Fruit Ripeness Identification From Digital Images Using Deep Learning

Bingjie Xiao

A thesis submitted to the Auckland University of Technology

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy of Computer and Information Sciences

2023

School of Engineering, Computer & Mathematical Sciences

Abstract

Computer vision serves as a foundational pillar in the domain of digital image processing, permeating diverse applications such as visual object detection, intelligent surveillance, pedestrian detection, autonomous driving, automatic picking, and industrial inspection. It exploits the computational capabilities of computers to automate functions that were traditionally conducted manually, ushering in substantial implications for conserving human resources. Within computer vision, visual object detection emerges as a pivotal element, extensively employed in numerous applications including, but not limited to, face recognition, gait analysis, segmentation, and pedestrian identification.

This thesis delves into deep learning-based visual object detection approaches, which are fundamentally segmented into three categories: Two-stage, one-stage, and transformer-based object detection methods. We have incorporated these methodologies to facilitate the recognition and categorization of fruits, utilizing the transformer-based model to attain a remarkable accuracy rate 99% in fruit classification. Furthermore, our model manifests the capability to execute accurate recognition within a mere 0.12 seconds. The insights derived from this exploration hold potential to augment the efficiency and applicability of computer vision in varied contexts, furthering the advancement of this multifaceted field.

Keywords: Deep learning, Transformer, YOLO, Visual object detection, CenterNet, ConNeXt, Multilayer perceptron

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: Bingjie Xiao Date: 27 September 2023

Acknowledgment

This PhD thesis was completed at the School of Computer and Mathematical Sciences (SCMS) in the Faculty of Design and Creative Technologies (DCT), Auckland University of Technology (AUT) in New Zealand. I would like to deeply thank my parents for the financial support during my entire time of academic study in Auckland, New Zealand.

My deepest thanks are to my supervisors Wei Qi Yan and Minh Nguyen who have provided me with much appreciated technological guidance and support. I believe that I could not be able to achieve my PhD degree without their invaluable help and supervision. In addition, I would like to thank the administrators in our school for their support and guidance through AUT in the past years.

Special thanks to my family and friends, who have assisted me overcoming the barriers of information exchanging, thus have offered me the opportunity to witness the horizon of this world.

Bingjie Xiao

Auckland, New Zealand

October 2023

Chapter 1 Introduction

In this chapter, the applications of visual object detection in the field of fruit recognition will be expounded upon. Simultaneously, the motivation, background, and contribution of the experiment, as well as the structure of this thesis, will be introduced. The core content of this chapter is the elucidation of the profound contributions that the experiments bring to the field. The innovative methodologies and novel approaches employed will be underscored, showcasing their capacity to expand the boundaries of knowledge and contribute to the overarching objectives of fruit recognition research.

1.1 Background and Motivation

Computer vision, as an artificial intelligence discipline, constitutes a critical methodology in modern technology. Deep learning or deep neural networks enable the categorization of semantic entities from visual input and yield pattern classification outputs, thereby demonstrating their remarkable ability to surpass the human visual system. Visual object detection from digital images, a fundamental facet of computer vision, has experienced rapid evolution due to the widespread adoption of deep learning. The primary objective of visual object detection is localization and classification of the objects. This technique can address fundamental computational vision tasks, including but not limited to traffic detection, text detection, gait recognition, pedestrian detection, remote sensing detection, and fruit detection.

The theoretical underpinnings in the field of deep learning are continuously advancing, with architectural innovations experiencing exponential growth. Simultaneously, the domain of deep learning-based digital imaging consistently achieves breakthroughs and innovations. As a prominent subdivision of visual objects, object detection fundamentally addresses the challenges pertaining to spatial localization and classification.

Thus, visual object detection has successfully tackled numerous predicaments in the field of computer vision, including the attainment of precise localization and identification of multiple targets in images. Conventional object detection methodologies heavily rely on sliding window algorithms. The sliding window algorithm adheres to a three-stage process: 1) Region selection, 2) feature extraction, and 3) execution of the classification task. However, conventional approaches tend to generate an excessive number of redundant bounding boxes, thereby burdening the computational complexity. This, in turn, hampers real-time detection capabilities of the model. Thereby, we shall expound upon our prior experiment involving apple detection utilizing the sliding window algorithm.

The primary objective of visual object detection challenge is to identify and precisely position objects within an image, while concurrently assigning them the most suitable categorical labels for precise depiction.

