote sensing to understand the natural environment have been developed, their application in harvesting root and tuber crops remains limited, especially in accurately capturing yields on a small spatial scale.

In the broader context of global agriculture, root crops like the Chinese Yam (*Dioscorea polystachya*), known as 'Nagaimo' in Japan, are a crucial component of the food supply due to their economic and nutritional value. However, yield estimation for these crops has traditionally been a manual, labor-intensive process with a high propensity for inaccuracy. This has necessitated the exploration of more efficient, accurate, and technologically advanced methods.

The advent of precision agriculture has brought about a transformation in managing and estimating crop yields, particularly for root crops. Recent advancements in computer vision, deep learning, and geospatial technologies have opened new avenues for precision agriculture. These technologies provide non-invasive, cost-effective, and accurate means for assessing crop yields, essential for effective farm management and planning. Image processing technology, using cameras and computers, presents a promising non-contact method for data collection. This approach not only aids in identifying problems during the growth process but also enables the analysis of factors affecting different growth stages. By providing practical information to farmers, this technology can significantly improve cultivation methods and increase yields.

In summary, the labor shortage and technical challenges faced in Hokkaido's agriculture, along with the broader challenges in global root crop cultivation, need innovative solutions and the application of advanced technology. This is imperative to achieve more efficient, sustainable, and profitable agricultural production.

1.2 Research Review

The evolution of precision agriculture has been significantly influenced by the integration of advanced technologies, particularly in the domain of yield estimation and quality assessment of crops. This progress is emblematic of a broader trend in agricultural technology, where traditional methods are being supplemented or replaced by more efficient, accurate, and technologically sophisticated approaches.

1.2.1 Traditional and Modern Methods

Traditionally, the identification of root and tuber crops has relied on manual classification, which is subject to many limitations, such as susceptibility to subjective factors, leading to significant discrepancies in results. Unlike other traditional manual identification techniques, computer vision technology has emerged as a crucial tool for quality assessment, classification, automatic grading, and yield estimation in fruit and vegetable recognition, achieving good results in practical applications. The classification of fruits and vegetables through machine recognition is a relatively complex process due to the wide variety of species, irregular shapes, colors, and textures, which places higher demands on the classification systems. The evolution of visual data for fruits and vegetables from binary images to hyperspectral images has facilitated the development of fruit and vegetable recognition, and the technology for recognition and classification typically combines feature descriptions from visual data with machine learning algorithms.

In 2019, Li and others used drones to capture ortho-images of potato plants during the emergence stage, employing the excess green index (ExG) and Otsu's method to extract potato plant objects from bare soil and calculate six shape features as variables for a Random Forest classifier (RF) to estimate the number of potato plants at the emergence stage. Koh and others, when most safflower seedlings were at the 2-4 leaf stage, used drones equipped with RGB cameras to capture remote sensing images from a height of 20 meters above the ground, generating ortho-mosaic images. They then set a threshold to distinguish potential safflower seedlings from the soil background using chessboard segmentation and calculated the correlation coefficient between the segmented objects and safflower seedling group templates to identify safflower seedlings among all possible objects.

1.2.2 Deep Learning in Agriculture

In recent years, deep learning methods have shown great potential in fields like computer vision and natural language processing. Various deep learning-based models have demonstrated superior performance in specific visual tasks like object detection, outperforming humans in some cases. In agriculture, these models have found applications in pest monitoring, species identification, crop detection, and counting.

In 2018, Dijkstra and others introduced a fully convolutional network (FCN) named CentroidNet, specifically for crop localization and counting. Researchers chose the U-Net semantic segmentation model as the base network, with the output feature map having the same size as the input image and a fixed number of channels. The first two channels contained a vector for each pixel, pointing towards the nearest object centroid, acting as a vote for the centroid's position, and the remaining channels contained the logistic map for each category, indicating the probability of the pixel belonging to that category. The centroids were determined through centroid voting.

In 2019, Liu and others proposed a cheap, portable, and fast fruit counting method. They first used the FasterR-CNN object detection model to detect fruits and tree trunks in continuous video frames, then integrated the object detection results with optical flow estimates through Kalman filtering. They tracked fruits and tree trunks in consecutive video frames and used a structure-from-motion algorithm for 3D modeling to identify repeat counting scenarios and estimate fruit numbers. Häni and others introduced a modular end-to-end system for apple orchard yield estimation. Researchers used the U-Net model for segmenting apple clusters in

images, then treated counting different apple clusters as a classification task, using the ResNet50 neural network model. They then projected apple clusters into a reconstructed global 3D space using a structure-from-motion algorithm and tracked them to count apples. Jiang and others developed a deep learning-based method for estimating the number of field plant seedlings. Researchers trained a FasterR-CNN model for detecting yam seedlings and used Kalman filtering and the Hungarian algorithm to match and track detected seedling objects in continuous video frames, ultimately estimating the number of yams based on the tracked count.

In 2020, Zhang and others introduced a deep learning-based method for counting maize plants. Researchers used the FasterR-CNN model to detect maize plants in images and then proposed a visual tracking method combining optical flow, Kalman filtering, and Kernelized Correlation Filter (KCF) to track target objects in continuous image frames and count maize plants. Machefer and others proposed a counting method combining remote sensing and deep learning. Researchers first captured remote sensing images of low-density crops like potatoes and lettuce, then used a MaskR-CNN model trained via transfer learning to detect and segment plant objects in the images. The model was fine-tuned for this new task. In 2021, Wang and others proposed a method for counting maize objects in continuous video frames. Researchers first collected smooth and stable videos using a camera-mounted cart, then extracted frames to create an image dataset. Based on this dataset, they trained a YoloV4 object detection model to detect maize seedlings in image frames. They predicted maize seedling trajectories using Kalman filtering and associated the same targets in adjacent frames using the Hungarian algorithm to count maize seedlings.

Mukhtar and colleagues proposed a wheat counting method based on a semantic

segmentation network and a regression network. The researchers used a semi-supervised method based on cross-consistency to train the semantic segmentation network using unlabeled images. The trained semantic segmentation network extracted tiny clusters of plants from RGB images. These segmented plant clusters were then fed into a regression network to extract multi-scale features. The regression network performed regression on the number of wheat in each plant cluster, and finally, the total number of wheat in all plant clusters in the image was summed up.

In conclusion, the field of precision agriculture is experiencing a significant shift, driven by the integration of deep learning models and computer vision techniques. These advancements not only address the challenges posed by traditional methods but also open up new possibilities for enhancing agricultural productivity and sustainability.

1.3 Research Objectives

1.3.1 Advancing Yield Estimation Techniques for Root Crops

The primary objective of this doctoral thesis is to develop an advanced framework for yield estimation of root crops, specifically focusing on Chinese Yams (*Dioscorea polystachya*). This research seeks to enhance the accuracy and efficiency of crop counting under field conditions. The approach involves optimizing a deep learning model, notably YOLOv5s, integrating it with an attention module (CBAM) for improved feature extraction, and employing Deep-SORT for effective multi-object tracking. This methodology aims to address the limitations of current manual classification methods by leveraging the precision and scalability of computer vision and

deep learning technologies.

1.3.2 Implementing Defect Detection and Size Grading Systems

A key goal is to establish a reliable defect detection and size grading system for Chinese Yams. The development of a lightweight deep learning model, referred to as CDD Net, will be central to this effort. This model will be combined with innovative computer vision techniques, such as minimum bounding rectangle (MBR) fitting and convex polygon approximation, to identify various defect categories and accurately grade yams based on size. This system is intended to provide more detailed and accurate quality assessments compared to traditional manual methods.

1.3.3 Utilizing Geospatial Data for Enhanced Yield Analysis

The integration of geospatial data into the yield estimation process forms a crucial aspect of this research. Utilizing Global Navigation Satellite System (GNSS) data and other geospatial information will allow for the precise mapping and visualization of yam distribution across farmlands. This integration is expected to provide deeper insights into the spatial variability of crop yields, assisting in better understanding and managing the factors contributing to yield variations.

In conclusion, this research is poised to make a substantial contribution to the field of precision agriculture. By providing a robust, scalable, and accurate method for yield estimation and quality assessment of Chinese Yams, it sets the groundwork for similar applications in other root crops. The successful implementation of this framework is expected to revolutionize yield

estimation practices, leading to enhanced productivity, optimized resource use, and promotion of sustainable farming practices. The objectives outlined in this thesis aim to address the critical challenges faced by Hokkaido's agriculture sector and the broader global agricultural community. By integrating advanced technologies such as deep learning, computer vision, and geospatial data analysis, this research strives to provide innovative solutions to improve agricultural efficiency, sustainability, and profitability.

Chapter 2 Counting yams by object detection and multi-object tracking

2.1 Methodology

2.1.1Brief introduction to convolutional neural network

Convolutional Neural Networks (CNNs), as a special type of artificial neural network, were actually proposed and studied by scholars before 2006, especially achieving good recognition results in the field of handwritten digit recognition. It was not until after AlexNet in 2012 that CNNs gained widespread attention, and simultaneously, algorithms based on CNNs also developed rapidly. Compared to the complex task of feature extraction performed by designers in the past, CNNs have enabled the automatic learning of data features, rather than manual operations. Additionally, what sets CNNs apart from other networks is their use of convolutional operations, making them an important technological tool in the field of image processing.

(1)Input layer of convolutional neural networks

As the entry point for convolutional neural networks, the input layer primarily records the

pixel information of images. The current mainstream format is RGB pixels, which undergo mean processing and variance processing for quantification of the image to reduce the total amount of information in the entire image. This simplification facilitates forward network propagation and backward derivative calculations. However, when inputting image features, the volume of information directly affects the accuracy of the prediction target. Therefore, extensive research and processing are usually conducted when inputting image features, such as pre-processing images through rotation, cropping, flipping, and Gaussian filtering. There are also specific restrictions on the image input format. For object detection and classification, grayscale images are commonly used, while for live detection and high-precision targets, Moiré patterns and image texture features are processed and directly inputted as RGB pixels. The input layer is a relatively important component in the entire convolutional neural network.

(2)Convolution layer of convolutional neural networks

The convolution layer primarily performs convolution operations on the input images. Its working mode can be viewed as conducting local operations while sliding through the feature layer, thereby generating multiple feature maps, with each feature map corresponding to the extraction of an image feature. Local operations involve multiplying the weight values of a given size of the convolution kernel with the corresponding input pixel values, followed by accumulative computation to obtain the local information of the image. Based on this, utilizing the weight-sharing characteristic of the convolution layer, a convolution kernel is used as a sliding window to scan the entire image, resulting in the final convolutional feature map. The convolution operations mentioned not only effectively extract the valid features of images but

also significantly reduce the number of parameters, thereby decreasing computational load and reducing the complexity of the neural network. Convolution layers are generally preset with scales of 3x3 or 5x5. The entire receptive field can quickly extract the feature information of the matrix under a limited sliding step length, effectively completing the task of convolution. Next, will introduce in more detail the working principle of the convolution layer and discuss its role in convolutional neural networks.

Taking a two-dimensional scenario as an example, the calculation formula, as shown in Equation (2.1), provides a more intuitive description of the convolution operation process.

$$y_{(i,j)} = \sum_{u=0}^{M-1} \sum_{V=0}^{N-1} W(u,v) X(i-u, j-v) + w_b \tag{2.1}$$

M represents the width of the convolution kernel, N its depth, W(u,v) is the weight of the u-th row and m-th column of the convolution kernel, X(i-u,j-v) represents the coordinate value of the input image data, and w_b is used to represent the bias.

The convolution operation process is as illustrated in Figure 2.1. First, set the convolution stride to 1, and use the predefined 3x3 convolution kernel to slide from left to right, and from top to bottom, performing the aforementioned convolution operations. This results in the corresponding position output, which ultimately determines the feature map output of the convolution layer.

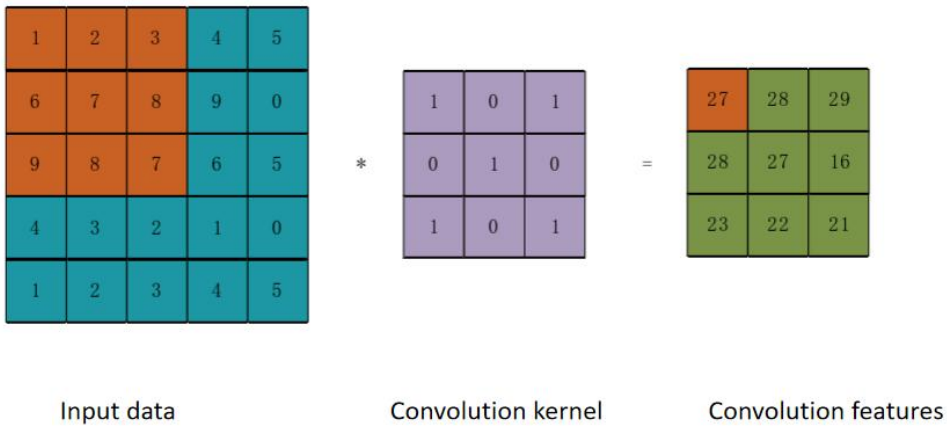


*Fig.*2.1 *Schematic diagram of convolution operation*

(3)Pooling layer of convolutional neural networks

The use of multiple convolution kernels results in the stacking of computations in the convolution layer, leading to high dimensions and excessive computational requirements for subsequent input layers, which can cause overfitting. To reduce computation, convolutional neural networks incorporate a pooling layer after the convolution layer to perform dimensionality reduction on the output of the previous layer. The primary function is to downsample the feature map, reducing the dimensions of the network layer ' s feature map to avoid overfitting. The pooling layer also uses a predefined sliding window size to scan the feature map output of the convolution layer and outputs a compressed new feature map for use in the next layer of the network structure. However, unlike the convolution layer, the pooling layer only requires selecting parameters such as the type of pooling, kernel size, and stride, without considering internal parameter values.

Common pooling methods include max pooling and average pooling. Max pooling involves selecting the maximum value from a local area of the image to replace all data information in

that area. It is fast and is the most widely used image processing method in practical applications; initially, the filtering algorithm in OpenCV was implemented using this concept. In contrast, average pooling uses the average value of a subregion to replace the information therein, thereby retaining overall data characteristics and highlighting background information.

Figure 2.2 shows two types of pooling operations commonly used with a kernel size of 2x2 and a stride of 2. First, the input feature map is divided into four equal-sized subregions based on the kernel size. The two aforementioned pooling methods are then applied, resulting in two different post-pooling feature maps.

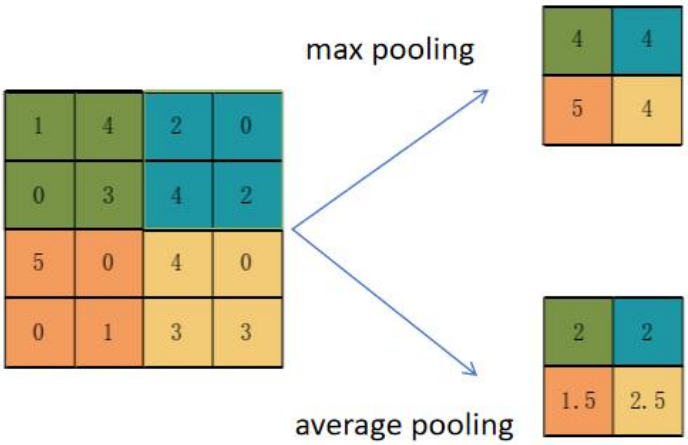


*Fig.*2.2 *Schematic diagram of convolution operation*

In summary, pooling operations use a single value to replace all data in a local area, effectively replacing all feature information within that area. This is equivalent to reducing the width and height dimensions of the feature map, thereby lowering the input dimensions and the number of parameters in subsequent network layers, reducing the model size, and enhancing the training efficiency of the convolutional neural network.

(4) Fully connected layer of convolutional neural network

After image convolution, multiple feature maps are produced, each corresponding to the extraction of an image feature. Subsequently, through pooling layer operations, dimensionality reduction is performed on the output from the previous layer. The reduced volume of data, after being processed by the Sigmoid function, abstracts discrete feature information, which is then connected to a fully connected layer.

To improve the classification results after the action of the above network layers, the fully connected layer connects the output information of the neurons correspondingly and flattens the output feature matrix into a one-dimensional vector, which serves as the input to the classifier. If the aforementioned convolution and pooling layers repeatedly process the input image's features, then the fully connected layer represents the refined result of this data. It is the fusion of features in a highly integrated form as input to enhance the quality of the output classification. The computation formula for the feature vector in the fully connected layer is shown in Equation (2.2).

$$y = f(wx + b) \tag{2.2}$$

Where f is the activation function, x and y represent the input and output feature vectors, respectively, and w is the weight matrix.

(5)Activation function and loss function

Activation Functions:

In convolutional neural networks, when processing initial image data through convolution

and pooling operations, the weights of image pixels are typically subjected to linear operations. However, sometimes due to insufficient expressive power of the information, activation functions are introduced to add non-linear factors for related operations. This section focuses on three commonly used activation functions.

(a) Sigmoid Activation Function

The Sigmoid activation function is a monotonically increasing function, and its calculation formula is shown in Equation (2.3).

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{2.3}$$

As Figure 2.3 shows, the Sigmoid activation function is a monotonically increasing function with an exponential shape, similar to biological neurons. It is also evident that the output values are compressed to arbitrary numbers between (0, 1), which allows for categorization of certain values for classification purposes. However, when the input data becomes very large or very small, the output data is more likely to be compressed to 1 or 0, leading to a 'saturation effect' that prevents timely updates of the network's parameters, thereby affecting the network's performance.

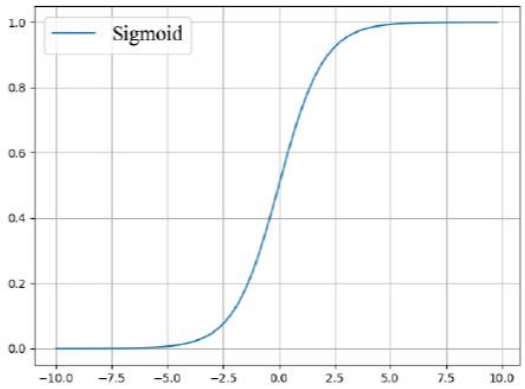


Fig.2.3 Sigmoid function graph

(b)Tanh function

The Tanh function is also a common activation function, and the Equation is shown in (2.4).

$$Tanh = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.4}$$

As can be seen from Figure 2.4, the Tanh function is a standard hyperbolic tangent curve, and it is also monotonically increasing. It was proposed based on the Sigmoid function, but the difference is that the output values after being processed by the Tanh function range between (-1, 1). This helps to amplify certain feature effects during the training process. However, it still exhibits a 'soft saturation phenomenon', which can lead to the vanishing gradient problem.

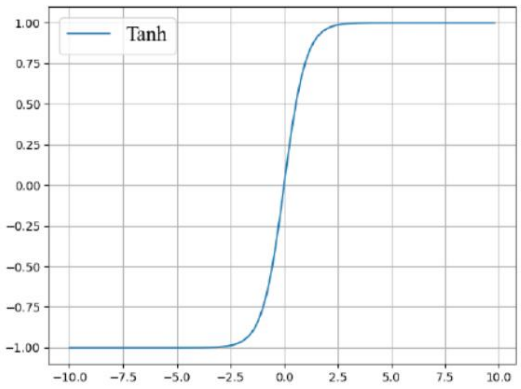


Fig.2.4 Tanh function graph

(c) ReLU Activation Function

The ReLU (Rectified Linear Unit) function can be regarded as a piecewise function, and its calculation formula is shown in Equation (2.5).

$$\mathrm{Re}\,LU(x) = \begin{Bmatrix} x, x \geq 0 \\ 0, x < 0 \end{Bmatrix} \tag{2.5}$$

As illustrated in Figure 2.5, the output of the ReLU function is either 0 or a positive number. The gradient of the function is 1 when x > 0 and 0 when x < 0. Particularly, in the positive range where x > 0, its gradient is constantly 1, effectively overcoming the gradient saturation issue. In

terms of computational load, both the Sigmoid and Tanh functions require exponential calculations, whereas the ReLU function is simpler in comparison. Additionally, experiments have also found that the ReLU function is conducive to faster network training convergence, with the model convergence speed increasing by nearly six times.