In the domain of conventional agriculture, continuous field monitoring can furnish farmers with pivotal information, thereby ushering in a novel era of agriculture. This involves the automated control of weeds (Young & Pierce, 2013), the management of greenhouses to counteract climate fluctuations through IoT, and the monitoring of crops, soil, and water, among other aspects (Choudhury, Biswas, Prateek, & Chakrabarti, 2021).

Drawing upon the principles of deep learning, computer vision is harnessed to locate visual objects in frame pictures and classify them accordingly. This research effort seeks to adopt deep learning methodologies in the agricultural domain. Our intention is to employ the object detection approach to automate the identification of fruits by machines.

Fruit detection from digital images denotes the capability of our program to locate apples, pears, and other fruits in an image, while simultaneously differentiating their level of maturity. As depicted in Figure 1.1, when the entire fruit image is inputted, the model can autonomously determine the fruit's spatial coordinates and identify its maturity at the given location. The same holds true for fruit video detection, wherein the video is segmented into frame pictures and subsequently recognized.

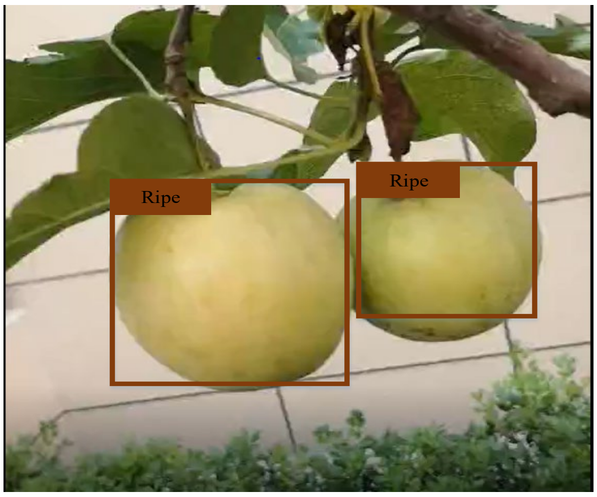


Figure 1.1: A sample of labelled apples in an image

The achievement of fruit detection serves as the fundamental basis for the realization of agricultural automation. In this study, a multitude of methodologies are employed to accomplish fruit object detection, followed by a comparative analysis of the most optimal algorithm through ablation experiments.

1.2 Research Questions

The primary objective of this research seeks to accurately identify and categorize fruits depicted in the provided images, while simultaneously identifying a model capable of precisely localizing fruits in said images and determining their respective ripeness statuses. This involves the ability to differentiate between ripe and overripe fruits.

This thesis seeks to employ deep learning techniques in order to classify the maturity levels of apples. Fruit ripeness experiments are conducted utilizing digital images captured by mobile devices. The apples or pears depicted in these images are subsequently labeled as either "ripe" or "overripe." The central focus of this thesis revolves around the implementation of an effective classification methodology for visual objects. This necessitates the localization of apples or pears in digital images, the extraction of fruit-specific features (predicated upon the number of surface wrinkles exhibited by the fruit), the determination of the degree of decay, and the subsequent classification and labeling of the fruits based on their ripeness ("ripe apple" or "overripe pear").

There exist fruit sorting algorithms in the current literature that have demonstrated commendable speed and precision. However, our study places greater emphasis on the identification and categorization of the quality of targets. The objective of this research is to determine the degree of fruit maturity, specifically in the case of pears and apples, which are classified into three distinct groups: unripe, ripe, and overripe. Therefore, the research question are as follows:

(1) *"How to distinguish whether fruits are ripe in a photograph and how to detect them?"*

(2) *“Which method should be used to detect the fruits?”*

(3) *“How to compare and choose the best method for detection?”*

This thesis was motivated by a news report that shed light on the scarcity of labor during the fruit harvesting season in New Zealand. Given the prevailing labor shortages, it becomes imperative to explore, through experimental means, effective methods for fruit sorting and harvesting. In this study, apples and pears were chosen as representative fruits, with the aim of devising a system capable of classifying fruits based on varying degrees of ripeness. In addition, we consider fruit quality to be a pivotal factor in the automation of harvesting processes.

The primary focus of this thesis revolves around the utilization of computer vision techniques for fruit detection. Drawing upon the features of computer vision in object detection, our experiment initially undertook localization (i.e., determining the precise location of the apple in the image) before proceeding with classification.

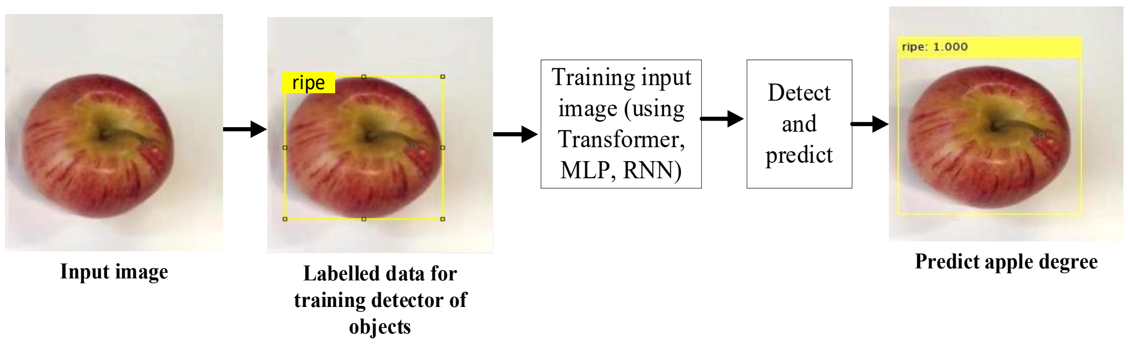

Input image
ripe
Labelled data for training detector of objects
Training input image (using Transformer, MLP, RNN)
Detect and predict
ripe: 1.000
Predict apple degree

Figure 1.2: Apple ripeness classification

The procedure for identifying fruit ripeness is depicted in Figure 1.2. The entirety of the detection process is segmented into the subsequent operations: 1) The position of the target is located in a two-dimensional map, subsequently enabling the prediction of the current fruit's maturity. 2) The identified target is surrounded by a bounding box, thereby facilitating target identification. 3) Finally, for fruit categorization, a deep neural network is employed as a classifier.

In Figure 1.2, the dataset is the one in which bounding boxes are manually labeled, followed by the annotation of maturity categories. The process of manually marking bounding boxes serves as a means of target localization. Annotation categories, on the other hand, constitute a classification process. We train the detector by using the annotated dataset and iteratively adjust the model parameters so as to achieve precise fruit ripeness classification.

1.3 Objectives of This Thesis

The impetus behind this project lies in the utilization of deep neural networks for fruit ripeness classification, employing multiple computational models for this purpose.

This experiment capitalizes on various methodologies to accomplish fruit object detection, subsequently comparing the optimal algorithm through ablation experiments. In practical applications, apples and pears represent the fruit categories necessitating manual harvesting. Fruits at different stages of maturity signify the quality level of the fruit post-harvest.

1.4 Contributions

The classification of various categories of fruits and the assessment of their respective levels of ripeness were successfully executed in this study. The primary objective of this experiment revolved around the application of computer vision for object detection. The Transformer and YOLO models were employed and juxtaposed against the MLP detection model. Within the scope of this thesis, the following accomplishments were achieved:

(1) The generation of alternative datasets based on the manifold attributes of the model, coupled with the adjustment of experimental parameters and the model itself in accordance with the dataset's inherent strengths and weaknesses.

(2) Swin Transformer is employed as the base model, which is suitable for fruit detection. The high-accuracy Swin Transformer is combined with the YOLO

model for rapid fruit detection, the detection model was built by using MLP (i.e., Multilayer Perceptron) which deepens the ability for visual object detection that can achieve better accuracy and faster speed.

(3) The implementation of diverse Transformer models for fruit detection, thereby extracting pivotal information.

(4) The integration of the Transformer models and the YOLO models to simultaneously fulfill the accuracy and speed prerequisites for fruit detection.

(5) The analysis of the accuracy of fruit detection.

Through the utilization of ablation experiments, a comprehensive evaluation of each model's merits and demerits on the fruit dataset was conducted. Notably, our model exhibits exceptional proficiency in detecting diminutive objects when confronted with the challenge of identifying minuscule fruits. Therefore, the model holds significant potential for application to agricultural harvesting.