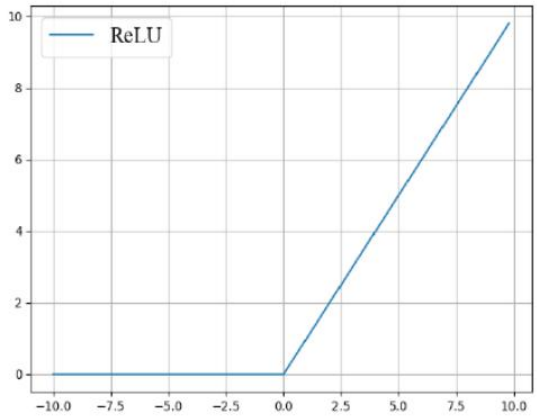


Fig.2.5 ReLU function graph

Loss Functions:

Loss functions are typically used to measure the deviation between a model's predictions and the actual targets. During the training of a neural network, the aim is to minimize the network's loss through parameter optimization, comparing the true values of the targets with the predicted values of the model. The loss is then calculated using a loss function. Generally, the smaller the loss, the higher the robustness of the model.

(a) Regression Loss Function

L1 loss function, also known as Mean Absolute Error, uses the average error magnitude as the measure between the true value y and the predicted f(x) value. Its calculation formula is shown in Equation (2.6).

$$l_{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y - f(x)\right| \qquad (2.6)$$

L2 loss function, or Mean Squared Error function, uses the sum of the squares of the differences as a measure. Its calculation formula is shown in Equation (2.7).

$$l_{MAE} = \frac{1}{n}\sum_{i=1}^{n}(y - f(x))^2 \qquad (2.7)$$

(b) Classification Loss Functions

0-1 Loss Function: It compares whether the output value is equal to the input value, i.e., whether the actual target value is equal to the prediction result. It is 0 if they are equal, otherwise 1. The calculation formula is shown in Equation (2.8).

$$l(y, f(x)) = \begin{cases} 0, y = f(x) \\ 1, y \neq f(x) \end{cases} \qquad (2.8)$$

Cross-Entropy Loss Function: Generally based on Softmax classification, it converts the required output values into class probabilities through an exponential form. The calculation formula is shown in Equation (2.9).

$$p = \frac{e^{zj}}{\sum_{j=1}^{C} e^{zj}} \qquad (2.9)$$

2.1.2 YOLOv5s target detection

(1)YOLOv5

Compared to previous object detection algorithms that used a sliding window approach and the time wastage issue caused by generating candidate regions, the introduction of the YOLO (You Only Look Once) algorithm undoubtedly brought a new research direction to the field of object detection. The YOLO algorithm eliminates the step of selecting candidate boxes and

adopts a method of predicting bounding boxes, using the entire image as the input for network prediction. In the output layer, it directly regresses the location and classification problem, balancing real-time performance and accuracy.

The YOLOv5 algorithm uses a grid division method, dividing the input image into several different detection regions. It uses the position of the target center to select the prediction bounding box, predicting the target when the center point is located in a specific grid. The divided cells are responsible for predicting the bounding box information and confidence, where the confidence includes the confidence score of the predicted box's target and the accuracy of the predicted box.In each data training load of the YOLOv5 algorithm, such as passing an RGB image, the data loader scales the image, adjusts the color space, and applies Gaussian preprocessing during training through the convolutional layers. These three methods achieve data augmentation, thereby implementing secondary data feature processing.Using the CSPDarknet53 backbone network, the algorithm extracts a large amount of deep features in the image through continuous downsampling operations. It integrates all gradient changes into the feature map, thereby reducing the number of parameters and FLOPs to increase the model's training speed.

At the input end of the YOLOv5 model, the Mosaic method is mainly used for data augmentation. During network training, due to dataset variations, the algorithm typically initializes the size of the anchor boxes. Unlike the method of obtaining anchor box sizes using K-means, the YOLOv5 algorithm embeds the calculation of anchor boxes into the overall code, thus allowing for adaptive anchor box training and eliminating the need for 'extra' operations before training. For different datasets, the adaptive computation generates prediction boxes

based on the initial fixed size, then calculates the Loss by comparing them with the actual boxes, continuously updating and calculating to determine the optimal size of the anchor boxes.

(2)YOLOv5s

In the DBT algorithm, the effectiveness of the detector significantly impacts the result of target tracking. Moreover, the speed of the detector and the model size are crucial for achieving real-time target tracking. Since most farmland sites utilize embedded devices with limited computing power, deploying large-scale detection models is impractical. To minimize computing costs and enhance practicality, this research selects YOLOv5s, the smallest model in the YOLOv5 series, as the foundational model for Chinese Yam detection. The structure of YOLOv5s is divided into four main parts: Input, Backbone, Neck, and Head, as shown in Figure 2.6

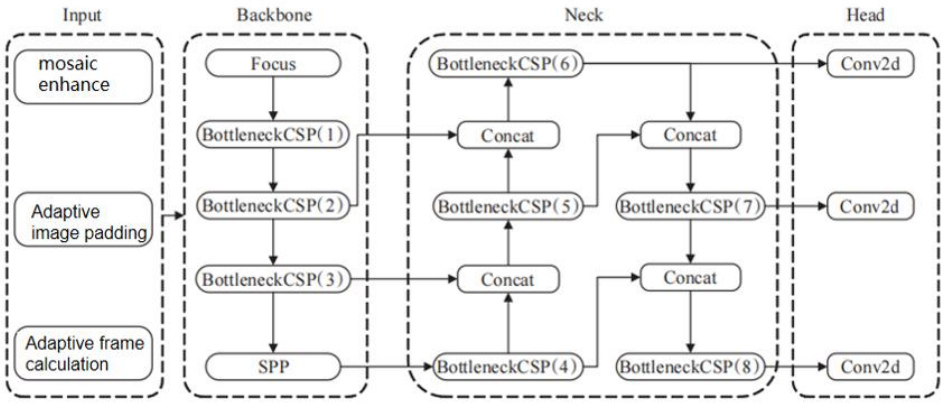


Fig.2.6. Structure of YOLOv5s

The Input primarily involves data preprocessing, which includes Mosaic data augmentation, adaptive image padding, and integrated adaptive anchor box calculations. This integration allows for automatic adjustment of the initial anchor box size when switching datasets. The Backbone network, through deep convolution operations, extracts various levels of features from the image. It primarily utilizes BottleneckCSP (bottleneck cross-stage partial) and SPP (spatial pyramid

pooling). The former aims to reduce computational load and improve inference speed, while the latter facilitates feature extraction at different scales on the same feature map, thereby enhancing detection accuracy.

The Neck layer comprises a Feature Pyramid Network (FPN) and a Path Aggregation Network (PAN). The FPN conveys semantic information from the top down in the network, whereas the PAN transmits positional information from the bottom up. This layer integrates information from different layers in the Backbone to further improve detection capabilities. Finally, as the concluding detection component, the Head outputs predictions for targets of varying sizes on feature maps of different scales.

(3) YOLOv5s Fusion Attention Mechanism

In the field of computer vision, the effectiveness of the attention mechanism has been well-established, finding widespread application in tasks like classification, detection, and segmentation. Within CNN networks, the attention mechanism focuses on feature maps to extract valuable attention information, which primarily includes spatial attention and channel attention. The Convolutional Block Attention Module (CBAM) concurrently addresses both spatial and channel information. It reconstructs the feature map in the network's intermediate layers through two sub-modules: the Channel Attention Module (CAM) and the Spatial Attention Module (SAM). These modules accentuate important features while suppressing general ones, thereby enhancing the effectiveness of target detection. The structure of this mechanism is depicted in Figure 2.7.

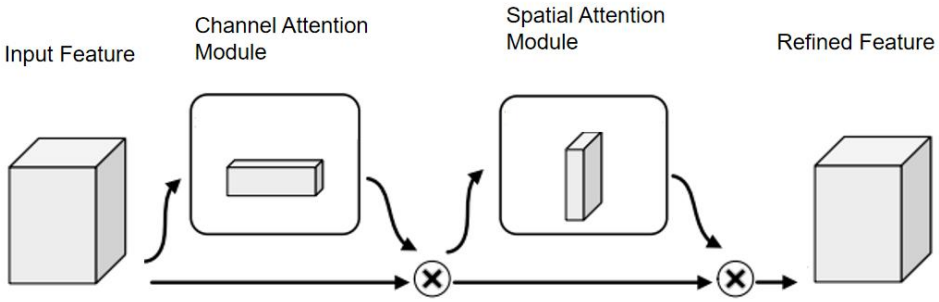


Fig.2.7. Structure of CBAM

For the 3D feature map F ∈ RC × H × W of a certain layer in the CNN network, CBAM sequentially infers the 1D channel attention feature map Mc and the 2D spatial attention feature map Ms from F, and performs element-by-element correlation respectively. Multiply, and finally get the output feature map of the same dimension as F, as shown in Equation (2.10). Where F represents the feature map of a network layer in the network, Mc(F) represents the channel attention reconstruction of F by CAM, Ms(F') represents the spatial attention reconstruction of F' by SAM on the result of channel attention reconstruction, ⊗ means element-wise multiplication.

$$\begin{cases} F' = Mc\,(F) \otimes F \\ F'' = Ms(F') \otimes F' \end{cases} \tag{2.10}$$

The structure of the CAM (Channel Attention Module) and SAM (Spatial Attention Module) is depicted in Figure 2.8 illustrates the computational process of CAM. In CAM, each channel of the input feature map F simultaneously undergoes maximum and average pooling. The resulting intermediate vector is then processed through a multi-layer perceptron (MLP). To reduce computational complexity, the MLP is designed with only one hidden layer. The feature vector output by the MLP is added element-wise and subjected to a Sigmoid activation operation to

yield the channel attention Mc. Figure 2.8 below describes the computational process of SAM. The feature map F', activated by Mc, undergoes maximum and average pooling along the channel direction. The intermediate vector obtained is then convolved, and the convolution result is activated by a Sigmoid function to derive the spatial attention Ms.

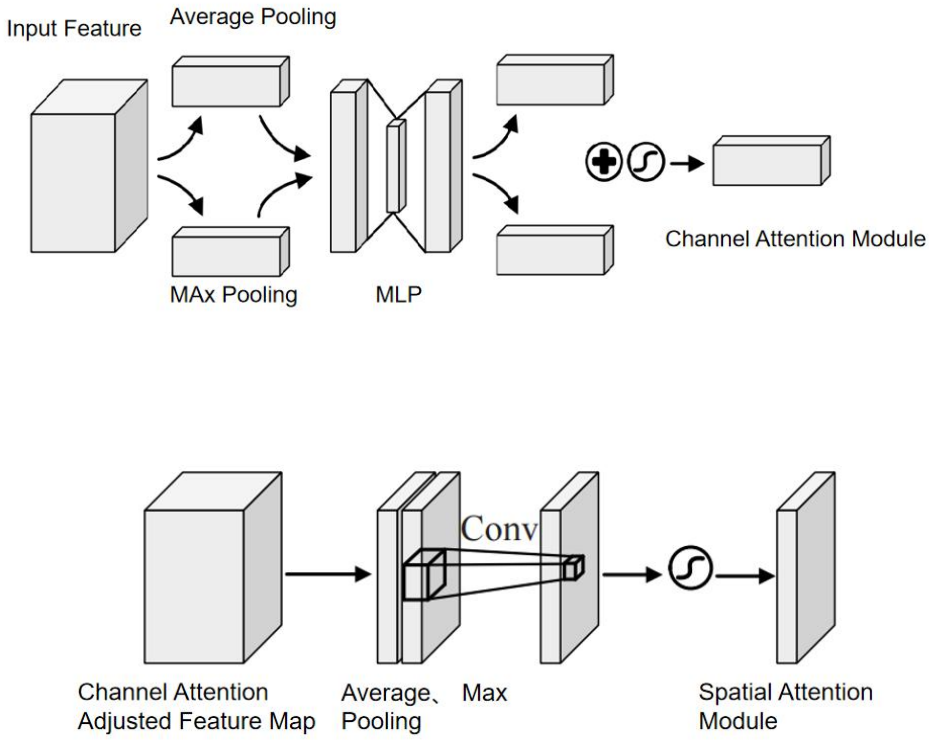


Fig.2.8 Structure of CAM and SAM

One of the key functions of the attention mechanism is to reconstruct the feature map, emphasizing important information while suppressing generic information. In the YOLOv5s network, the most critical phase of feature extraction occurs in the backbone. Therefore, this article integrates CBAM after the Backbone and before the Neck network for feature fusion. The rationale for this placement is that YOLOv5s completes feature extraction in the Backbone, and following the feature fusion in the Neck, it performs predictions on different feature maps.

Implementing CBAM at this juncture serves to bridge past and future processes. The specific structure is illustrated in Figure 2.9.

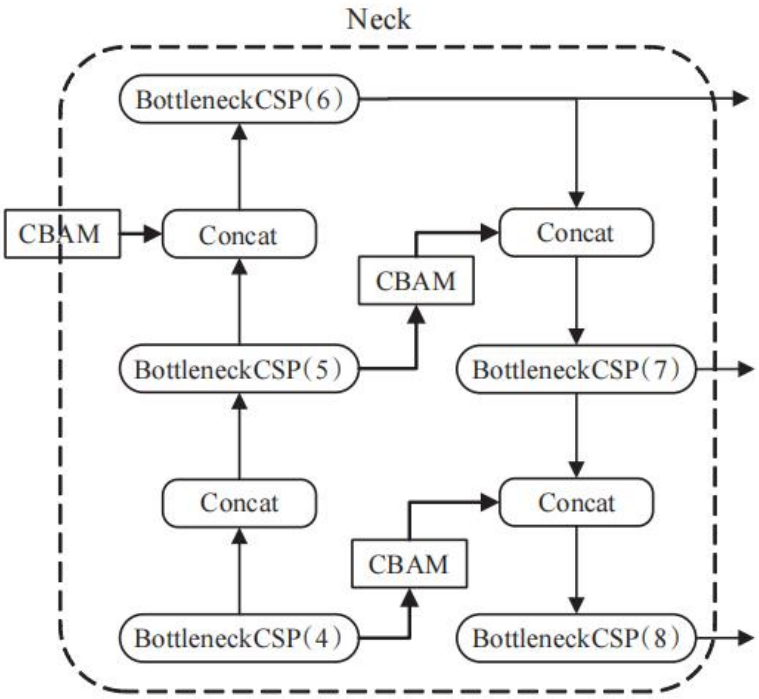


Fig.2.9. Structure of Neck integrating CBAM

(4) Loss function improvement

YOLOv5s uses GIoU Loss as the bounding box regression loss function to judge the distance between the predicted box(PB)and the ground truth(GT),such as Equation (2.11):

$$\begin{cases} \mathrm{GIoU} = \mathrm{IoU} - \frac{\mathrm{A^c} - \mathrm{U}}{\mathrm{A^c}} \\ \mathrm{L_{GIoU}} = 1 - \mathrm{GIoU} \end{cases} \quad (2.11)$$

In the Equation, IoU represents the intersection ratio of PB and GT, $\mathrm{A^c}$ represents the area of the smallest rectangular box that includes PB and GT at the same time, U represents the union of PB and GT, and $\mathrm{L_{GIoU}}$ is the GIoU loss. The advantage of GIoU Loss is scale invariance, that is,

the similarity of PB and GT has nothing to do with their spatial scale. The problem with GIoU Loss is that when PB or GT is completely surrounded by the opponent, GIoU Loss completely degenerates into IoU Loss. Because it relies heavily on IoU items, the convergence speed in actual training is too slow, and the accuracy of the predicted bounding box is relatively low. For these problems, CIoU Loss also considers the overlapping area of PB and GT, the distance between the center points, and the aspect ratio, such as Equation (2.12):

$$\begin{cases} \mathrm{CIoU} = \mathrm{IoU} - \frac{\rho^2(\mathrm{b},\mathrm{b}^{\mathrm{gt}})}{\mathrm{C}^2} - \alpha\gamma \\ \mathrm{L_{GIoU}} = 1 - \mathrm{CIoU} \end{cases} \quad (2.12)$$

In the Equation, b and b^{gt} represent the center points of PB and GT, $\rho^2(...)$ represents the Euclidean distance, c represents the shortest diagonal length of the smallest bounding box of PB and GT, α represents a positive balance parameter, γ represents the consistency of the aspect ratio of PB and GT. α and γ are defined as Equation (2.13):

$$\begin{cases} \gamma = \frac{4}{\pi^2}(\arctan\frac{\mathrm{w}^{\mathrm{gt}}}{\mathrm{h}^{\mathrm{gt}}} - \arctan\frac{\mathrm{w}}{\mathrm{h}})^2 \\ \alpha = \frac{\gamma}{(1-\mathrm{IoU})+\gamma} \end{cases} \quad (2.13)$$

which w^{gt} and h^{gt} represent the width and height of GT and PB, respectively.

Compared with the GIoU Loss and CIoU Loss used in YOLOv5s, the penalty items of PB, GT center distance and aspect ratio are added to the loss item, so that the network can ensure faster convergence of the prediction frame during training and get a higher return. For positioning accuracy, this research uses CIoU Loss as the loss function of the Chinese Yam detection network.

(5) NMS non-maximum suppression improvement

In the prediction stage, Non-Maximum Suppression (NMS) is commonly employed to eliminate redundant detection frames. The decision to remove a detection frame is based on the Intersection over Union (IoU) ratio between a given frame and the frame with the highest prediction score. If the IoU exceeds a predefined threshold, the predicted frame will be discarded. While effective in general scenarios, this approach can be problematic in environments with densely packed targets. Due to mutual occlusion, detection frames for different targets may be closely positioned with significant overlap, leading to inadvertent removal by NMS and, consequently, failure in target detection. In camera videos of yam fields, targets are often concentrated in the middle of the image, representing a densely populated scene prone to occlusion. To address this issue, this part adopts the Distance-IoU (DIoU) as the criterion for NMS, aiming to improve target detection in such challenging environments.

DIoU considers the distance between the center points of two bounding boxes on the basis of IoU, as in Equation (2.14):

$$\mathrm{DIoU} = \mathrm{IoU} - \frac{\rho^2(\mathrm{b},\mathrm{b}^{\mathrm{gt}})}{\mathrm{C}^2} \qquad (2.14)$$

b and b^{gt} represent the center points of PB and GT, $\rho^2(...)$ represents the Euclidean distance, c represents the shortest diagonal length of the smallest bounding box of PB and GT.

DIoU-NMS is defined as Equation (2.15):

$$\mathrm{s_i} = \begin{cases} \mathrm{s_i}, \mathrm{DIoU}(\mathrm{M}, \mathrm{B_i}) < \varepsilon \\ 0, \mathrm{DIoU}(\mathrm{M}, \mathrm{B_i}) \geq \varepsilon \end{cases} \qquad (2.15)$$

M represents a prediction frame with the highest prediction score, $\mathrm{B_i}$ represents the prediction frame that needs to be removed, $\mathrm{s_i}$ represents the classification score, and ε

represents the threshold of NMS.DIoU-NMS considers the IoU while judging the distance between the center points of the two bounding boxes M and B. When the distance is far away, the prediction box will not be removed, but another target is detected, which helps to solve the mutual occlusion of the targets. The problem of missed detection in the case. This research uses DIoU-NMS to replace the original NMS.