1.5 Structure of This Thesis

The first part of this thesis provides a succinct overview of the fundamental particulars pertaining to the experiment, such as background, motivation, and object. Subsequently, the second section expounds upon the interrelation between computer vision and object detection, as well as the development of object detection in the field of artificial intelligence. Concurrently, an introduction of diverse detection models is presented, accompanied by an analysis of the insights obtained from previous wo, which serve as a source of inspiration for our own empirical undertakings. In the subsequent segment, a concise account of our previous experimental findings is offered, followed by an assessment of the consistency with the current experiment. Subsequently, the current experimental model, regression function, and the model's inherent advantages are explicated. The following chapter, Chapter 4, is dedicated to the presentation of the results. Chapters 5 and 6 draw the conclusion regarding the present studies, while also proffering avenues for further research. The fruit maturity classification experiment serves as a testament to our capacity to harness artificial intelligence for the purpose of labor recruitment.

1.6 Summary of This Chapter

This chapter thoroughly explored the manifold applications of visual object detection within the specialized domain of fruit recognition. By delving into this subject matter, a comprehensive understanding of the complex nature of this technology, as well as the intricate web of factors that envelop it, is achieved. The rationale behind the incorporation of visual object detection in fruit recognition shall be assessed. To fully grasp the underlying rationale, it is necessary to acknowledge the significance and potential impact of this research in the actual application.

To provide a contextual framework for our study, a comprehensive evaluation of the background is also proffered. This evaluation offers a concise overview of the pivotal breakthroughs that have propelled us to the current zenith of technological advancement, as well as the historical evolution of fruit recognition systems.

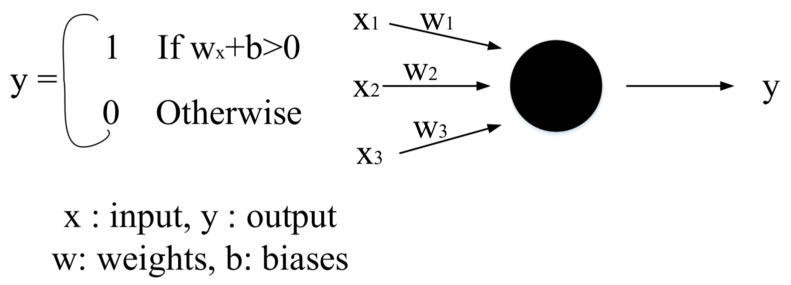
Finally, the organizational structure of this thesis is specially designed for reading conveniently, thereby being equipped with a roadmap for their intellectual journey. Each section and chapter shall be succinctly summarized, facilitating readers' understanding of the logical progression of the study and preparing them for the exhilarating voyage of exploration and revelation that lies ahead.

Chapter 2 Literature Review

In this chapter, a range of visual object detection methods will be introduced concerning their role in facilitating fruit recognition and their connection to previous research. The primary objective is to illuminate a diverse array of detection techniques that form the foundation for advancing the field of fruit recognition. Through this comprehensive exploration, a profound understanding of these fundamental methodologies and their direct applicability will be explained.

2.1 Deep Learning & Machine Learning

The perceptron, a pivotal machine learning model that emerged in 1958, revolutionized intelligent machines. Functioning as a simplistic binary classifier, the perceptron possesses the capability to determine whether an input data point belongs to a specific class. Employing a unit-step activation function, the perceptron yields an output of 1 if the input surpasses 0; otherwise, the output is 0.



$y = \begin{cases} 1 & \text{If } w_x+b>0 \\ 0 & \text{Otherwise} \end{cases}$
x_1
w_1
x_2
w_2
x_3
w_3
y
x : input, y : output
w: weights, b: biases

Figure 2.1: A perceptron

Originally conceived as a machine rather than a model or algorithm, the perceptron is a fully electrical machine comprising 400 photodetectors (or photocells). The weights associated with these photodetectors are stored in potentiometers, while weight updates (occurring during the process of backpropagation), are executed by the motor implement.

The objective of the perceptron is solely to accomplish the recognition of images, limited to the identification of only two categories. Subsequently, the Multilayer Perceptron (MLP) was introduced, showcasing its ability to construct intricate functions by incorporating multiple layers. In contrast to the perceptron, which functions as a single-layer neural network, the MLP possesses the capacity to handle more complex tasks by stacking additional layers.

The Hopfield network, a recurrent neural network that combines memory systems and binary systems, possesses the capability to emulate human memory. Two types of activation functions are employed based on the specific requirements: continuous activation functions are utilized for optimizing calculations, while discrete activation functions are employed to enhance associative memory. However, the Hopfield network is susceptible to the flaw of local minima.

The BP algorithm is well-suited for the backpropagation algorithm of the multilayer perceptron. It expedites the process of propagating errors backwards, in contrast to the conventional forward propagation employed in traditional neural networks. By adjusting the weights and thresholds of the neurons in accordance with predetermined training durations, the backpropagation process effectively reduces output errors. The BP method flawlessly resolves nonlinear classification issues.

The early stages of computer hardware were characterized by the limitations that resulted in inadequate computational power for network models. As the neural network delves deeper, it encounters the predicament of vanishing gradients. In neural networks, the quandary of gradient disappearance is also addressed through the utilization of the BP method. SVM and other shallow machine learning techniques exhibit commendable performance in classification and regression tasks. Nevertheless, shallow machine learning algorithms diverge from deep learning networks, thereby ushering the development of artificial neural networks into a renewed period of stagnation.

By following the demonstration of the efficacy of multilayer perceptron in tackling image recognition challenges, the focus shifted towards the modeling of sequential data (e.g., text).

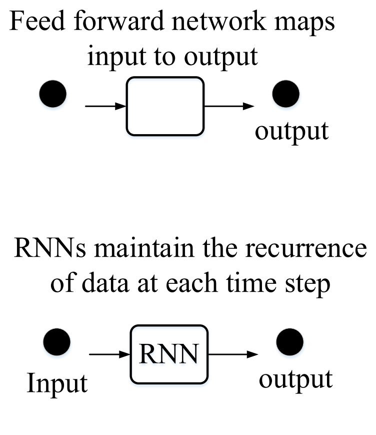

Feed forward network maps input to output
output
RNNs maintain the recurrence of data at each time step
RNN
Input
output

Figure 2.2: Early neural networks

To process sequences, recurrent neural networks were presented. Diverging from feedforward networks such as MLP, RNNs incorporate an internal feedback loop responsible for monitoring the state of information at each time step. The inception of RNN units occurred between 1982 and 1986. However, simple RNN units encounter significant obstacles when applied to lengthy sequences, as they suffer from limited memory capacity and unstable gradients.

In 1998, handwritten character recognition saw the utilization of LeNet-5, which stands as one of the pioneering convolutional network architectures. Comprising three layers, LeNet-5 consists of two convolutional layers, two subsampling or pooling layers, and three fully connected layers. It is worth noting that the convolutional layers are with the activation function sigmoid.

The inherent issue of unstable gradients renders simple RNN units inadequate for handling lengthy sequences. To address this concern, LSTMs, which represent a variant of RNNs, come into play. LSTMs can be perceived as RNN cells taken to the nth degree. These LSTM cells incorporate a gating mechanism that effectively governs the information flow across multiple time steps.

In four ways, the utilization of gates in LSTM architectures serves to regulate the flow of information from the current time step to subsequent time steps.

1) Input gates identify the input sequence.

2) The forget gate eliminates all superfluous data from the input sequence and saves important data to long-term memory.

3) Status value for LTSM cell updates.

4) The output gate regulates the data that must be transmitted to the following time step.

1) Forget 2) Store 3) Update 4) Output

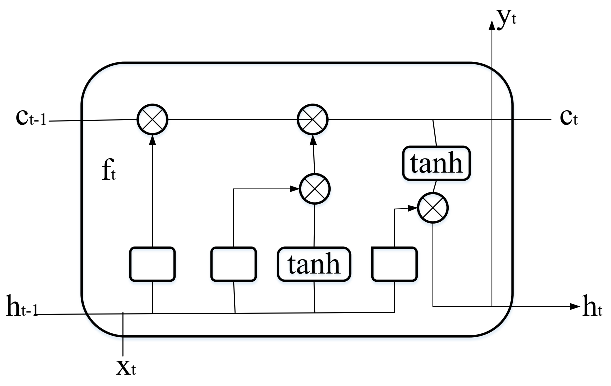


Figure 2.3: Long short-term memory

The design of LSTM neural networks enables the effective handling of lengthy sequences and diverse sequential tasks. The versatility of LSTM extends to involve a wide range of applications, such as sentiment analysis, audio recognition, image caption generation, text categorization, and machine translation.

However, it is worth noting that LSTMs involve a significant computational burden. To address this concern, the Gated Recurrent Unit (GRU) presents itself as a viable alternative, effectively mitigating the computational overhead associated with LSTM. In comparison to the LSTM model, GRU exhibits a reduced parameter count while maintaining satisfactory performance.