2.1.3 Deep-SORT object tracking

The multi-target online tracking algorithm SORT (Simple Online and Real-Time Tracking) employs the Kalman filter and Hungarian matching algorithm. It utilizes the Intersection over Union (IoU) between tracking and detection results as a cost matrix to implement a straightforward, efficient, and practical tracking approach. However, a limitation of the SORT algorithm is that its association metric is only effective when there is low uncertainty in state estimation. Consequently, this can lead to a high frequency of identity switches during the algorithm's execution, making tracking failures more likely. To address this issue, Deep-SORT enhances the algorithm by combining the target's motion information and appearance information as a correlation metric, thereby improving the robustness of target tracking.

(1) SORT algorithm process

The Kalman filter algorithm is one of the most commonly used algorithms in current mainstream tracking algorithms and is a fusion algorithm often used in continuously changing systems. Even in the presence of many uncertainties and external noise interference, the Kalman filter can make corresponding predictions based on the target's previous motion, thereby

obtaining the motion state information of the target at the next moment. In the prediction process, the state equation is constructed based on the model of the detected target. The state estimates obtained at different times are not the actual values, hence each state estimate at every moment carries uncertainty. To better represent the degree of uncertainty in the estimated states in the Kalman filter and the correlation between state variables (such as the correlation between the position and velocity of a moving target), a covariance matrix P is usually used to represent this. The Equation (2.16):

$$P_{K|K-1} = F_K P_{K-1} F_K^T \qquad (2.16)$$

At the same time, in practical application scenarios, such as the control input vector mentioned earlier, the estimation of predicted values will inevitably be subject to some noise interference. Therefore, it is essential to consider the impact of the external environment. Generally, this noise cannot be measured or eliminated through mathematical modeling and is typically represented by a covariance matrix denoted as Q. Hence, by estimating the uncertainty of the state and the uncertainty of external noise interference, we can derive the second formula in the Kalman filter prediction process, as shown in Equation (2.17):

$$P_{K|K-1} = F_K P_{K-1} F_K^T + Q_K \qquad (2.17)$$

The core of the SORT algorithm lies in the Kalman filter algorithm and the Hungarian algorithm. The primary function of the Kalman filter is to predict the next moment's motion variables based on the current series of motion variables, with the first detection result being used to initialize the Kalman filter's motion variables. In simple terms, the Hungarian algorithm is

used to solve allocation problems. It allocates a set of detection frames and frames predicted by Kalman, enabling the Kalman-predicted frames to find the best matching detection frame, thus facilitating effective tracking.

The workflow of the entire algorithm is as follows:

(1) Create corresponding Tracks from the detection results of the first frame. Initialize the motion variable of the Kalman filter and predict its corresponding frame using the Kalman filter.

(2) Perform IOU matching between the frame of the frame target detection and the frame predicted by Tracks from the previous frame, then calculate the cost matrix based on the IOU matching result (calculated as 1-IOU).

(3) Use all the cost matrices obtained in step (2) as inputs for the Hungarian algorithm to achieve linear matching. At this point, three outcomes are generated. The first is Unmatched Track, where we directly delete the mismatched Tracks; the second is Unmatched Detection, where we initialize such Detection as a new Track; the third is the successful pairing of the detection frame and the predicted frame, indicating successful tracking from the previous to the next frame, with the corresponding Tracks variable updated through the Kalman filter for the corresponding Detection.

(4) Repeat steps (2)-(3) until the end of the video frames.

(2) Deep-SORT algorithm process

The DeepSort target tracking algorithm primarily predicts target positions based on the intersection of the normal distribution functions of predicted effective targets and the actual

calculated normal distribution functions. It extracts and predicts information about the target in the next frame based on the effective information of the features. The matching results of the DeepSort algorithm are shown in Figure 2.10.

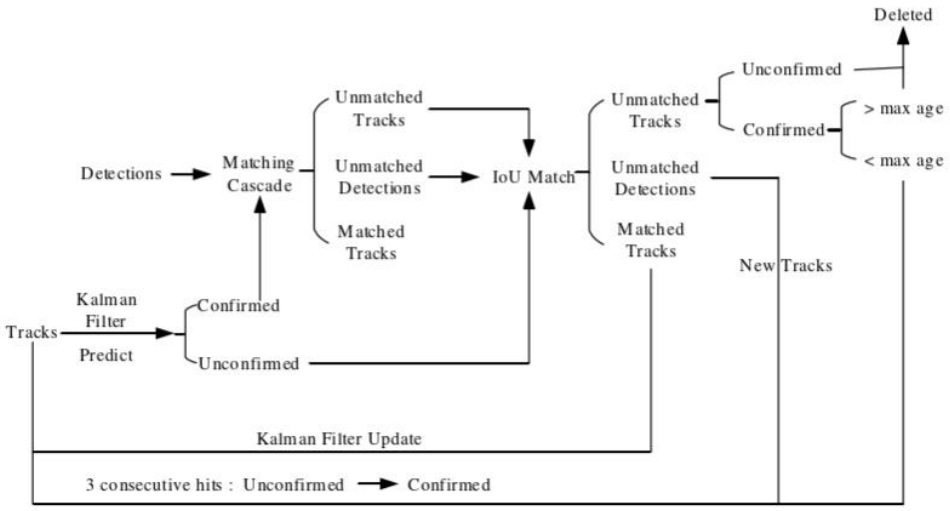


Fig.2.10. Matching result graph

The workflow of the entire algorithm is as follows:

(1) Create corresponding Tracks from the detection results of the first frame. Initialize the motion variable of the Kalman filter and predict its corresponding frame using the Kalman filter. At this stage, Tracks must be unconfirmed.

(2) Perform IOU matching between the frame from the target detection and the frame predicted by Tracks in the first frame, then calculate the cost matrix using the IOU matching result (calculated as 1-IOU).

(3) Use all the cost matrices obtained in step (2) as inputs for the Hungarian algorithm to achieve linear matching. Three outcomes are possible at this point. The first is Unmatched Tracks; we directly delete these mismatched Tracks, as they are in an uncertain state. If they are in a definite

state, deletion can only occur after a certain number of occurrences (default 30 times). The second is Unmatched Detections; we initialize such detections as new Tracks. The third is the successful pairing of the detection frame and the predicted frame, indicating successful tracking from the previous to the next frame. The corresponding Detections update their respective Tracks variables through the Kalman Filter.

(4) Repeat steps (2)-(3) until confirmed Tracks appear or the video frame ends.

(5) Predict the boxes corresponding to the Tracks in both confirmed and uncertain states using the Kalman filter. Cascade match the frames from the confirmed Tracks with the Detections. Save the appearance features and motion information of the Detections each time Tracks are matched. By default, the first 100 frames are saved, and this information is used for cascade matching with the Detections, as confirmed state Tracks and Detections are more likely to match.

(6) Three outcomes are possible after cascade matching. The first one is Tracks matching, where such Tracks update their corresponding variables through Kalman filtering. The second and third are mismatches between Detections and Tracks. In this case, previously unconfirmed Tracks and mismatched Tracks will be individually matched with Unmatched Detections for IOU matching. The cost matrix will then be calculated based on the IOU matching results, using the calculation method of 1-IOU.

(7) Repeat steps (5)-(6) until the end of the video frames.

(3) State Estimation and Tracking Processing

Deep-SORT uses the result of the detector to initialize the tracker. Each tracker will set a

counter. After Kalman filtering, the counter is accumulated. When the prediction result and the detection result successfully match, the counter is set to 0.If a tracker does not match a suitable detection result within a period of time, the tracker is deleted.Deep-SORT assigns a tracker to the new detection results in each frame. When the prediction results of the tracker match the detection results for 3 consecutive frames, it is confirmed that a new track has appeared, otherwise the tracker is deleted.

Deep-SORT uses an 8-dimensional state space (u, v, γ , h, $\dot{x}$, $\dot{y}$, $\dot{\gamma}$, $\dot{h}$) to describe the state of the target and its motion information in the image coordinate system. u and v represent the center coordinates of the target detection frame, γ and h represent the aspect ratio and height of the detection frame respectively, and ($\dot{x}$, $\dot{y}$, $\dot{\gamma}$, $\dot{h}$) represent the relative speed of the first four parameters in image coordinates. The algorithm uses a standard Kalman filter with a constant velocity model and a linear observation model, taking the bounding box parameters (u,v, γ ,h) as direct observations of the object state.

(4) Allocation Problem

Deep-SORT combines motion information and appearance information, and uses the Hungarian Algorithm to match prediction boxes and tracking boxes. For motion information, the algorithm uses the Mahalanobis distance to describe the degree of correlation between the Kalman filter prediction results and the detector results, such as the Equation (2.18):

$$\mathrm{d}^{(1)}(i,j) = (d_j - y_i)^T S_i^{-1} (d_j - y_i) \qquad (2.18)$$

In the Equation, dj and yi represent the state vectors of the j-th detection result and the i-th

prediction result respectively, and Si represents the covariance matrix between the detection result and the average tracking result.

The Mahalanobis distance measures the standard deviation of the detection results from the average tracking results, taking into account the uncertainty of the state estimation, and can exclude low-probability associations.

When the uncertainty of target motion information is low, the Mahalanobis distance is a suitable correlation factor, but when the target is occluded or the camera perspective shakes, only using the Mahalanobis distance correlation will cause the target identity to switch. Therefore, consider adding appearance information, calculate the corresponding appearance feature descriptor rj for each detection frame dj, and set // rj // =1 . For each tracking track k, set the feature warehouse Rk= $\{\mathrm{r}_k^{(i)}\}_{k=1}^{L_k}$, which is used to save the feature descriptors of the last 100 objects successfully associated, Lk = 100. Calculate the minimum cosine distance between the i-th tracking frame and the j-th detection frame, such as the Equation(2.19):

$$\mathrm{d}^{(2)}(i,j)=\min\{1-r_j{}^T r_k^{(i)} \,\big|\, r_k^{(i)} \in R_i\} \qquad (2.19)$$

When $\mathrm{d}^{(2)}(i,j)$ is less than the specified threshold, the association is considered successful.The Mahalanobis distance can provide reliable target location information in the case of short-term prediction, and the cosine similarity of appearance features can be used to restore the target ID when the target is occluded and reappears. In order to make the advantages of the two metrics complement each other, a linear weighting method is used to combine:

$$\mathrm{c}_{i,j}=\lambda d^{(1)}(i,j)+(1-\lambda)d^{(2)}(i,j) \qquad (2.20)$$

(5) Deep Appearance Features

Before the adoption of convolutional neural network-based tracking algorithms, target tracking algorithms primarily relied on pure algorithms, with the cost matrix as the core algorithm for tracking. However, there were issues in the tracking process, such as the extraction of an object's feature information. Merely using two-dimensional information is not sufficient to fully represent these features. Therefore, using only the cost matrix as the core algorithm does not fully express the tracked object's feature information. After the application of convolutional neural networks in target tracking algorithms, the SORT algorithm uses convolutional neural networks to extract the appearance features of the target object to generate the cost matrix, thereby improving target tracking. The DeepSort algorithm, on the other hand, combines the Mahalanobis distance related to motion information and appearance feature information to generate the cost matrix for feature extraction, and it also includes cascade matching to correlate target data, thus accurately tracking target information.

Although the DeepSort tracking algorithm has improved stability and precision in tracking Chinese yams compared to the SORT algorithm, and has reduced the frequency of ID switches during tracking, its ability to extract features of target appearance in complex lighting backgrounds is still relatively weak. This paper introduces the Darknet53 network as the primary method for feature extraction, enhancing tracking precision in conjunction with the tracking algorithm. The structure of the Darknet53 network is shown in Figure 2.11.

	Type	Filters	Size	Output
	Convolutional	32	3×3	256×256
	Convolutional	64	3×3/2	128×128
1×	Convolutional	32	1×1	
	Convolutional	64	3×3	
	Residual			128×128
	Convolutional	128	3×3/2	64×64
2×	Convolutional	64	1×1	
	Convolutional	128	3×3	
	Residual			64×64
	Convolutional	256	3×3/2	32×32
8×	Convolutional	128	1×1	
	Convolutional	256	3×3	
	Residual			32×32
	Convolutional	512	3×3/2	16×16
8×	Convolutional	256	1×1	
	Convolutional	512	3×3	
	Residual			16×16
	Convolutional	1024	3×3/2	8×8
4×	Convolutional	512	1×1	
	Convolutional	1024	3×3	
	Residual			8×8
	Avgpool		Global	
	Connected		1000	
	Softmax			

*Fig.2.*11 *Darknet53 network structure diagram*

The original algorithm used a residual convolutional neural network to extract the appearance features of targets and was trained on a large-scale pedestrian re-identification dataset, making it suitable for pedestrian detection and tracking. As the original algorithm was solely used for pedestrian categories, all input images were scaled to 128x64, which is inconsistent with the aspect ratio of Chinese yam targets. To adapt the model for Chinese yam feature extraction, the network model was modified. The adjusted network input image size is 256x64.

2.2 Results and Discussion

2.2.1 Dataset and Experimental Environment

The dataset used in the experiment is mainly from the Michishita Hironaga Farm in Obihiro City, Hokkaido. The camera used is GoPro Hero 7 Black, and pixel count is 2.76 million pixels and shot at 24fps. The camera is equipped with image stabilization for mounting the tractor on unstable fields. Fix the GoPro camera on a tractor running in parallel and at constant speed, taking a bird's-eye view of the Chinese Yams placed on the field. During image acquisition, frame-triggered mode was used to capture images of Yams moving at a low speed, as in Fig.2.12. The experiment uses PyTorch as the software framework, and the model training hardware environment is Intel(R) Core(TM) i7−11800H (16 GB) and NVIDIA GeForce RTX 3060 Laptop GPU (6 GB).

*Fig.2.*12 *GoPro and dataset collection scenario*

2.2.2 Parameter setting and evaluation index

Model training parameter settings: the input image size is 600×450, the number of iterations is 100, the batch size is 16, and the initial learning rate is 0.001.

The recall rate M_r, the average precision M_p, the average missed detection rate M_m, and the average false detection rate M_f are used as the evaluation criteria of the target detection model.

$$
\begin{aligned}
M_r &= \frac{T_P}{T_P + F_N} \\
M_p &= \frac{T_P}{T_P + F_P} \\
M_m &= \frac{F_N}{F_N + T_P} \\
M_f &= \frac{F_P}{F_P + T_N}
\end{aligned}
\qquad (2.21)
$$

In the Equation(2.21): T_P is the yam that is correctly detected; F_N is the yam that is not detected; F_P is the yam that is falsely detected; T_N is the yam that is not falsely detected.

Define the number of real trajectories in the t frame as g_t, the number of successfully matched trajectories in these trajectories is recorded as c_t, the matching cost of the i pair of successful matches is recorded as d_t^i, and the true trajectories that have not been successfully matched are recorded as mt.The trajectory predicted by the model but not successfully matched in the t frame is recorded as fp_t. If there is an inconsistency in trajectory matching in two adjacent frames, it means that identity switching has occurred, and the number is recorded as mme_t. Based on these definitions, some comprehensive multi-target tracking model performance

evaluation indicators can be generated.

$$1-\frac{\sum_{\mathrm{t}}\left(m_t+fp_t+mme_t\right)}{\sum_t g_t} \qquad (2.22)$$

The number of identity switching IDs and the number of tracking frames per second speed are used as the evaluation criteria of the tracking model. The statistical accuracy of Chinese Yam flow is used as the evaluation standard of the whole scheme.

2.2.3 YOLOv5s ablation and comparison experiment

In this paper, the dataset is randomly divided in a 6:2:2 ratio into training, validation, and testing sets for the detection network. To verify the three improvement strategies proposed for YOLOv5s, an ablation study was conducted on the dataset to assess the effectiveness of each enhancement. CBAM and CIoU Loss were sequentially added to the initial YOLOv5s model. The experiments did not utilize pre-trained models, and the training process followed the same parameter configuration.

CBAM	CIoU Loss	Precision	Recall	AP@0.5
×	×	90.2	93.9	93.3
√	×	92.6	96.8	95.1
×	√	91.0	93.7	94.2
√	√	93.9	96.6	95.8

*Table 2.*1 *Ablation of YOLOv5s*

The first row of Table 2.1 displays the baseline performance of the original YOLOv5s on the dataset, achieving an average detection accuracy of 93.3%. After introducing CBAM and CIoU Loss separately, it is observed that CBAM more significantly enhances detection results, with notable improvements in Precision, Recall, and AP, while the impact of CIoU Loss is slightly less pronounced. This difference is attributed to the distinct functions of the two modules. The attention mechanism, CBAM, is designed to enhance the network's ability to extract important features, resulting in increased accuracy. In contrast, CIoU Loss focuses on speeding up the regression of the prediction frame and improving regression accuracy, leading to only a modest improvement in detection accuracy. When both CBAM and CIoU Loss are introduced together, the detection network achieves optimal results, with the average precision AP increasing by 2.5 percentage points compared to the original network. Some test results of CIoU Loss and DIoU-NMS are visualized in Figure 2.13, demonstrating the detection of previously missed Chinese yams while maintaining high detection accuracy.

Fig.2.13 CBAM+CIoU Loss+DIoU-NMS of ablation result

For horizontal comparison, this part selects Faster R-CNN+FPN, YOLOv3+SPP, and mobilenetv2-YOLOv4 to train and test on the same data set, all using pre-trained models. The experimental results are shown in Table 2.2. It can be seen that the improved YOLOv5s in this research is ahead of the other three networks in terms of detection rate FPS, weight size, and average precision AP. For the network weight size, the original YOLOv5s is 7.2 MB, and the improved network only increases 0.5MB.

Model Name	FPS	Weight Size/MB	AP@0.5
Faster R-CNN+FPN	12	311	88.7
YOLOv3+SPP	56	238	92.1
Mobilenetv2_YOLOv4	82	47.59	93.5
Proposed	120	7.7	95.8

Table 2.2 Comparison of different detection networks

2.2.4 Deep Appearance Feature Extraction Network Experiment

The adjusted re-identification network is trained on the data-set YaRi, the input image size is 256×64, and the other parameters remain unchanged. Connect the improved YOLOv5s and Deep-SORT after yam re-identification for testing, the results are shown in Table 2.3

Model Name	IDs/time	Speed/Hz
SORT	65	60
Deep-SORT	28	33
YaRi Deep-SORT	22	31

Table 2.3 Chinese Yam tracking experiment

Since the SORT algorithm only uses motion features as the basis for target association, a total of 65identity switches occurred in the yam tracking of the above data. Compared with SORT, Deep-SORT reduced by 56%, and the model after the yam re-identification in this research further reduces IDs, not only IDs are reduced to 22 times, but also the detection speed can reach

31Hz on the local test platform, which meet the standard of real-time detection.

2.2.5 Statistical experiment of Chinese Yam quantity

This section adapts statistical methods used in pedestrian and vehicle flow monitoring, with a key distinction: instead of moving detection targets and a fixed camera, it uses fixed detection targets and a moving camera. This approach involves setting a detection line in the video to count the flow of Chinese yams.

The method employed is as follows: a red dot is placed on the left border of the target tracking bounding box to represent the target's trajectory, as shown in Figure 2.14. A virtual detection line, perpendicular to the camera's direction of movement, is established on the field road, as depicted in Figure 2.15. When the trajectory of a point representing the target tracking frame intersects with the detection line, the total yam flow count is incremented, and the coordinates of the point are recorded. Since the detection targets pass in only one direction, there is no need for bidirectional counting. The test results are presented in Figure 2.16.

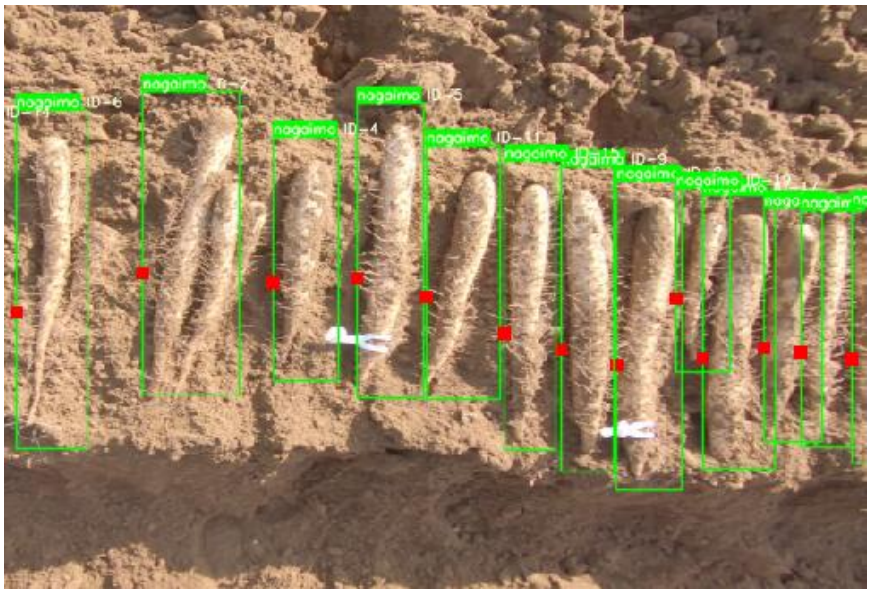


*Fig.2.*14 *Target Tracking Bounding Box With Dot*

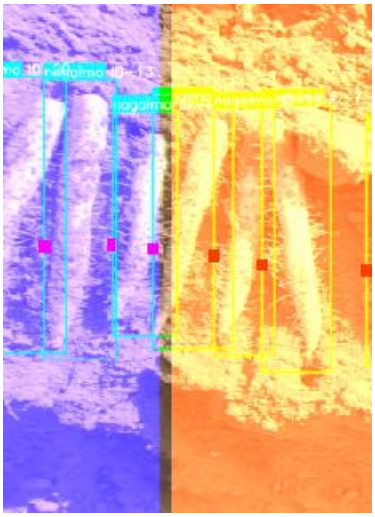


*Fig.2.*15 *Virtual Vertical Detection Line*

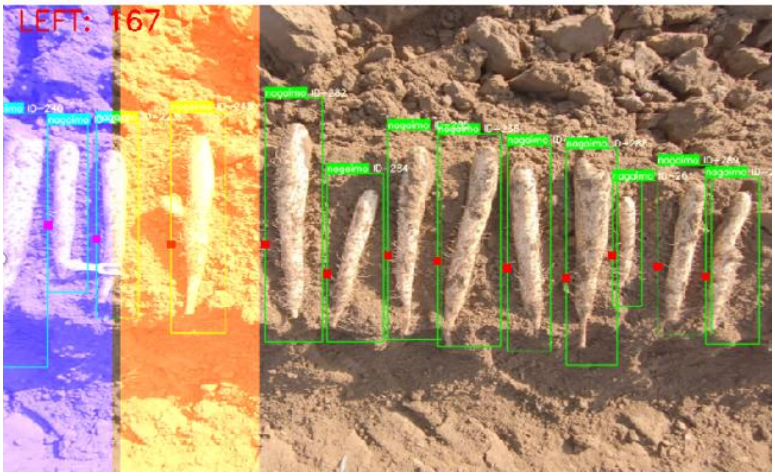


*Fig.2.*16 *Statistical Results of Chinese Yam Traffic*

The Chinese yam flow statistical test data collected in this part is manually counted and compared with the experiment. The results are shown in Table 2.4. Each row of data represents the traffic flow counted by manual statistics, the original algorithm and the improved algorithm in the test video. From the results, it can be seen that the accuracy of the improved method proposed in this research is higher than that of the original algorithm. Due to light problems and soil coverage on the surface of yams, some yams may be missed, which affects the accuracy of

target tracking.

Model	Result
Manual Count	197
Original YOLOv5s+Deep-SORT	153
Proposed YOLOv5s+Deep-SORT	181

Table 2.4 Chinese Yam tracking experiment

2.2.6 Conclusion

In this chapter, the YOLOv5s model is strategically employed as a detector and seamlessly integrated with the Deep-SORT target tracking methodology to accurately count the number of Chinese Yams. This integration significantly amplifies the detector's accuracy, a result of synergizing the CBAM attention mechanism with the robust capabilities of YOLOv5s. The adoption of the CIoU Loss function and the DIoU-NMS non-maximum suppression method, in lieu of the conventional GIoU Loss and standard NMS, marks a pivotal improvement. This innovative replacement not only refines the detector's localization precision but also markedly diminishes the occurrences of missed detections, particularly in complex scenarios where Chinese Yams are densely clustered or overlapping.

Further augmenting the system's proficiency, the original feature extraction network within Deep-SORT has been meticulously tailored for enhanced input adjustment and re-identification training. This modification renders the algorithm exceptionally compatible with the unique attributes of root crops, such as Chinese Yams. By integrating the advanced YOLOv5s detector

into the algorithmic framework, a comprehensive experiment was conducted to enumerate Chinese Yams. The experimental outcomes underscore the algorithm's exemplary statistical precision, demonstrating its effectiveness in accurately identifying and counting Chinese Yams, even in challenging agricultural environments. This advancement in detection and tracking technology not only showcases the versatility of the integrated system but also opens up new avenues for precision agriculture and automated crop monitoring.

Chapter 3 Defect detection and size grading of yam in harvest

3.1 *Methodology*

Quality assessment of Chinese Yam is an essential step in detecting defects and grading sizes before market entry. Implementing quality inspection and graded sales enhances the overall quality and market competitiveness of Yams. Currently, the inspection and grading of Yam quality predominantly rely on manual processes. However, these methods face challenges such as low efficiency, inconsistent accuracy, and a lack of uniformity. These issues significantly impede the quality and efficiency of Yam grading, ultimately affecting market performance.

While traditional techniques have seen some success in detecting defects in Yams, they are inherently complex, involving multiple steps and suffering from notable limitations in terms of accuracy and flexibility. This complexity restricts their practical applicability.

Deep learning, with its automatic feature extraction capabilities, offers significant

improvements over conventional image processing methods. It has already shown promising results in detecting defects in fruits and vegetables like apples, cucumbers, and carrots, and is increasingly being applied in assessing the appearance quality of root crops. Despite the advancements in the performance of deep learning methods, they still face challenges related to efficiency.

In the realm of size grading research, past studies have typically focused on classifying agricultural products into 'normal' and 'defective' categories, without further grading within the 'normal' group. In real-world Yam processing, workers often first sort out defective Yams and then classify the remaining normal Yams into different grades based on size. Therefore, there is a pressing need to establish a specialized image acquisition system and develop effective methods for both detecting and grading Yam defects.

The goal of this section is to devise an automated, online system capable of fulfilling the requirements for accurate, real-time defect detection and the automatic grading of Yams. This system is envisioned to revolutionize the Yam grading process, offering a high degree of precision and efficiency. The development of such a system will not only streamline the grading process but also ensure consistent quality standards, thereby boosting the marketability and consumer appeal of Chinese Yams.

3.1.1 Materials

The underground tuber is the primary edible part of the Chinese Yam. During harvest, a specialized two-tractor system is employed. The first tractor is fitted with a plow that efficiently

uproots the Yams, which are then promptly loaded into containers on the following tractor. Between these tractors, a team of workers diligently removes the soil from the unearthed Yams and organizes them in the field. Following this, additional personnel behind the second tractor engage in a preliminary manual inspection and grading process. They meticulously sort the Yams, segregating any defective ones and classifying the ordinary Yams into various size-based grades. This paper focuses on Yams of diverse sizes and shapes as experimental samples, encompassing a range from healthy, normal Yams to those with defects such as curvature, forked roots, fractures, or flattening, as illustrated in Fig.3.1.

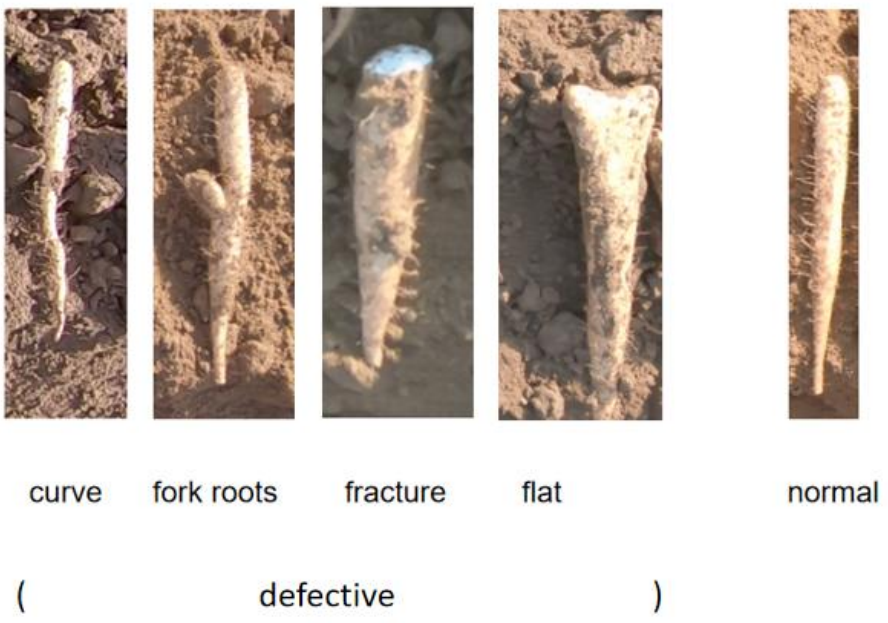


*Fig. 3.*1. *Defective Chinese Yams and Normal Chinese Yam.*

3.1.2. Dataset collection and Image acquisition

In the development of an automated yam quality assessment system, our approach encompasses a comprehensive four-step process: dataset collection and image acquisition, image preprocessing, defect detection, and yam grading. The initial phase, dataset collection and image acquisition, is pivotal as it establishes the foundation for the entire system. During this stage, a diverse array of yam images is captured under various conditions, ensuring a rich dataset that includes multiple representations of yam defects and varying degrees of ripeness. This step is critical for the success of the subsequent stages, as the quality and diversity of the image dataset directly influence the system's capability to accurately identify and classify defects.

The second step, image preprocessing, involves a series of techniques aimed at enhancing the image quality and preparing the data for effective analysis. This includes standardization of image sizes, adjustment of lighting conditions, and the application of filters for noise reduction. By refining the image quality, this step ensures that the defect detection algorithms can operate more efficiently and accurately.

In the third phase, defect detection, advanced image processing algorithms are employed to analyze the preprocessed images. The goal here is to accurately identify any deviations from the standard yam quality, such as blemishes, discolorations, or irregular shapes. This phase often leverages sophisticated machine learning techniques, including convolutional neural networks, to discern subtle defects that might be missed by the human eye.

The final step in our system is yam grading. Based on the outcome of the defect detection analysis, each yam is assigned a grade that reflects its overall quality. This grading process categorizes yams into various classes, ranging from premium to lower quality, based on established industry standards or specific customer requirements. The grading system is crucial for determining the market value of each yam and ensuring that quality standards are consistently met.

Overall, this four-step process forms a robust framework for automated yam quality assessment, enhancing both the efficiency and accuracy of yam grading in agricultural and commercial contexts. The integration of advanced image processing and machine learning techniques is instrumental in achieving high precision in defect detection and grading, thereby promising significant improvements in the quality control measures for yam distribution. as presented in Fig.3.2.

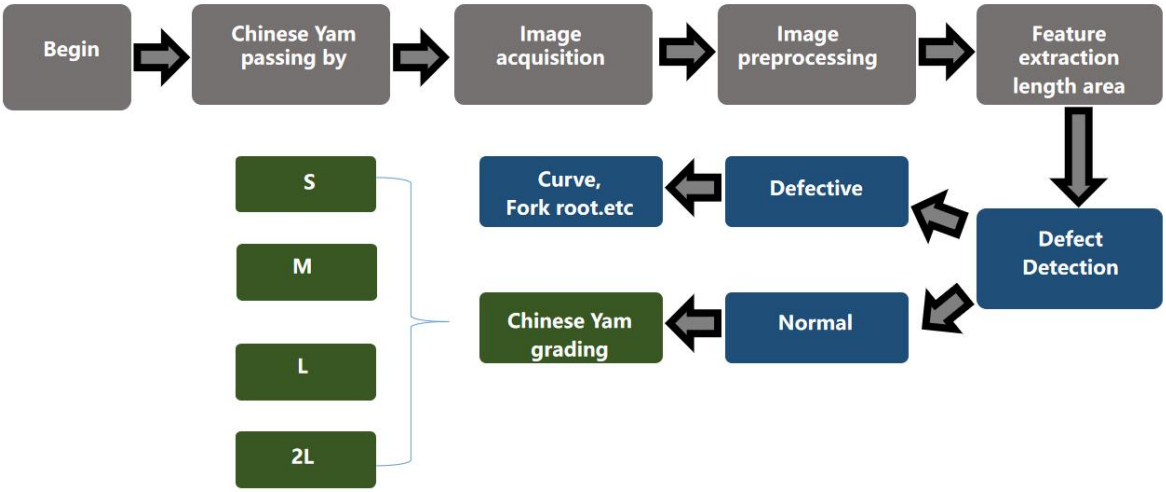


Fig. 3.2. System technical flow chart

In our methodology, during the image acquisition phase, a specialized approach was adopted to ensure the capture of high-quality images of yams. We utilized a frame-triggered mode to photograph yams moving at a controlled, low speed. This method was particularly effective in maintaining consistency in the imaging process, as it allowed for precise timing in capturing images, thereby reducing motion blur and ensuring clarity. The low-speed movement of the yams was critical in achieving uniform illumination across each image, which is a vital factor in subsequent image analysis stages.

To address the challenges associated with varying lighting conditions and to achieve brightness equalization in the captured images, we employed a novel approach centered around the Ohta color space. The Ohta color space, known for its effectiveness in color segmentation and insensitivity to illumination changes, provided a robust framework for our grayscale conversion

method. By referring to the Ohta color space, we developed a new grayscale method that significantly enhanced the detection of subtle variations in yam surface textures and defects, which are often missed in standard RGB color space conversions.

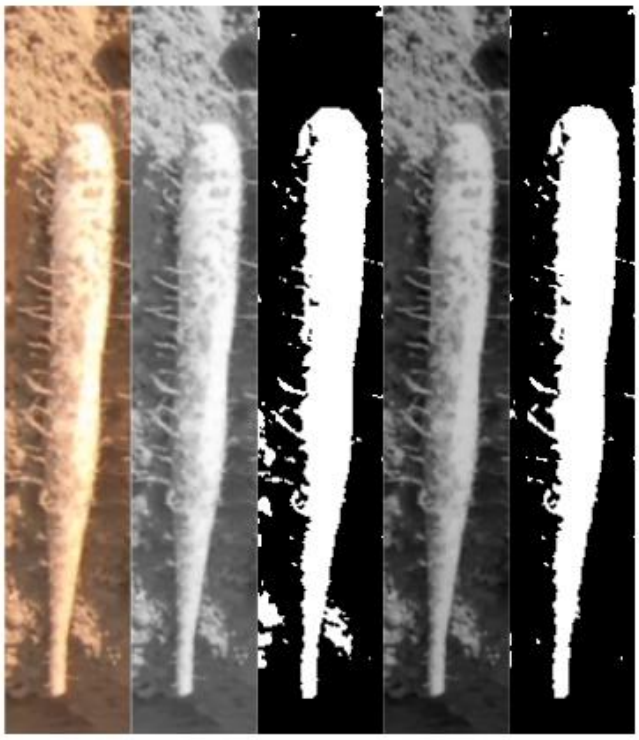
This grayscale method in the Ohta color space involved converting the color images into a format where the luminance (brightness) information is separated from the color information. By focusing on luminance, we could more accurately analyze the textural and shape-related features of the yams, which are crucial for defect detection. This method proved to be highly effective in reducing the influence of color variations and shadows, leading to more reliable and consistent image data for further processing steps.

The integration of the frame-triggered image acquisition mode with the advanced grayscale conversion technique based on the Ohta color space was a significant advancement in our image processing methodology. It ensured that the captured images of yams were of high quality, with uniform brightness and enhanced detail visibility, setting a strong foundation for the subsequent stages of image preprocessing, defect detection, and yam grading. This approach highlights the importance of innovative techniques in overcoming common challenges in agricultural image analysis, particularly in the context of automated quality assessment systems. we used a new grayscale method calculated by Eq (3.1)

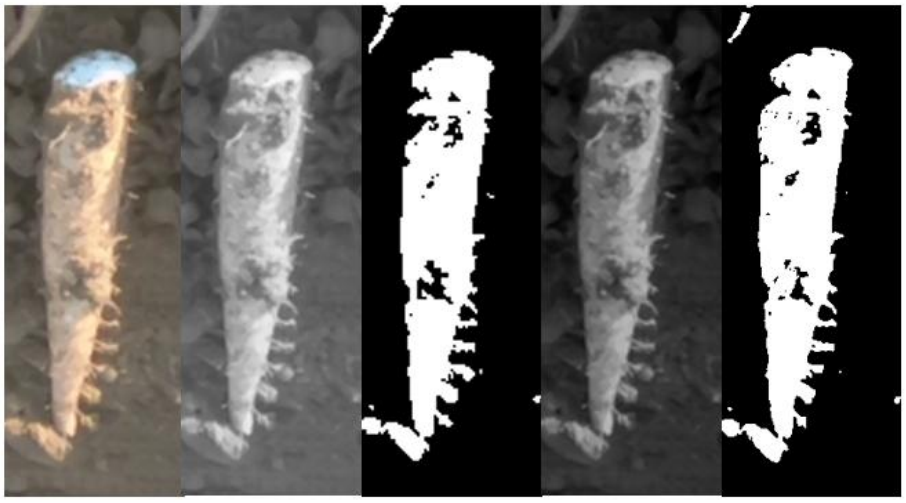
$$I_{gray} = 2.5I_r - 2I_g - 0.5I_b \qquad (3.1)$$

where I_r, I_g and I_b represent the red, green and blue components of the source image I_{gray}, respectively. The process of image preprocessing is illustrated in Fig.3.3.