The assessment of object classification and image classification architectures on extensive datasets is conducted through the utilization of ImageNet. AlexNet, a prominent architecture, comprises five convolutional layers, succeeded by three fully connected layers, a softmax layer, and a max pooling layer. According to AlexNet, deep convolutional neural networks possess the capability to effectively address image recognition tasks. To enhance the computational efficiency of the model, AlexNet leverages the employment of GPU and the ReLU activation function, which effectively mitigates the issue of gradient disappearance. Remarkably, the introduction of AlexNet in ImageNet resulted in a successful reduction of the error rate from 26% to 15%. ConvNets of larger scale exhibit superior performance. VGG, consisting of 19 layers, outperforms ImageNet with a reduced error rate of 7.3%. The utilization of GoogleNet (Inception-v1) further reduces the error rate to 6.7%. In 2015, Inception v1 was extended with ResNet (i.e., Deep Residual Networks), which contributed to a further reduction in the error rate to 3.6%. This exemplifies the potential of training network models with greater depth (exceeding 100 layers) through the incorporation of residual connections. Prior to the advent of ResNet, training such deep networks was deemed unattainable. The superior performance exhibited by deeper networks has spurred the development of novel architectures such as ResNeXt, DenseNet, Xception, Inception-ResNet, and other similar networks.

Utilizing patterns discovered from previously collected training data, such as music and images, generative networks are employed to generate or manufacture novel data samples. A prominent strategy in this domain is referred to as a Generative Adversarial Network (GAN), which comprises two primary components: A generator responsible for manufacturing counterfeit samples, and a discriminator tasked with distinguishing genuine samples from those generated by the aforementioned generator. The discriminator and generator engage in a state of rivalry and undergo distinct training processes. With each iteration in the training process, the discriminator seeks to enhance its ability to identify between spurious and authentic samples, while the generator continually refines its aptitude for producing counterfeit samples that closely emulate genuine ones.

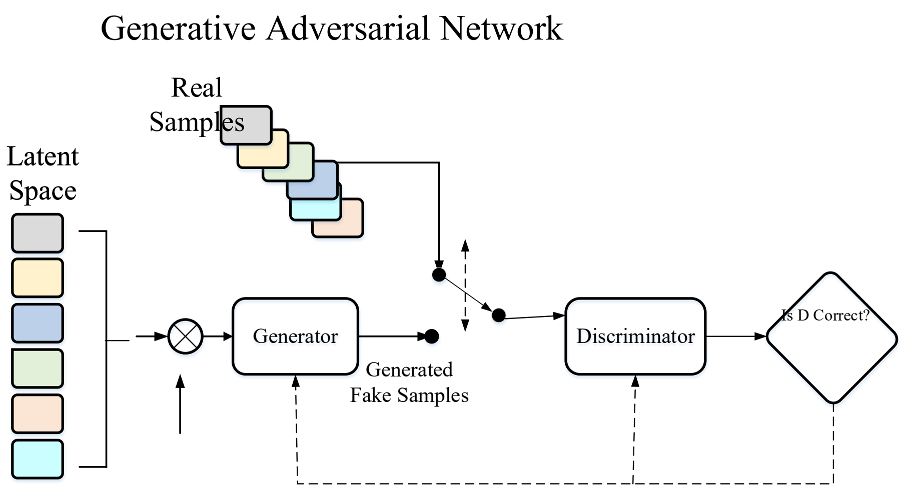


Figure 2.4: Generative adversarial network (GAN)

GANs represent a category of generative models. Other prevalent types of generative models include AutoEncoder, Variation Autoencoder (VAE), and diffusion models.

GANs can be leveraged to generate lifelike images, thereby facilitating the completion of computer vision tasks such as object recognition and segmentation. The Transformer neural network architecture, which is founded upon the attention mechanism, draws inspiration from advancements in Natural Language Processing (NLP). Devoid of recurrent networks or convolutions, The Transformer exemplifies a class of neural network techniques that rely solely on attention-based mechanisms. In order to preserve the sequential order of data, the Transformer incorporates pivotal components, including multi-head attention, residual connections, layer normalization, fully connected layers, and positional encoding.

Multimodal models of vision and language include both visual and verbal constituents, playing a pivotal role in activities such as text-to-image generation (producing an image from a text description), image captioning (making text descriptions for images), and visual question answering (responding to inquiries about the content of an image). The advent of Transformer has paved the way for unified networks with multifarious capabilities, thereby enabling successful forays into vision and language domains.