(a)



(b)



(c)

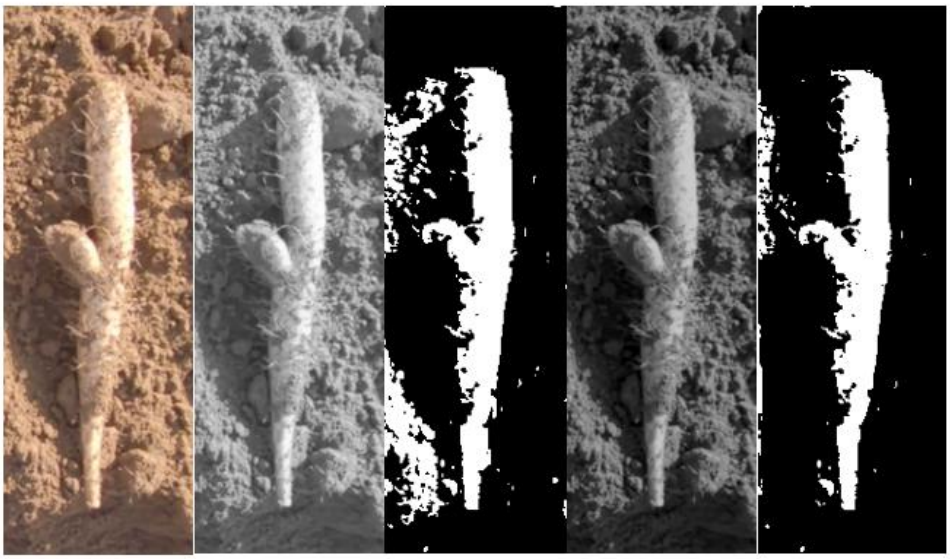


*Fig.3.*3 *Illustration of image preprocessing. (a) Original* Chinese *Yam image of normal; gray image and its binary images; gray image calculated by Eq.1 and its binary images. (b) Original* Chinese *Yam image of fracture; gray image and its binary images; gray image calculated by Eq.1 and its binary images. (c) Original* Chinese *Yam image of fork root; gray image and its binary images; gray image calculated by Eq.1 and its binary images.*

3.1.3. Chinese Yam defect detection model based on deep learning

In the development of our Yam Defect Detection Network (CDDNet), we strategically chose to base our architecture on ShuffleNet, a network known for its lightweight and low-complexity design. ShuffleNet is particularly renowned in the field of deep learning for its efficiency in

processing, which stems from its unique use of pointwise group convolutions and channel shuffle operations. These features enable ShuffleNet to maintain a balance between computational efficiency and model accuracy, making it an ideal choice for applications where resources are limited or where real-time processing is crucial.

Given the constraints often associated with training data in specialized domains like agricultural product analysis, we integrated transfer learning into our CDDNet. Transfer learning is a powerful technique in machine learning where a model developed for one task is repurposed on a second, related task. By leveraging pre-trained models on extensive datasets, transfer learning allows for significant improvements in learning efficiency and performance, especially in scenarios with limited training data. This approach is particularly effective in object recognition tasks, where nuanced features and patterns need to be discerned for accurate classification.

In the context of our system, the combination of ShuffleNet and transfer learning offers a highly effective solution for online defect detection of Chinese Yams. The lightweight nature of ShuffleNet ensures that our network remains computationally efficient, a critical factor for real-time processing in online systems. Concurrently, the application of transfer learning enhances the network's ability to accurately recognize and classify various defects in yams, despite the potential limitations in the volume and variety of the training data.

The CDDNet, therefore, represents an innovative approach in the realm of agricultural quality control, particularly suited to the challenges of detecting defects in Chinese Yams. This network not only provides the accuracy needed for thorough defect detection but does so with the efficiency required for integration into online processing systems, where speed and resource optimization are key considerations, as illustrated in Fig. 3.4

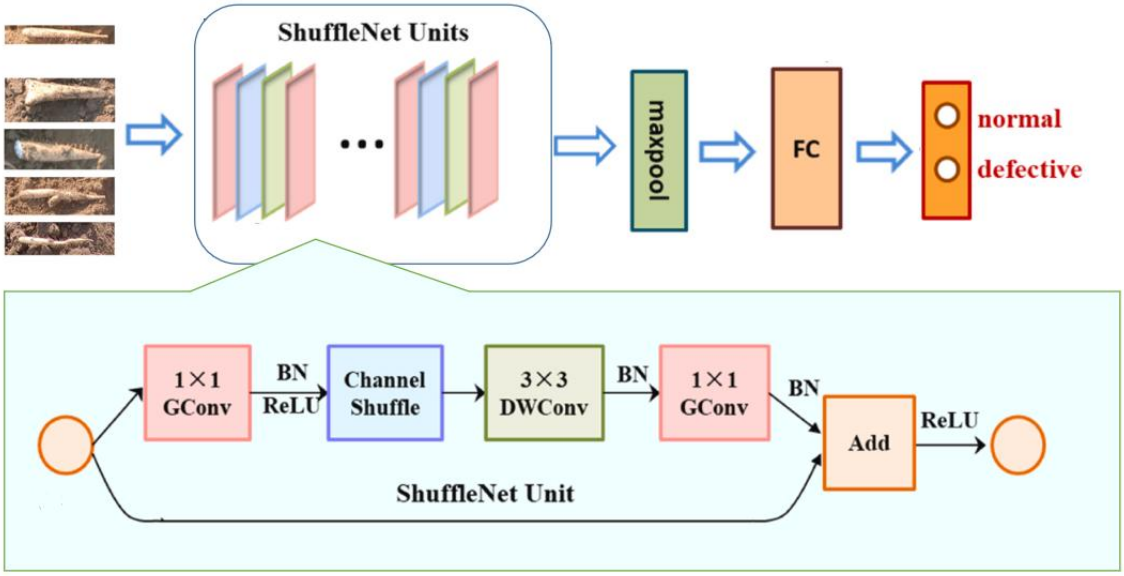


Fig.3.4 Model architecture of the proposed defect detection model. CDDNet model and its ShuffleNet unit.

The proposed defect detection model, CDDNet, incorporates the advanced ShuffleNet unit, a crucial element that significantly contributes to the model's efficiency and effectiveness. The ShuffleNet unit, a key component of our CDDNet architecture, operates on a unique workflow designed to optimize processing speed while maintaining high accuracy, making it particularly suitable for real-time applications like defect detection in agricultural products.The workflow of the ShuffleNet unit is characterized by its innovative use of grouped convolutions and a channel shuffle operation. The grouped convolution technique divides the input channels into several groups, and convolutions are performed within these groups. This approach reduces

computational complexity by decreasing the number of connections between layers compared to standard convolutions. However, this reduction could lead to information bottlenecks as each group is processed independently. To counter this, the ShuffleNet unit employs a channel shuffle operation, which effectively rearranges the output channels of grouped convolutions. This shuffling ensures cross-group information flow, allowing the network to maintain powerful representational capabilities despite the reduced computational cost.

Additionally, the ShuffleNet unit integrates depth-wise separable convolutions, further enhancing its efficiency. This involves decomposing a standard convolution into a depth-wise convolution and a point-wise convolution, drastically reducing the number of parameters and computational expense. The depth-wise convolution applies a single filter per input channel, and the point-wise convolution, which is essentially a 1x1 convolution, combines the outputs of the depth-wise layer. This separation allows for efficient feature extraction and combination, making the ShuffleNet unit highly effective for processing complex image data like that of yam defects.In the context of the CDDNet model, the incorporation of the ShuffleNet unit brings forth significant advantages. It enables the model to rapidly process high volumes of image data, essential for real-time defect detection, without compromising on the depth and quality of feature extraction and analysis. This balance of speed and accuracy is pivotal in achieving a reliable and efficient defect detection system, as required in the high-throughput and precision-oriented domain of agricultural quality control. The ShuffleNet unit, with its unique workflow, stands as a core component that elevates the CDDNet model to an advanced level of performance in the field of automated defect detection.

3.1.4. Size grading methods based on MBR fitting and convex polygon approximation

In our research, we developed an innovative approach for size grading of Chinese Yams, integrating Minimum Bounding Rectangle (MBR) fitting and convex polygon approximation methods. This technique was particularly designed to classify yams into different size specifications based on their length, following the initial defect detection phase. Understanding the correlation between the pixel dimensions in images and the actual physical dimensions of yams is crucial for accurate size grading. Our method focuses on accurately estimating this relationship, paving the way for precise and automated size classification.The process begins with the calculation of the pixel length of the yams, which is achieved by determining the minimum bounding rectangle (MBR) of the yam region in the image. The MBR fitting involves encapsulating the yam in the smallest possible rectangle within the image frame, thus providing a pixel-based representation of the yam's length and width. This method is advantageous as it simplifies the complex shape of the yam into a more manageable geometric form, allowing for easier and more accurate measurements.

However, to translate these pixel measurements into actual physical lengths, a new regression method was proposed. This method involves using linear fitting techniques to establish a robust relationship between the pixel length measured by the MBR and the real length of the Chinese Yam. By analyzing a dataset of yams with known dimensions, we were able to develop a regression model that accurately predicts the actual length of the yams based on their pixel representation in the images.

Furthermore, to enhance the accuracy of our size grading method, we incorporated convex

polygon approximation. This technique involves approximating the shape of the yam using a convex polygon, which provides a more precise representation of the yam's outline compared to the MBR. By combining the MBR fitting with convex polygon approximation, we were able to refine our measurements and improve the overall reliability of our size grading approach.

In summary, the integration of MBR fitting and convex polygon approximation into our size grading methods represents a significant advancement in the field of agricultural product processing. This approach not only facilitates accurate size classification of Chinese Yams but also exemplifies the potential of geometric and computational techniques in enhancing the efficiency of agricultural sorting and grading systems. The development of such sophisticated methods is crucial in meeting the growing demands for precision and automation in agricultural production and distribution.

Can be described as follows:

Step 1: Convert the source image (Fig. 3.5) into binary image.

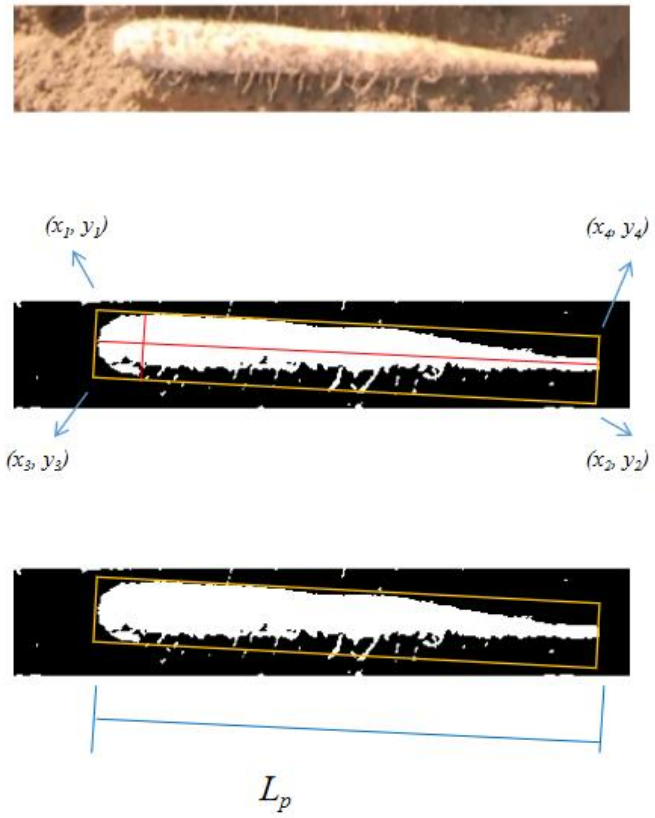


Fig. 3.5 Fitting of Yam length： Source image; minimum bounding rectangle computation; calculation of L_p;

Step 2: Compute the minimum bounding rectangle of the Chinese Yam region (Fig. 6) based on approximation algorithm, and MBR[15]corner point positions tl (tlx, tly), tr (trx, try), bl (blx, bly), br (brx, bry) can be obtained by the formula in Eqs. (3.2)–(3.5):

$$\begin{aligned} tl_x &= \frac{x_1 \tan\theta + x_3 \cot\theta + y_3 - y_1}{\tan\theta + \cot\theta} \\ tl_y &= \frac{y_1 \cot\theta + y_3 \tan\theta + x_3 - x_1}{\tan\theta + \cot\theta} \end{aligned} \tag{3.2}$$

$$\begin{aligned} tr_x &= \frac{x_1 \tan\theta + x_4 \cot\theta + y_4 - y_1}{\tan\theta + \cot\theta} \\ tr_y &= \frac{y_1 \cot\theta + y_4 \tan\theta + x_4 - x_1}{\tan\theta + \cot\theta} \end{aligned} \tag{3.3}$$

$$\begin{aligned} \mathrm{b}l_x &= \frac{x_2 \tan\theta + x_3 \cot\theta + y_3 - y_2}{\tan\theta + \cot\theta} \\ bl_{\mathrm{y}} &= \frac{y_2 \cot\theta + y_3 \tan\theta + x_3 - x_2}{\tan\theta + \cot\theta} \end{aligned} \quad (3.4)$$

$$\begin{aligned} \mathrm{br}_x &= \frac{x_2 \tan\theta + x_4 \cot\theta + y_4 - y_2}{\tan\theta + \cot\theta} \\ br_{\mathrm{y}} &= \frac{y_2 \cot\theta + y_4 \tan\theta + x_4 - x_2}{\tan\theta + \cot\theta} \end{aligned} \quad (3.5)$$

In the realm of bounding rectangle, where θ is the angel (radian) of the object orientation. (x_1, y_1) is the top edge point of Chinese Yam region, (x_2, y_2) is the bottom edge point, (x_3, y_3) is the left edge point, and (x_4, y_4) is the right edge point. These markers are used to define the four corners of the rectangle and thus determine the position and size of the bounding box. Through these coordinates, the position and size of the rectangle can be clearly specified for graphics processing and calculations.

Step 3: Calculate the pixel length Lp of Chinese Yam (Fig. 3.5).

Step 4: Establish the relationships between the actual length L_a (cm) and the pixel length L_p by linear fitting (Fig. 3.6):

$$\mathrm{L}_a = 0.1115 L_p - 0.3241$$

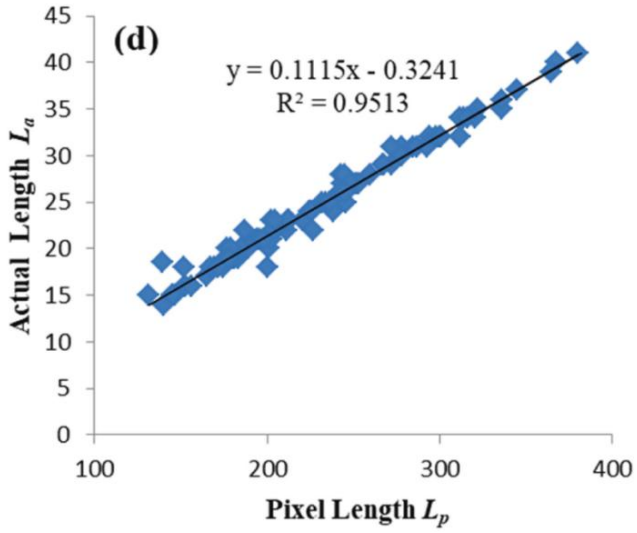


Fig. 3.6 linear fitting of L_a and L_p.

After obtaining the actual length L (cm), normal Chinese Yams can be divided into four size grades: S, M, L and 2L by:

$$\text{Grade} = \begin{cases} \text{S}: L_a \leq 30\text{cm}; \\ \text{M}: 30\text{cm} < L_a \leq 50\text{cm}; \\ \text{L}: 50\text{cm} < L_a \leq 70\text{cm}; \\ \text{2L}: L_a > 70\text{cm}; \end{cases}$$

According to the grading requirements for Chinese Yam sales, the normal shape of Chinese Yam should be natural and uniform. Based on the shape characteristics of Yams, a convex polygon approximation method is proposed, dividing each specification into two levels (Level 1 and Level 2) to maintain consistency and uniformity in appearance (Fig. 3.7). An external convex polygon is

generated to approximate the contour of the Chinese Yams, and shape regularity R_s was defined as (3.6):

$$R_s = \frac{A_c}{A_p}$$

where A_c is the area of Chinese Yam region and A_p is the area of its external convex polygon. R_s is used to measure the regularity of Chinese Yam shape, and its value ranges from 0 to 1. The bigger the R_s, the more regular the Yam is. In addition, it is not affected by image size and Chinese Yam position, and has good adaptability.

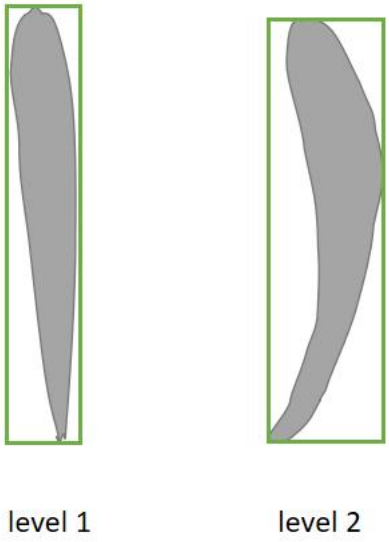


Fig. 3.7 Two specifications divided with convex polygon.

A threshold T can be set to classify Chinese Yam into two levels (Fig.3.8):

$$level = \begin{cases} 1 & R_s \geq T \\ 2 & R_s < T \end{cases}$$

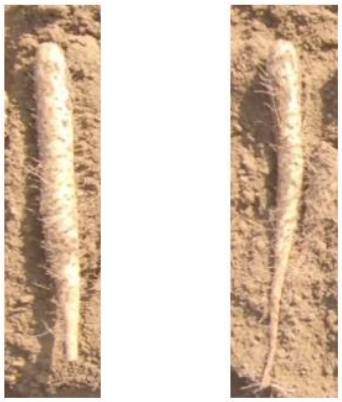


Fig.3.8 Two specifications divided into two levels

3.1.5 Algorithm of defect detection and grading

In this part, it present a detailed algorithm for defect detection and grading of agricultural products, specifically focusing on Chinese Yams. This algorithm, referred to as Algorithm 3.1: Defect Detection and Grading, is a comprehensive process that integrates advanced image analysis techniques with custom-designed classification rules.

The algorithm commences with an input source image, denoted as 'I', of a yam.

The initial step involves setting a decision signal, 's', to -1, which acts as a flag for the processing status. The core of the algorithm is a loop that applies our custom-designed CDDNet model for defect detection on each image 'i' derived from the source image 'I'. The CDDNet model evaluates the image and returns a defect flag, 'dFlag'. If 'dFlag' equals 'defective', the algorithm sets the decision signal 's' to 0, indicating the presence of a defect, and terminates further processing for that particular image.

In cases where no defects are detected, and 's' remains at -1, the algorithm proceeds to the grading phase. This involves calculating the actual length 'La' of the yam using the Minimum Bounding Rectangle (MBR) fitting method. The MBR method provides an efficient way to measure the yam's size based on its pixel dimensions in the image.

Depending on the computed length 'La', the yam is then classified into different levels using a set of predefined size thresholds and the polygon approximation method. For instance, if 'La' is less than or equal to 30, the yam is classified into one of two levels (s=1 or 2). Similarly, yams with lengths within the ranges of 31-50, 51-70, and above 70 are classified into levels s=3,4; s=5,6; and s=7,8, respectively. This classification is based on both the size and shape of the yams, with the polygon approximation method offering a refined analysis of the yam's geometric properties.

The output of the algorithm is the decision signal 's', which provides a comprehensive assessment of each yam, indicating both its quality in terms of defect presence and its size category.

Algorithm 3.1. Defect detection and grading

Algorithm 3.1: Defect detection and grading

Input:Source image I ;

Output:Decision signal s ;

Initialization s= -1 ;

for i do

Defect detection on image i by CDDNet: dFlag=CDDNet(i)

if(dFlag=’defective’)

S=0;

exit for

end if

end for

if(s=-1)

Calculate the actual length La of image I by MBR fitting method;

If(La≤30)then classify I into two levels(s=1,2)by polygon Approximation method;

else if(La≤50)then classify I into two levels(s=3,4);

else if(La≤70)then classify I into two levels(s=5,6);

else classify I into two levels(s=7,8);

end if

```
    end if
    return s;
```

3.1.6 Evaluation standards

In the evaluation of proposed model for defect detection and grading of agricultural products, particularly Chinese Yams, implemented comprehensive standards to assess its performance rigorously. To ensure an objective and multifaceted analysis, five key statistical parameters were calculated: Accuracy, Recall, Specificity, Precision, and F1-Score. These parameters are foundational in the field of machine learning and provide a holistic view of the model's effectiveness.

Accuracy: This parameter measures the overall correctness of the model and is calculated as the ratio of correctly identified instances (both true positives and true negatives) to the total number of instances. It provides a general idea of the model's performance across all classes.

Recall (Sensitivity): Recall assesses the model's ability to correctly identify positive instances. It is the ratio of true positives to the sum of true positives and false negatives. In the context of defect detection, it reflects the model's capability to

correctly identify all defective yams.

Specificity: Specificity measures the model's ability to correctly identify negative instances. It is the ratio of true negatives to the sum of true negatives and false positives. This is crucial for ensuring that non-defective yams are not incorrectly classified as defective.

Precision: Precision indicates the accuracy of positive predictions. It is calculated as the ratio of true positives to the sum of true positives and false positives. A high precision means that a large proportion of yams predicted as defective are indeed defective.

F1-Score: The F1-Score is the harmonic mean of precision and recall. It is a single metric that combines both precision and recall to give a balanced view of the model's overall performance, particularly when the classes are imbalanced.

which are specified by Equations(3.7)(3.8)(3.9)(3.10)(3.11) :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

$$Recall = \frac{TP}{TP + FN} \quad (11)$$

$$Specificity = \frac{TN}{TN + FP} \quad (12)$$

$$\mathrm{P}recision = \frac{TP}{TP + FP} \quad (13)$$

$$\mathrm{F}_1 - Score = \frac{2 \times \mathrm{Pr}\, ecision \times \mathrm{Re}\, call}{\mathrm{Pr}\, ecision + \mathrm{Re}\, call} \quad (14)$$

These parameters were derived from the confusion matrix elements: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These elements represent the correctly and incorrectly classified instances in different categories. To validate model, constructed four distinct image datasets, each designed for different experimental purposes. These datasets were collected using our specialized image acquisition system, ensuring a variety of scenarios and conditions typical in yam processing and grading. The diversity of these datasets is critical for testing the robustness and adaptability of the model under different conditions, ranging from varying lighting and background scenarios to different yam sizes and defect types.

Dataset Name	Categories	Training size	Validation size	Total size
Dataset 1	Normal	1684	726	2439
	Defective	3579	1566	5227
Dataset 2	Normal	700	300	1000
	Fork roots	212	91	303
	Curve	409	75	478
	Fracture	988	422	1396
Dataset 3	S	-	238	238
	M	-	245	245
	L	-	211	211
	2L	-	211	211
Dataset 4	Level 1	-	649	649
	Level 2	-	217	217

Table 3.1 Description of four image datasets.

3.2 Results and Discussion

3.2.1 Effects of model parameters on CDDNet

This part of the research is dedicated to analyzing how various parameters within the CDDNet model influence its overall performance and efficiency, particularly in

the context of defect detection in Chinese Yams. CDDNet, being a deep learning model, consists of numerous parameters that can be tuned to optimize its performance. These parameters include, but are not limited to, the number of layers, the size of the convolutional filters, learning rate, and the number of neurons in each layer. The impact of these parameters on the model's ability to accurately detect defects in yams is critical to the success of the system.

Number of Layers and Filter Size: The depth of the network (number of layers) and the size of the convolutional filters play a crucial role in determining the model's capability to extract and learn features from the input images. A deeper network with more layers can potentially learn more complex features, but it also increases the risk of overfitting and requires more computational resources. Conversely, shallower networks are quicker and require less data to train but might not capture detailed features effectively.

Learning Rate: The learning rate is a key hyperparameter that controls the rate at which the model learns. A higher learning rate allows the model to learn faster, but it can overshoot the optimal solution. A lower learning rate ensures more precise learning but at the cost of increased training time.

Regularization Techniques: Parameters concerning regularization techniques like dropout rates and L2 regularization are also vital. These techniques help prevent

overfitting by penalizing the model for complexity, ensuring that the model generalizes well to new, unseen data.

Batch Size and Epochs: The batch size and the number of epochs the model is trained for are other crucial parameters. The batch size determines the number of samples the model sees before updating its weights, and the number of epochs determines how many times the entire dataset is passed through the network.

In this section, provide a detailed analysis of how these parameters were adjusted and the consequent effects on the model's performance. For instance, we discuss the trade-offs between model complexity and computational efficiency, and how we struck a balance to ensure real-time processing capabilities without sacrificing accuracy. This involved rigorous experimentation, where the model was trained and tested under various parameter configurations, and the results were meticulously recorded and analyzed.The discussion also includes insights into how certain parameter choices led to improvements in specific aspects of the model, such as increased accuracy in defect detection or faster processing times, which are crucial for real-time applications. This comprehensive analysis not only demonstrates the effectiveness of CDDNet in its current configuration but also provides a valuable foundation for future research.

Effect of batch size:

In our research, we conducted a detailed exploration of the impact of batch size on the training dynamics and detection performance of CDDNet, our convolutional neural network model designed for defect detection in Chinese Yams. Batch size, a critical hyperparameter in CNN training, essentially dictates the number of samples processed before the model updates its internal parameters. This size plays a significant role in determining the direction and stability of the gradient descent during optimization, influencing both the training efficiency and model accuracy.For smaller datasets, it's often feasible to use the entire dataset as the batch size, benefiting from a precise gradient calculation at each step. However, with larger datasets, such an approach becomes impractical due to memory constraints. Moreover, excessively large batch sizes can lead to insufficient memory errors, halting the training process. Therefore, selecting an appropriate batch size is crucial for balancing memory usage with the speed and stability of the training process.

In our experiments, we tested various batch sizes (10, 20, 40, 80, and 160) to train and validate CDDNet on dataset 1. Figure 10 in our paper illustrates the relationship between batch size and two key outcomes: validation accuracy and training time, over 10 epochs of training. Validation accuracy, a measure of the model's generalization ability, exhibited a trend where it initially increased with batch size and then decreased, suggesting an optimal point. Notably, the highest accuracy was observed at a batch size of 40.Concurrently, the analysis of training time revealed that it decreased with increasing batch size up to 40, beyond which no significant change was observed. This suggests that larger batch sizes do not necessarily confer a

time advantage, likely due to the overhead of processing larger data chunks per iteration.

Considering these findings, a batch size of 40 was identified as the optimal choice for CDDNet. This size offers a balanced trade-off between the model's generalization ability and training efficiency. By optimizing the batch size, we were able to enhance the training process, ensuring quicker convergence to high accuracy levels, which is pivotal for the practical deployment of the model in real-world agricultural settings.

Effect of the learning rate:

The learning rate is a fundamental hyperparameter in deep learning that plays a pivotal role in the convergence of the objective function during model training. It essentially determines the step size at each iteration while moving toward a minimum of the loss function. A well-chosen learning rate ensures that the model learns efficiently and effectively, striking a balance between the speed of convergence and the risk of overshooting the minimum. In our study, we conducted thorough experiments to identify an optimal learning rate for CDDNet, a model specifically designed for detecting defects in Chinese Yams. For this purpose, CDDNet was trained and validated on dataset 1 using a range of learning rates: 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, and 0.00001. These experiments were critical in

understanding how different learning rates impact the model's ability to generalize, as reflected by its validation accuracy.

The results, as presented in Figure 3.9 of our study, indicate a clear pattern: both excessively high and excessively low learning rates lead to suboptimal validation accuracy. Specifically, a learning rate of 0.00001 was too small, causing the model to learn very slowly and possibly get stuck in local minima, while a rate of 0.05 was too large, leading to erratic and unstable training progress, where the model potentially overshoots the optimal point in the loss landscape. The optimal performance was observed at a learning rate of 0.0005. This rate strikes an effective balance, providing a sufficiently fast convergence rate while maintaining stability in the training process. It allowed the model to steadily decrease the loss and avoid getting trapped in local minima or skipping over the global minimum.

Selecting this appropriate learning rate was crucial for the efficiency and effectiveness of the CDDNet model. It demonstrates the sensitivity of deep learning models to learning rate settings and highlights the necessity of meticulous hyperparameter tuning in developing robust models for practical applications, such as in agricultural defect detection where precision and reliability are of utmost importance.

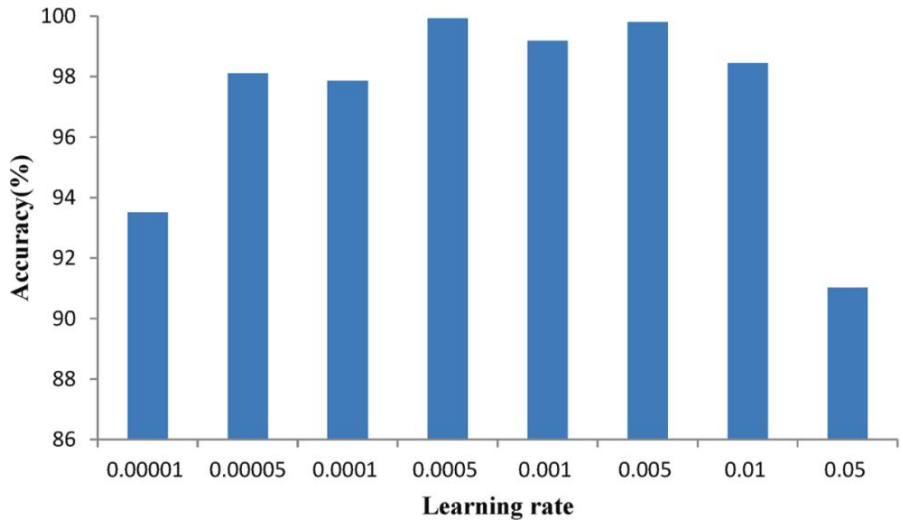


Fig. 3.9 Validation accuracy of different learning rate.

3.2.2 Performance of CDDNet to detect defective Chinese Yams

To rigorously assess the efficacy of CDDNet in detecting defects in Chinese Yams, we conducted a series of experiments using two distinct datasets: Dataset 1, which focused on binary classification (defective or non-defective), and Dataset 2, aimed at multiclass classification (categorizing various types of defects). These experimental trials were essential for evaluating CDDNet's performance in real-world scenarios and for benchmarking it against traditional manual inspection methods employed by workers, as well as comparing it with several prominent deep learning models, including AlexNet, ResNet, MobileNet v2, and ShuffleNet.

In these comparative analyses, the performance of CDDNet and the other deep learning models was quantified using average values of statistical parameters, as defined by specific equations in our study. These parameters included metrics such as

accuracy, precision, recall, F1-score, and specificity, offering a comprehensive view of each model's capabilities in terms of both accuracy and reliability in defect detection. All deep learning models in our study, including CDDNet, were trained over a span of 10 epochs. This duration was chosen to balance between adequate learning and computational efficiency. We employed the cross-entropy loss function and the Adam optimizer for training, with the learning rate set to 0.0005, a value determined from previous experiments to be optimal for our model's performance.

The experimental setup consisted of a computer equipped with an Intel(R) CoreTM i7-6700 processor at 3.20GHz, 16GB RAM, and running on Windows 11 operating system. This hardware configuration ensured sufficient computational power to handle the demands of training complex deep learning models without significant bottlenecks in processing speed. These experiments not only provided insights into CDDNet's performance in comparison to both traditional and modern automated methods but also highlighted the model's potential in revolutionizing defect detection in agricultural products. By comparing CDDNet with both manual inspection methods and other deep learning approaches, we aimed to demonstrate its practical applicability and superiority in various aspects, including accuracy, consistency, and efficiency in defect detection. Such comprehensive evaluation is crucial for advancing agricultural technology and meeting the increasing demand for quality and precision in food production and processing.

Performance evaluation of CDDNet for binary classification:

Table 3.2 in our study provides a comprehensive overview of various performance metrics for different deep learning models, including AlexNet, ResNet50, MobileNet v2, ShuffleNet, and our proposed CDDNet, in the context of binary classification of Chinese Yam defects. These metrics encompass accuracy, recall, specificity, precision, F1-Score, training time, processing time, and model size. Notably, the accuracy of manual defect detection methods by workers was significantly lower compared to these advanced deep learning models. Traditional manual methods typically involve several complex and inflexible steps such as background removal, region of interest detection, and custom feature design for each type of defect. In contrast, deep learning models have the capability to automatically extract features relevant to Yam defects with minimal prior knowledge, offering a more efficient and accurate approach.

Focusing on CDDNet, the model demonstrated exceptional performance, with accuracy, precision, specificity, sensitivity, and F1-Score being 98.94%, 99.02%, 98.93%, 98.76%, and 98.87%, respectively. These figures are only marginally lower than those of ResNet, which had 98.98%, 100%, 98.93%, 98.71%, and 98.92% respectively. However, the most significant advantage of CDDNet over ResNet lies in its training and processing efficiency, as well as its compact model size. CDDNet's training time is only about one-eighteenth of that required by ResNet, and its model size is merely one-thirtieth of ResNet's. When compared to the original ShuffleNet, CDDNet showed a reduction in model size but still maintained high detection

accuracy. These characteristics underline the importance of selecting an appropriate model and network structure tailored for Chinese Yam defect detection.

Furthermore, time efficiency is a critical factor in the grading of Yams. As illustrated in the data, CDDNet not only provides high accuracy in defect detection but also meets the requirements for real-time processing. With a processing time of only 25 milliseconds per image and a substantially reduced model size of 2.1 MB, CDDNet is well-suited for integration into automated systems where rapid, on-the-fly defect detection is essential. This combination of high accuracy, speed, and compactness makes CDDNet an outstanding model for real-time defect detection, significantly outperforming traditional methods and offering a practical solution in agricultural grading and quality control processes.

Methods	AlexNet	ResNet50	MobileNetv2	ShuffleNet	CDDNet
Accuracy(%)	97.98±0.60	98.98±0.90	98.06±0.19	98.89±0.27	98.94±0.11
Recall(%)	98.10±0.76	100	100	99.08±0.06	99.02±0.10
Specificity(%)	97.92±0.97	98.93±0.11	97.56±0.29	98.80±0.32	98.93±0.19
Precision(%)	96.64±1.98	98.71±0.32	95.88±0.59	98.45±0.57	98.76±0.32

F1-Score (%)	97.35± 0.91	98.92± 0.13	97.47±0 .30	98.76± 0.33	98.87± 0.17
Training time(min)	243.2	2033.2	245.7	114.0	107.5
Processing time(ms)	54.4	84.7	48.7	26.4	25.0
Model size(MB)	213.5	85.9	7.8	2.7	2.1

Table 3.2. Experimental results of the defect detection performance of binary classification

3.2.3 Performance evaluation of CDDNet for multiclass classification

Table 3.3 in research provides a detailed comparison between various deep learning models and traditional handcrafted methods in a multi-class classification task, focusing on the detection of different types of defects in Yams. The models compared include well-known deep learning architectures such as AlexNet, ResNet50, MobileNetv2, ShuffleNet, and our proposed model, CDDNet. The defects categorized for this experiment are normal, curve, fork root, and fracture.

Methods	Handcrafted	AlexNet	ResNet50	MobileNetv2	ShuffleNet	CDDNet

Normal(%)	91.12	95.71±3.62	97.03±2.16	98.09±0.8 8	97.98±0.96	97.45±0.95
Curve(%)	93.84	90.31±9.80	95.29±1.76	91.59±2.9 1	92.08±4.80	96.93±2.36
Fork root(%)	95.43	93.17±3.41	90.59±8.41	90.81±8.3 0	95.66±2.88	91.05±7.22
Fracture (%)	81.09	84.25±11.1 4	88.43±3.10	80.89±6.9 1	83.04±7.84	76.87±6.94
Total(%)	90.92	91.19±4.10	94.19±0.60	91.56±1.8 0	92.54±1.46	92.92±1.60
Training time(min)	-	55	446	91	44	43
Processing time(ms)	-	54.4	84.7	48.7	26.4	25.0

*Table 3.*3 *Experimental results of the multiclass detection performance.*

The ResNet model emerged as the top performer in terms of detection accuracy, achieving 94.19%, while CDDNet recorded an accuracy of 92.92%. Although slightly lower in accuracy, CDDNet's most notable advantage lies in its significantly shorter training time compared to ResNet. This balance between time efficiency and accuracy demonstrates CDDNet's overall superiority in the multiclass classification task.An intriguing observation from the experiment is the variance in detection accuracy

across different defect types. The accuracy for detecting normal and curved Yams is relatively high, indicating effective feature recognition by the models for these categories. However, the accuracy significantly drops for fork roots and fractures, with fractures being particularly challenging. This discrepancy can be attributed to two primary factors:

Sample Imbalance Problem: The dataset used in the experiment exhibits a substantial imbalance in sample sizes among different defect categories. For example, there are 138 curve samples but only 29 fibrous root samples. This imbalance can lead to a bias in the classification models, as they tend to perform better on defect types with more abundant data. To mitigate this issue, various strategies can be employed, such as data augmentation, oversampling and undersampling techniques, or manual generation of data samples. In future work, we plan to expand the dataset by collecting more Yam samples or employing data augmentation techniques to enhance the diversity and volume of image samples, thereby improving CDDNet's performance.

Variety of Defect Appearance: Different types of defects, such as mechanical damage, decay, bruises, and wormholes, result in a wide variance in surface appearance. This diversity makes it challenging for CDDNet to extract effective features consistently across all defect types. One potential solution to this issue is to categorize each distinct appearance as a separate category, allowing the model to

specialize in recognizing specific defect characteristics.

Overall, the findings from Table 3.3 highlight the complexities involved in defect detection in agricultural products and underscore the necessity for continuous improvement and adaptation of deep learning models like CDDNet. By addressing these challenges, we aim to enhance the accuracy and reliability of defect detection, thereby contributing to more efficient and effective quality control processes.

3.2.4 Comparison with previous studies

We delve into the complexities of comparing our CDDNet model with other studies in the realm of agricultural defect detection. A notable challenge in this comparative analysis stems from the absence of universally accessible, standardized datasets that can serve as benchmarks within this field. The majority of existing studies on defect detection in agricultural products rely on their unique, proprietary datasets. This practice, while understandable, poses significant obstacles in directly comparing the effectiveness of different models, as each is tailored to specific data characteristics.To ensure a level playing field for comparison, our study adopts a uniform approach in training all evaluated deep learning models, employing consistent model parameters across the board. This strategy includes the use of a pre-trained AlexNet model for experimental purposes, avoiding direct comparisons

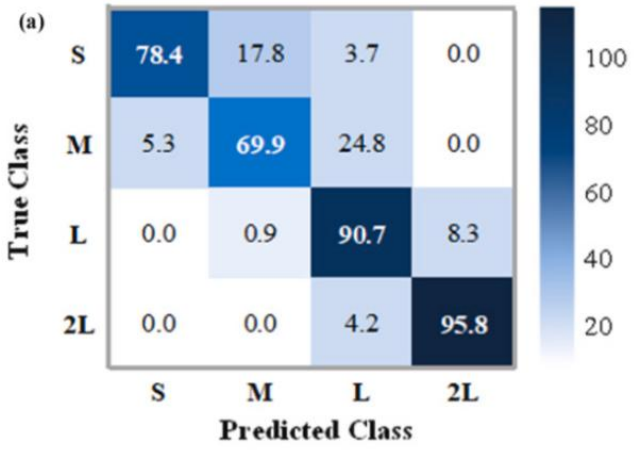
with the experimental results of other studies. Consequently, our research focuses on comparing the performance of CDDNet with our previous work, as well as with some of the most prevalent deep learning methods, namely AlexNet, ResNet, and ShuffleNet, utilizing our own specifically curated dataset.