In the domain of computer vision, pre-training often involves the fine-tuning of a network that has been trained on a vast dataset, such as ImageNet. In natural language processing, it frequently necessitates the fine-tuning of a pre-trained model such as BERT. Prior to the emergence of convolutional neural networks and Transformers, perceptron held sway over the deep learning landscape. Subsequently, ConvNets have exhibited remarkable performance across a plethora of recognition applications, effectively supplanting MLPs. A notable advancement in this regard is the MLP-Mixer architecture, which comprises two primary MLP layers: one that independently operates on image blocks for channel mixing, and another that operates across blocks for spatial mixing.

With regard to image classification, the Vision Transformer (ViT) has demonstrated state-of-the-art performance, albeit encountering challenges in vision downstream tasks such as object detection and segmentation. Swin Transformers have emerged as a viable solution for implementing vision downstream tasks. Concurrently, the Transformer is also employed in conjunction with ResNet to accomplish visual detection tasks.

2.1.1 Visual Object Detection in Computer Vision

Computer vision is a portion of Artificial Intelligence (AI) that facilitates the acquisition of information from image data by artificial systems. This field finds extensive applications in signal science, neurobiology, cognitive science, and other related disciplines, rendering it a comprehensive subject of study.

In the field of computer vision, machines and computers are employed in lieu of human eyes to identify, track, measure, and engage in various activities pertaining to detected objects. Subsequently, computers are utilized to further process the identified objects. Essentially, computer vision represents a computerized emulation of biological vision, wherein the aim is to replicate the visual experiences encountered in everyday life. The computer receives the image as perceived by the human eye and undertakes the task of simulating the human brain's processing of the visual scene. Finally, the computer generates an image that is more amenable to human observation. This entire process is encapsulated in computer vision technology.

Various imaging systems convert the detected object into an image, which is subsequently fed into a computer model in lieu of the human sensory system. These networks simulate the intricate workings of the human brain in order to execute image processing. Concurrently, the primary objective of computer vision research is to endow computers with the capacity to assimilate information in a manner akin to human cognition, thereby enabling the model to autonomously adapt to its surroundings. Just as humans invest substantial time and effort in constructing a visual system, computers follow suit. The construction of a computer recognition model necessitates protracted periods of model training.

Notwithstanding the fact that computer vision simulates human visual processing methods, it cannot entirely supplant human information processing. Presently, learning-based approaches have gained unprecedented prominence in a multitude of specialized computer vision applications. Computers possess pre-programmed solutions tailored to specific tasks.

2.1.2 Supervised Learning & Unsupervised Learning Object Detection

As we all know, unsupervised learning and supervised learning constitute two popular types of machine learning models.

The representative algorithm of unsupervised learning is the clustering algorithm, which signifies that the unsupervised algorithm does not necessitate the acquisition of knowledge by the computer. The clustering algorithm constructs K cluster groups containing N objects. K represents the input parameters and denotes the number of algorithmic groups. Following the determination of the initial segments, iterative calculations are employed to optimize the model. The clustering technique ensures that samples within the same class are comparable, while samples within other classes exhibit distinct characteristics.

Supervised learning employs labeled datasets and leverages computational systems to acquire knowledge from these datasets, finally achieving the function of model prediction. Visual object detection typically relies on a supervised learning model.

2.1.3 Visual Object Detection

With the remarkable advancements in deep learning, visual object detection has also experienced significant progress. To date, visual object detection has successfully addressed the fundamental issue of accurate target localization and classification. Moreover, contemporary object detection models possess the capability to accomplish multiple objectives.

In Figure 2.5, the apple in the given image is surrounded by a bounding box. The positional data of the detected object in the image corresponds to the coordinate information of the four corners of said bounding box. The yellow label denoting "ripe" represents the annotated object class tag. The output of the object detection detector comprises both class labels and positional coordinates.

The traditional methodologies employed in deep learning include the selection of regions of interest, manual extraction of features, and classification by a classifier. Despite the inclusion of multiple steps, these approaches fail to adequately address the diverse characteristics inherent in the learning objectives.

Deep neural networks possess the ability to automatically extract significant features and possess fitting capabilities from vast quantities of data. In deep learning, visual object detection algorithms can be broadly categorized into three types: two-stage object detection, one-stage object detection, and transformer-based object detection.