While acknowledging the limitations of not having a comprehensive comparison with a wide array of studies, it is important to note the specific context of our research. Given the relatively few studies that employ computer vision for defect detection in Chinese Yams, our work presents several distinct advantages. Notably, CDDNet demonstrates robust defect detection capabilities without the need for manually defined features or extensive preprocessing. This attribute suggests that CDDNet can be readily adapted for use with other agricultural products, provided a sufficiently comprehensive dataset is available. Consequently, our model holds promising potential for application in the defect detection of a broad range of root and tuber agricultural products, demonstrating its versatility and broad applicability in the agricultural sector. This aspect of CDDNet underscores its value not only as a tool for Yam inspection but also as a flexible solution adaptable to various agricultural quality control scenarios.

3.2.5 Evaluation of MBR fitting method

We present an in-depth analysis of the effectiveness of our proposed method for

grading Chinese Yams based on size. For this experiment, a total of 419 Chinese Yams, categorized into four distinct size grades (S, M, L, 2L), were selected as experimental samples from a farm in Obihiro, Hokkaido. Utilizing a computer vision system, images of these yams were captured and subsequently graded using the MBR (Minimum Bounding Rectangle) fitting method. The accuracy of the MBR fitting method was thoroughly validated against traditional manual measurements. The manual grading process involved measuring the actual length of each Yam with a ruler, the results of which are depicted in Figure 3.10 of our research. It's important to note that manual methods are subject to significant subjectivity and variability in stability. This inconsistency is evident even when the same individual performs repeated measurements, leading to fluctuations in the results.



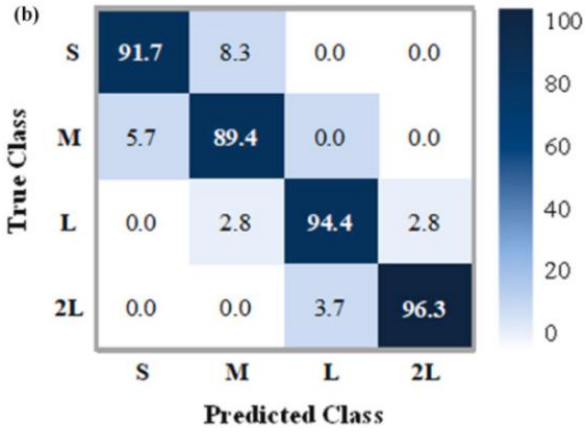


*Fig 3.*10 *Confusion matrix of (a) manual method by workers and (b) proposed method.*

Our experimental findings demonstrated that the overall accuracy achieved by our MBR fitting method was an impressive 92.8%, markedly surpassing the accuracy of the manual method, which stood at 83.1%. When delving into the accuracy across different size grades, it was observed that the manual method's accuracy varied substantially. For smaller sizes (S and M), the accuracy was notably lower at 78.4% and 69.9%, respectively. In contrast, for larger sizes (L and 2L), the accuracy improved significantly, reaching 90.7% and 95.8%. Meanwhile, the MBR fitting method consistently maintained high accuracy across all size grades: 91.7% for S, 89.4% for M, 94.4% for L, and an impressive 96.3% for 2L. These results unequivocally demonstrate the superior reliability and accuracy of our method compared to manual grading.

To the best of our knowledge, there is currently no existing research focusing on the fitting and grading of Chinese Yam size using a method similar to ours. Therefore, in this study, our comparison is primarily limited to manual methods. The drawbacks of manual grading, including its low accuracy and inherent instability, are clearly

highlighted through our analysis. In contrast, our MBR fitting method offers substantial improvements in classification accuracy and reliability. This advancement is not just a step forward in the precision and efficiency of Yam classification but also a significant contribution to the field of agricultural product grading, showcasing the potential of computer vision and machine learning techniques in modern agricultural practices.

3.2.6 Practicability of the proposed approach

The practicability of our proposed approach is demonstrated through the development of a comprehensive defect detection and image recognition system, specifically tailored for Chinese Yam grading. This system integrates the practical size and grade specifications prevalent in the processing of Chinese Yams. It employs CDDNet, our custom-developed deep learning model, to initially screen and identify defective Yams, which may include various defects such as curvature, fork roots, fractures, or flatness. Subsequently, the system utilizes an advanced grading method to classify the non-defective, or normal, Chinese Yams into their respective specifications and grades.This integrated approach marks a significant advancement over previous methodologies. Our system aligns closely with the actual operational processes involved in Yam grading and is designed for ease of application in real-world scenarios. The CDDNet model within this system exhibits a high proficiency in detecting most defects that can affect the appearance quality of Chinese

Yams, thereby enhancing the overall efficacy of the grading process.

Currently, most relevant research in the field predominantly focuses on the detection of surface defects in root crops prior to harvesting. However, reports on post-harvest yam grading using computer vision are relatively scarce. Our method fills this gap, providing a straightforward and practical solution for yam grading. Despite existing grading standards for yams, which often consider factors like shape uniformity, surface smoothness, color uniformity, and maturity, many of these criteria are qualitative and challenging to quantify objectively.

Looking forward, our research aims to delve deeper into establishing more precise and quantifiable quality grading standards for yams. This future work will encompass a broader range of factors, including but not limited to flatness, color uniformity, size, and maturity. By integrating these additional parameters into our grading system, we seek to develop a more holistic and accurate approach to yam quality assessment. This endeavor is not just a technical challenge but also a step towards modernizing agricultural practices, ensuring consistent quality in food products, and meeting the increasing demands for precision in agricultural production and processing.

3.2.7 Conclusion

In conclusion, this chapter highlights the significant research importance and

practical application value of combining deep learning with computer vision for the detection of defects and automatic size grading in Chinese Yams. This innovative approach surpasses the limitations inherent in traditional worker manual methods, utilizing advanced computer vision and deep learning technologies to achieve accurate detection and automated grading of Chinese Yam defects. This methodology not only enhances the efficiency of quality inspection processes but also provides robust support for the estimation of harvest yields of Chinese Yams and other similar root crops.

Throughout this chapter, we have comprehensively discussed various aspects of quality inspection, including the design of the image acquisition system, the development of quality inspection methods, and the establishment of grading standards. A pivotal component of our research is the development of CDDNet, a lightweight network based on ShuffleNet and transfer learning, specifically tailored for defect detection in Chinese Yams. Additionally, we introduced a novel size grading method that employs MBR fitting and convex polygon approximation, demonstrating the versatility of our approach.

The experimental results underscore the effectiveness of our proposed methods. CDDNet achieved impressive detection accuracies of 98.94% in binary classification and 92.92% in multiclass classification. Furthermore, the size grading accuracy using MBR fitting and convex polygon approximation was recorded at 92.8% and 95.1%, respectively. These results not only fulfill the objectives of this study but also validate the feasibility of using deep learning for Chinese Yam defect detection. They pave the

way for implementing similar techniques in the defect detection and size grading of a broader range of root crops.

Overall, the integration of deep learning and computer vision in this domain represents a significant step forward in agricultural technology. It offers a practical and highly effective solution for enhancing the quality control processes in the agricultural sector, promising to revolutionize how we approach crop grading and yield estimation. As we continue to refine and improve these techniques, there is a clear potential for broader applications, extending beyond Chinese Yams to other root corp products, thereby contributing to more efficient, precise, and sustainable agricultural practices.

Chapter 4 Geospatial mapping of yam count and its visualization

The integration of geospatial technology in agricultural practices, particularly in crop analysis and harvesting, has emerged as a transformative approach in modern farming. This chapter focuses on the innovative use of Geospatial Navigation Satellite System (GNSS) technology and Geographic Information Systems (GIS), specifically QGIS, for the mapping and visualization of yam counts. These technologies provide

farmers with invaluable insights, facilitating informed decision-making and optimizing resource allocation for both pre- and post-harvesting activities.

4.1Methodology

4.1.1 Overview of geospatial mapping in agriculture

Geospatial mapping in agriculture refers to the use of geographic information system (GIS) technologies to collect, display, and analyze spatial data related to agricultural activities. This technology is increasingly important in modern farming, as it allows for more precise and efficient practices. Here are some key aspects and applications of geospatial mapping in agriculture:

(1) Land Use Planning and Management: Farmers and agricultural planners use geospatial data to assess land suitability for various crops, understand soil types and topography, and plan land use to optimize crop yields.

(2) Precision Agriculture: Geospatial technologies enable precision agriculture, where farmers can apply inputs like water, fertilizer, and pesticides in exact amounts needed at specific locations in a field. This is made possible through the use of GPS technology, satellite imagery, and field sensors.

(3) Crop Monitoring and Health Assessment: Satellite and drone imagery are used to monitor crop growth, identify stress areas, and detect diseases or pest infestations. This allows for timely intervention to protect crops.

(4) Irrigation Management: Geospatial mapping helps in designing efficient irrigation systems and managing water resources effectively. It can identify areas of a field that require more or less water, aiding in conservation and reducing waste.

(5) Yield Prediction and Harvest Planning: By analyzing historical geospatial data alongside current conditions, farmers can predict crop yields more accurately. This helps in harvest planning and logistics management.

(6) Climate Change Impact Analysis: Geospatial mapping plays a crucial role in understanding the impacts of climate change on agriculture. It helps in modeling future scenarios and developing adaptation strategies.

(7) Supply Chain and Distribution Optimization: Geospatial data aids in optimizing the supply chain for agricultural products. It helps in identifying the best routes for transportation, reducing costs and improving efficiency.

(8) Disaster Management and Mitigation: In the event of natural disasters like floods or droughts, geospatial mapping can help assess damage, plan recovery efforts, and develop mitigation strategies.

(9) Regulatory Compliance and Reporting: Farmers can use geospatial data to ensure compliance with environmental regulations and reporting requirements, such as proving adherence to sustainable farming practices.

(10) Educational and Research Applications: Academics and researchers use geospatial data to study agricultural trends, develop new farming techniques, and educate future generations of farmers.

In summary, geospatial mapping is a crucial tool in modern agriculture,

enhancing productivity, sustainability, and resilience. Its applications range from day-to-day farm management to long-term strategic planning and policy-making.

4.1.2 Importance of precise yam count and mapping for yield estimation

The precise counting and mapping of yam plants are critically important for yield estimation in agriculture, particularly in regions where yams are a staple food and a key agricultural product. Accurate yam count and mapping provide several significant benefits:

(1) Yield Prediction and Planning: Accurate yam counts allow for better prediction of crop yields. This information is vital for farmers to plan for the market supply, manage storage needs, and forecast income. It also assists in making informed decisions about resource allocation for the upcoming planting seasons.

(2) Resource Optimization: Knowing the exact number of yam plants helps in optimizing the use of resources like fertilizers, water, and pesticides. Precise mapping ensures that these resources are applied efficiently, targeting areas that need them most, thus reducing waste and cost.

(3) Disease and Pest Management: With precise mapping, farmers can quickly identify areas where yam plants may be suffering from diseases or pest infestations. This allows for targeted interventions, minimizing the spread of problems and reducing the overall use of pesticides.

(4) Irrigation Management: In areas where irrigation is necessary, knowing the exact location and density of yam plants aids in designing more effective irrigation

systems. This ensures that water is distributed evenly and optimally across the entire crop.

(5) Land Use Efficiency: Precise yam counting and mapping help in understanding the spatial distribution of the crops, which in turn aids in better land use management. This can lead to more efficient planting strategies and higher overall productivity per unit of land.

(6) Data for Research and Improvement: Accurate data on yam counts and their spatial distribution is valuable for agricultural research. It can be used to study plant growth patterns, breed improvement, and the development of more efficient farming techniques.

(7) Market and Supply Chain Management: For stakeholders in the supply chain, precise yield estimation is crucial for planning transportation, storage, and distribution. It helps in managing the supply chain more effectively, reducing losses and improving market supply.

(8) Financial Planning and Credit Access: Farmers with accurate yield estimates can better plan their finances. This information can also be important when seeking credit, as it provides a more reliable estimate of the farmer's potential income.

(9) Policy Making and Food Security: On a larger scale, accurate information about yam yields is important for regional and national agricultural policy-making. It contributes to assessments of food security and can guide investment in agricultural infrastructure and support services.

In summary, the precise counting and mapping of yams are fundamental for

efficient agricultural practices, optimal resource use, effective market planning, and the overall sustainability of yam farming. This precision aids not only individual farmers but also contributes to larger economic and food security strategies.

4.1.3 GNSS in Precision Agricultural

Today, there are two Global Navigation Satellite Systems (GNSS) that are fully operational and commercially available to provide all-weather guidance virtually 24 h a day anywhere on the surface of the earth. GNSS are the collection of localization systems that use satellites to know the location of a user receiver in a global (Earth-centered) coordinate system and this has become the positioning system of choice for precision agriculture technologies. At present North American Positioning System known as Navigation by Satellite Timing and Ranging Global Position System (NAVSTAR GPS or simply GPS) and Russian Positioning System known as Globalnaya Navigatsionnaya Sputnikovaya Sistema or Global Navigation Satellite System (GLONASS) both qualify as GNSS.

The basic principle of operation on which GNSS systems is based is often referred to as resection (also called triangulation), and it involves estimating the distances from at least three satellites orbiting the Earth along different and sufficiently separated trajectories to determine the position of an object in 2-D along

with the uncertainty in measurement. Typically, each GPS satellite continuously transmits at least two carrier waves consisting of two or more codes, and a navigation message. GNSS receivers measure the time it takes for the signal to travel from the transmitter on the satellite to the receptor in the receiver antenna and use that time to calculate the distance (or range) between them. To perform a positioning or navigation task, a GNSS receiver must lock onto the signals from at least three satellites to calculate a two-dimensional (2D) position (latitude and longitude). If four or more satellites are in view, the receiver can determine three-dimensional (3D) position (latitude, longitude, and altitude) of the user.

(1) Applications of GNSS in agriculture

The use of satellite-based localization solutions, particularly GNSS receivers, has become increasingly sophisticated and prevalent in agriculture. These receivers play a crucial role in precision agriculture technologies, where accurate position information is essential for site-specific crop management. However, the level of positioning accuracy required varies depending on the agricultural task. For operations such as yield monitoring, soil sampling, or variable rate applications, submeter accuracy differential GPS (DGPS) is sufficient, as errors below 1 meter are acceptable. In contrast, tasks like mechanical intra-row weed control, thinning of crop plants, precise planting, or autonomous navigation within tight rows require decimeter- or even centimeter-level accuracy. This higher degree of precision can be achieved with real-time kinematic GPS (RTK-GPS).

While there are various global positioning satellite systems available, GPS and GLONASS are currently the only two fully operational GNSS systems. Both systems are similar, but the North American GPS has been in continuous use since the mid-1990s and many agricultural applications have been developed using this system.In recent years, the application of GPS in agriculture has surged, and the literature is replete with many interesting examples. This discussion will focus on six specific applications, which the authors have been closely involved in:

1) Yield monitoring;
2) Compaction profile sensing;
3) Site-specific fumigant application for tree planting;
4) RTK GPS-based plant mapping;
5) Precise weed management system;
6) Robotic applications in agriculture.

(2)Yield monitors

The ability to continuously monitor and map yield during harvest, and to understand its spatial variability, is a crucial aspect of site-specific crop management. This spatial variation in yield data often mirrors differences in soil composition, plant growth, and environmental factors within a field. Yield monitors, extensively used by farmers, consultants, and researchers, have been instrumental in mapping the yields of

various crops. However, the adoption of precision agriculture practices has been most prominent in grains, oilseeds, and cotton.

In terms of technology, cereal grain combines typically employ physical sensors to gauge grain flow, whereas cotton yield monitors utilize microwave or near-infrared sensors to quantify the cotton harvest. A GPS device is integral to the yield monitor system, providing essential position data to pinpoint spatial variations in crop yield. Additionally, other sensors like a forward speed sensor (such as radar, ultrasonic sensor, or magnetic pickup on the transmission drive shaft), crop moisture sensor, and header height sensor are mounted on the combine. The synergy of these sensors and instruments enables the mapping of spatial yield variability, leading to the creation of detailed yield maps. These maps are invaluable in tracking field performance over time and in delineating different management zones for field inputs.In a specific study, soil samples were collected from a depth of 0-20 cm in the spring, on a 20x20 m grid, resulting in 133 samples. The soil was analyzed for various properties, including texture (sand, silt, clay), organic matter (OM), phosphorus (P), and potassium (K). Kriged maps of each soil property and crop yield were created using Surfer software. In this particular case, the main contributor to spatial variability in cotton yield was identified as the variation in soil texture, especially in sand and clay content. This variation significantly affected water content distribution and, consequently, the uniformity of plant growth.

4.1.4 QGIS in mapping agricultural

The development of GIS follows two principal development paradigms: the open source or the closed source development model. In the open-source model, the source code is typically published under a free software license, which grants the user four essential freedoms: the right to run the code for any purpose; to study how the code works and to modify it; to redistribute copies and even redistribute modified copies. Besides the desktop GIS applications, the QGIS project also provides server and related web mapping applications, as well as versions adapted to the requirements of mobile devices.

QGIS is an open-source GIS project, which was started in 2002. The initial goal of the QGIS project was to create a spatial data viewer. Today, QGIS has reached a point in its evolution where it is used by a variety of users for daily spatial data viewing, editing and analysis tasks as well as in education . Plugins are defined as components that can form user interfaces and interact with users to realize algorithms.

QGIS runs on most Linux and Unix platforms, windows and Mac OS X and it has been published under the GNU and General Public License (GPL). The QGIS core is developed using the Qt toolkit and C++. Furthermore, QGIS provides a python interface (API), which is used to expand its functionality. As noted, a powerful python interface can help to efficiently exploit the capabilities of a GIS and integrate different

tools and programming languages to expand it. This effect an increasing number of contributions, has been very noticeable since the introduction of the QGIS python API in QGIS 3.3 when the new developers started to add functionality using python plugins.

4.1.5 Transferring and loading GPS data

In the realm of geospatial analysis, the integration of GPS data into GIS software stands as a crucial process, and Quantum GIS (QGIS) offers a robust platform for such integration. The procedure of transferring and loading GPS data into QGIS involves several key steps, starting with the acquisition of data in standard GPS formats, such as GPX (GPS Exchange Format). This data, often comprising waypoints, tracks, and routes, encapsulates critical spatial information collected in the field. Upon initiating QGIS, this data is imported through the ′Add Vector Layer′ function, found under the 'Layer' menu, facilitating a seamless transition from raw GPS data to a geospatially-referenced format displayed on the QGIS interface. The significance of this process lies not only in the visual representation of the data on a map but also in its readiness for subsequent analytical tasks. QGIS, with its comprehensive toolbox, allows for a range of manipulations and analyses, including but not limited to spatial querying, overlay analysis, and temporal-spatial mapping. This integration thus plays a pivotal role in transforming raw field data into actionable

insights, applicable across a myriad of disciplines such as environmental monitoring, urban planning, and agricultural management. Through this methodology, QGIS effectively bridges the gap between on-field GPS data collection and desktop-based geospatial analysis, underscoring its utility as a powerful tool in the arsenal of researchers and practitioners in the field of geographic information systems, as Figure 4.1. and 4.2

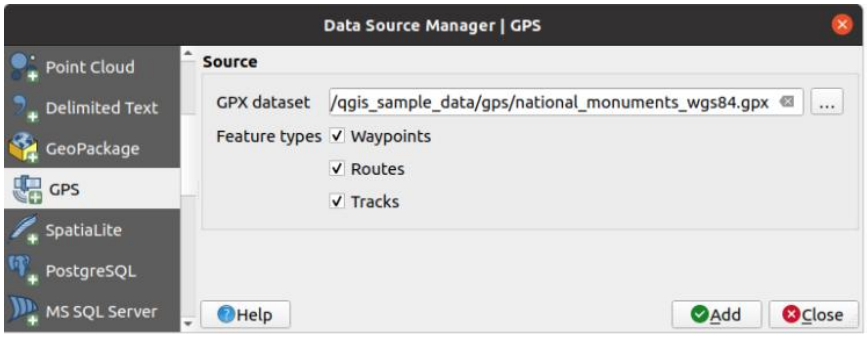


Fig. 4.1 Load GPS data dialog box

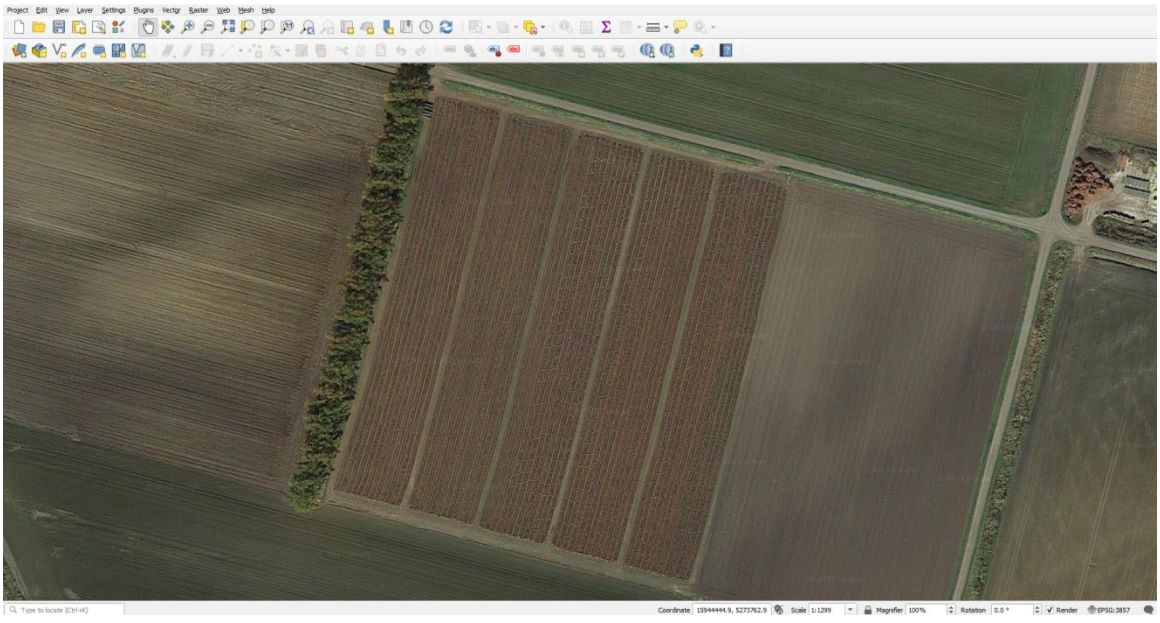


*Fig. 4.**2** QGIS map containing geographical information of collected data*

4.2Results and Discussion

4.2.1 Algorithm for correlating geographical Data with Video Frames

GNSS capturing is synchronized with GoPro video capturing so we are able to match the GNSS coordinates with the video during the counting process. During data gathering, the GoPro consistently records its current coordinate, as Figure 4.3. After video data capturing and analysis are complete, we annotate the counts from the video frames with their respective location on a map. To accomplish this, we record the frequency at which the GPS points are captured and assign a GPS point to a set of frames. In this research, we record 1 GPS point every 15 s (GPS period). Our video is recorded at 30 frames per second, which means 1 GPS point is assigned to every 450 frames (15 * 30). Then formulate an equation for the frames per GPS point in Equation (4.1).

$$frames\_per\_GPS\_point = GPS\_period * frames\_per\_second \quad (4.1)$$

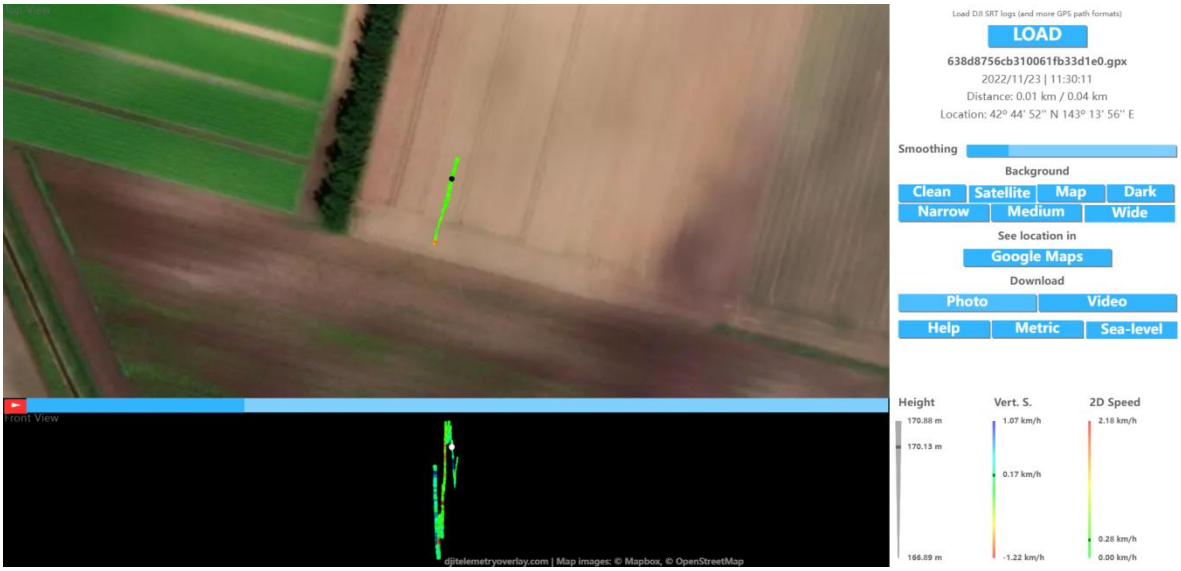


Fig. 4.3 Track recording of GoPro video capture

We then utilize YOLO+Deepsort to pass the line count (Chapter 2) next to the yam counting pipeline, which increments until the maximum number of frames per GPS point is reached. As such, the count is precisely recorded with the exact geolocation. The counter is reinitialized after each GPS annotation and the following GPS point is ready to be assigned.

Lat	Lng	Count
42.71909	143.26337	13
42.71909	143.26337	15
42.71909	143.26337	10
42.71832	143.26251	8

42.71832	143.26251	9
42.71832	143.26251	12
42.71740	143.26109	10
42.71740	143.26109	8
42.71740	143.26109	13

Table 4.1 A sample from the GNSS data in an excel sheet after counting.

The GPS coordinates and their recorded count are used to provide better visualization of the data. (Table 4.1) Research aim is to create a map containing the yield information at a point-by-point basis on a map. To do so, then use the Folium package in Python based on the QGIS and create a map containing GPS points with a tooltip containing the count in an OpenStreetMap template. A runtime visualization is shown in Figure 4.4

Fig. 4.4 Mapping the data after counting.

4.2.2 Statistical Methods for Spatial Data

It is an essential topic in the field of spatial analysis, focusing on understanding the distribution patterns of various phenomena, central to this study are spatial auto correlation techniques like Moran's I and Geary's C. These methods are instrumental in deciphering whether high yam counts are mere random occurrences scattered across a field or if they exhibit a pattern of clustering in specific areas. Moran's I is particularly adept at identifying and measuring the degree of clustering or dispersion, providing insights into the spatial relationships between observed data points. On the other hand, Geary's C offers a contrasting approach, emphasizing the dissimilarities between neighboring data points, As Equation (4.2) Together, these techniques

empower researchers and agronomists to analyze yam counts more comprehensively, leading to a deeper understanding of their spatial distribution.

$$I = \frac{N}{\sum_i \sum_j W_{ij}} \frac{\sum_i \sum_j W_{ij}(X_i - X)(X_j - X)}{\sum_i (X_i - X)^2} \quad (4.2)$$

Where I is is Moran's I, N is the number of spatial units indexed by i and j,W_{ij} is the spatial weight between units i and j, X_i is the value at unit i.

4.2.3 Visualizing the Yield Distribution Map

This Figure 4.5 is a testament to the capabilities of QGIS in transforming raw agricultural data into a comprehensive visual tool for yield analysis. It presents a yield distribution map of Chinese yam crops overlaid on a high-resolution satellite image of the farmland. The map is divided into a grid, where each cell is color-coded to represent varying levels of yield. In this instance, lighter shades indicate areas with lower yields of Chinese yam, providing a clear visual cue to areas that may require more attention or different cultivation strategies. Conversely, the cells with darker hues pinpoint regions of higher productivity, highlighting the most fertile and efficient areas of the farm. This differentiation allows for a nuanced analysis of the land, enabling farmers to identify patterns in root crop performance that correlate with environmental and management variables. The precise GNSS coordinates associated with each cell ensure the accuracy of the data, lending credibility to the subsequent analytics derived from the map. By offering a spatially explicit representation of yield

variability, this map serves as an invaluable resource for decision-making, allowing for targeted interventions that can enhance both the yield and sustainability of the farming practices employed for Chinese yam cultivation.

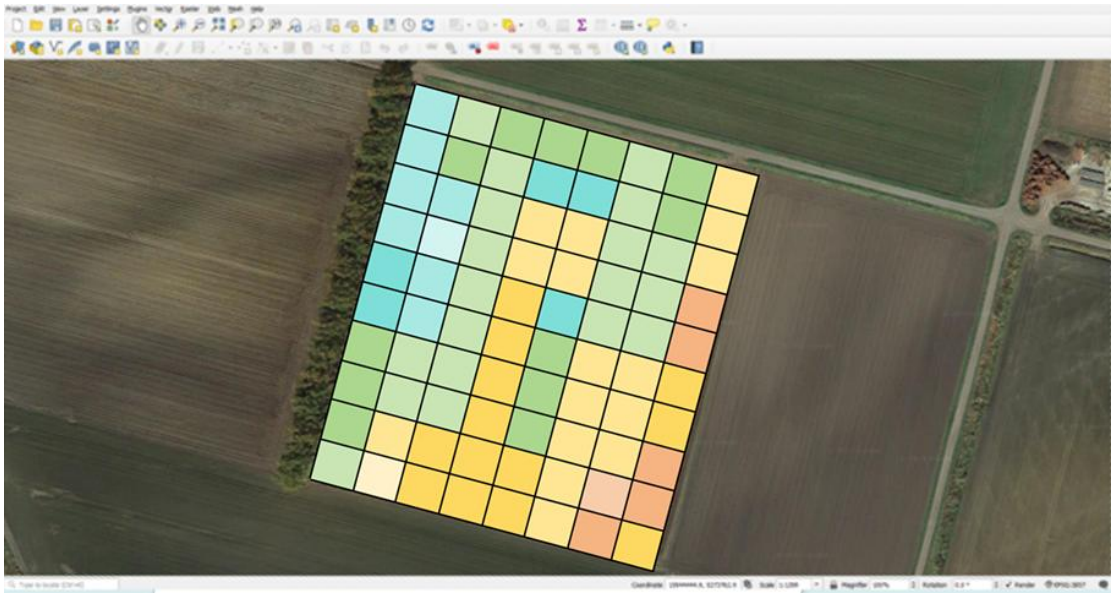


Fig. 4.5 Chinese yam yield map visualization

Chapter 5 Conclusion

In summary, this research introduces a groundbreaking approach in the realm of precision agriculture, specifically tailored for the yield estimation of Chinese Yams (Dioscorea polystachya). As a vital root crop in Japan, the accurate assessment of yam yield is crucial for efficient agricultural planning and management. This study presents an innovative framework that marries the capabilities of computer vision and deep learning, offering a sophisticated yet practical solution for farmers. The primary objective is to provide accurate yield estimates from video footage of harvested yams, with the added benefits of defect detection and size grading.

Key components of the framework include:

Counting Yams with Enhanced Detection: This research has made a significant leap in object detection by integrating the attention module Convolutional Block Attention Module (CBAM) into the YOLOv5s network. This integration has led to a marked improvement in feature extraction capabilities, essential for accurate yam counting. The system is designed to count yams as they pass a virtual detection line in video footage. It employs the Complete Intersection over Union (CIoU) Loss function, which enhances the accuracy of bounding box predictions. Further refinements to the Deep-SORT network reduce the issues of identity switches and target overlap, common challenges in object tracking. This robust system ensures

precise yam counting, which is a fundamental aspect of yield estimation.

Defect Detection and Size Grading: Recognizing the importance of quality assessment, the study introduces the CDD Net, a lightweight deep learning model. This model is ingeniously crafted using Shuffle-Net architecture and transfer learning techniques, enabling it to accurately identify yams with surface and shape defects. The approach takes into account the diverse nature of defects that can occur in yams, ranging from minor blemishes to significant shape deformities. To further enhance the model's accuracy, methodologies involving Minimum Bounding Rectangle (MBR) fitting and convex polygon approximation are employed. These methodologies allow for not only high-accuracy defect classification but also precise size grading of the yams. Such detailed assessment is invaluable for quality control and helps in categorizing yams for various market requirements.

This research represents a significant advancement in agricultural technology, offering a scalable and efficient solution for yam yield estimation. It demonstrates how innovative technology can be effectively harnessed to address specific challenges in agriculture, thereby contributing to the broader goals of sustainable farming and food security.

References

Da Costa, A. Z., Figueroa, H. E. H., & Fracarolli, J. A. Computer vision based detection of external defects on tomatoes using deep learning. Biosystems Engineering.,(2020).https://doi.org/10.1016/j.biosystemseng.2019.12.003

Rong J C, Wang P B, Yang Q, Huang F. A field-tested harvesting robot for oyster mushroom in greenhouse. Agronomy, (2021): 1210. doi: 10.3390/agronomy11061210.

Zhang F, Chen Z J, Wang Y F, Bao R F, Chen X G, Fu S L, et al. Research on flexible end-effectors with humanoid grasp function for small spherical fruit picking. Agriculture, (2023): 123. doi: 10.3390/agriculture13010123.

Nasiri, A., Omid, M., & Taheri-Garavand, A. An automatic sorting system for unwashed eggs using deep learning. Journal of Food Engineering, (2020). https://doi.org/
10.1016/j.jfoodeng.2020.110036

Kala, J. R., Viriri, S., & Tapamo, J. R. An approximation based algorithm for minimum bounding rectangle computation. In IEEE international conference on adaptive science and technology, ICAST, 2015-Janua. (2015). https://doi.org/10.1109/

Rong J C, Wang P B, Wang T J, Ling H, Yuan T. Fruit pose recognition and directional orderly grasping strategies for tomato harvesting robots. Computers and Electronics in Agriculture, (2022); 202: 107430. doi: 10.1016/j.compag.2022.107430.

Afroza A, Ambreen N, Baseerat A; Nigeena N, Ahmad S P, Azrah I S, Amreena S; Insha J, Majid R. Evaluation of Cherry Tomato (Solanum lycopersicum L. var. cerasiforme) Genotypes for Yield and Quality Traits. Journal of Community Mobilization and Sustainable Development, (2021): 72-76.

Vasseghian, Y., Bahadori, A., Khataee, A., Dragoi, E. N., & Moradi, M. Modeling the interfacial tension of water-based binary and ternary systems at high pressures using a neuro-evolutive technique. ACS Omega, (2020). https://doi.org/10.1021/acsomega.9b03518

Yamamoto K, Guo W, Ninomiya S S. Node detection and internode length estimation of tomato seedlings based on image analysis and machine learning. Sensors, (2016): 1044.

Wu, J. S., Zhang, B., & Gao, Y. L. An effective flame segmentation method based on OHTA color space. Applied Mechanics andMaterials,485,7–11.(2012).https://doi.org/10.4028/www.scientific.net/AMR.485.7.

Wang Z L. Underwood J. Walsh KB. Machine vision assessment of mango orchard flowering. Computers and Electronics in Agriculture, (2018); 151: 501–511.

Wu J G, Zhang B H, Zhou J, Xiong Y J, Gu B X, Yang X L. Automatic Recognition of Ripening Tomatoes by Combining Multi-Feature Fusion with a Bi-Layer Classification Strategy for Harvesting Robots. Sensors, (2019): 612- 612.

Xiong J T, Lin R, Liu Z, He Z L, Tang L Y, Yang Z G Zou X J. The recognition of litchi clusters and the calculation of picking point in a nocturnal natural environment. Biosystems Engineering, (2018); 166: 44-57.

Bechar A, Vigneault C. Agricultural robots for field operations: Concepts and components. Biosystems Engineering, (2016); 149: 94-111.

Silwal A, Davidson J R, Karkee M, Mo C K, Zhang Q, Lewis K. Design, integration, and field evaluation of a robotic apple harvester. Journal of Field Robotics, (2017); 34(6: 1140-1159.

Silwal A, Karkee M, Zhang Q. A hierarchical approach to apple identification for robotic harvesting. Transactions of the ASABE, (2016); 59(5): 1079-1086.

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

He K M, Zhang X Y, Ren S Q, Sun J. Deep residual learning for image recognition. (2016)IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2016); 770-778.

Xu Z F, Jia R S, Liu Y B, Zhao C Y, Sun H M. Fast Method of Detecting Tomatoes in a Complex Scene for Picking Robots. IEEE Access, (2020); 8: 55289 - 55299.

Ren S Q, He K M, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, (2017); 39(6): 1137-1149.

Thuyet, D. Q., Kobayashi, Y., & Matsuo, M. A robot system equipped with deep convolutional neural network for autonomous grading and sorting of root-trimmed garlics. Computers and Electronics in Agriculture,(2020). https://doi.org/10.1016/j. compag.2020.105727.

Wang C Y, Liao H M, Wu Y H, Chen P Y, Hsieh J W, Yeh I H. CSPNet: A new backbone that can enhance learning capability of CNN. In: (2020) IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), (2020); pp.1571-1580.

Woo S, Park J C, Lee J, Lweon I. CBAM: Convolutional Block Attention Module. In: Computer Vision - ECCV, (2018); pp.3-19.

Zhong Y, Wang J, Peng J, et al. Anchor Box Optimization for Object Detection[C]. 2 020 IEEE Winter Conference on Applications of Computer Vision (WACV), 20 20: 1275-1283.

Neubeck A, Gool L V. Efficient Non-Maximum Suppression[C]. 18th International C onference on Pattern Recognition (ICPR'06), 2006: 850-855.

Xu Q, Lin R, Yue H, et al. Research on Small Target Detection in Driving Scenarios Based on Improved Yolo Network[J]. IEEE Access, 2020, 8: 27574-27583.

Szegedy C, Wei L, Yangqing J, et al. Going deeper with convolutions[C]. 2015 I EEE

Conference on Computer Vision and Pattern Recognition (CVPR), 2015: 1-9.

Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-ResNet and the i mpact of residual connections on learning[C]. Proceedings of the Thirty-First A AAI Conference on Artificial Intelligence, 2017: 4278–4284.

Jie H, Li S, Gang S, et al. Squeeze-and-Excitation Networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, PP(99).

Zhang K, Zhang L, Yang M-H, et al. Fast Tracking via Spatio-Temporal Context Lear ning[J], 2013.

Henriques J F, Caseiro R, Martins P, et al. High-Speed Tracking with Kernelized Corr elation Filters[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015, 37(3): 583- 596.

Tang J, Cheng Y. Selfish misbehavior detection in 802.11 based wireless networks: A n adaptive approach based on Markov decision process[C]. 2013 Proceedings IE EE INFOCOM, 2013: 1357-1365.