This thesis undertakes an extensive investigation of object detection algorithms and generates novel concepts for visual object detection based on MLP in the context of deep learning.

The visual object detection pipeline is depicted in Figure 2.1. The dataset is fed into the model for training purposes, subsequently yielding a predicted bounding box. As illustrated in Figure 2.5, the YOLO model leverages the entire image as input, employing a CNN architecture for end-to-end design, effectively providing the position and class label of the bounding box at the output layer.

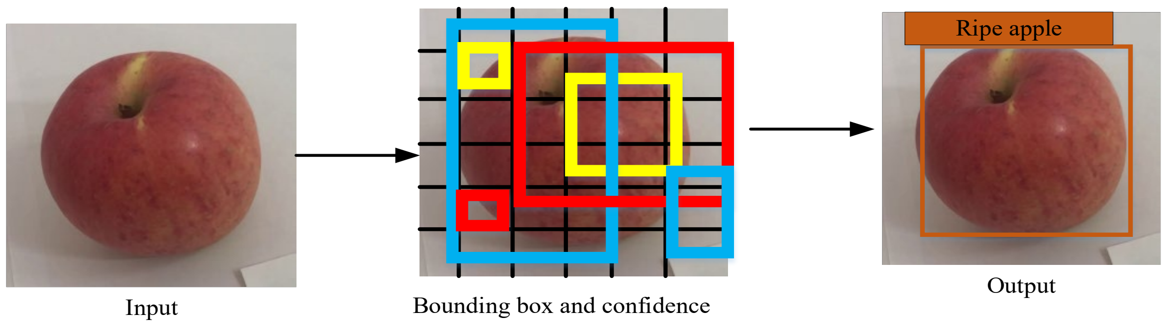


Figure 2.5: The pipeline of YOLO model in visual object detection

2.1.4 Detection, Classification & Identification

Visual object detection refers to the computer's ability to detect the precise location of targets in an image.

Visual object classification and recognition involve in the capacity of deep learning models. Our model demonstrates proficiency in detecting apples and pears, while also detecting their respective levels of ripeness.

Visual object identification pertains to the computer's capability to differentiate between different targets. For instance, our model can accurately determine whether an image depicts a ripe apple or an overripe pear.

Visual object classification algorithms, in the field of machine learning, are employed to make predictions based on the samples in the dataset to which they pertain. In visual object detection, the algorithms aime at locating items in images, it is crucial to differentiate between object detection and object recognition. The former involves the ability to detect multiple objects from a single image. Numerous approaches, predicated upon convolutional neural networks, have been devised to facilitate the detection of targets through visual object detection.

2.2 Fruit Identification

The process of agricultural harvesting is known to be labor-intensive. Therefore, the impetus behind this thesis lies in the utilization of a visual object detection model to classify fruit.

In the era of machine learning, the power of colors and shapes is harnessed as features to identify fruits from images (Zawbaa, Abbass, Hazman, and Hassenian, 2014). Specifically, clustering algorithms were employed for fruit detection, comprising three distinct stages: data preprocessing, feature extraction, and classification. The fruit's image pixels were limited to a mere 90, whereupon the scale-invariant feature transform (SIFT) was employed to collect the color and shape features of the fruit. These shape and color features were subsequently employed as vectors for dataset classification, with the classification process itself being facilitated by the employment of K-NN and SVM algorithms.

Fruit detection from digital videos involves the utilization of computer-based methodologies as opposed to manual operations. Fundamentally, the video is segmented into individual frame images, subsequently employing computer vision models to identify apples and pears in the dataset. However, the crux of fruit ripeness identification lies in a classification + positioning detection task that leverages pre-trained labels and bounding boxes.

The dataset employed for the deep learning object detection quandary comprises ground truth data. This ground-truth data essentially consists of a bounding box + labelled tag. By means of the ground truth, the model extracts the visual characteristics of the fruit and generates predicted box + labelled tag.

The central quandary of detection is defined as follows:

(1) Classification problem: To which class does a given region in an image belong? In our experimental setup, we need to analyze whether the fruit depicted in the image is an apple or a pear.

(2) Localization problem: Where does the target object appear in the image? We need to confirm the presence of fruits in the image and accurately identify their spatial location.

(3) Size problem: What are the dimensions of the target objects in the image? The position and size of the fruit in the image are not fixed.

(4) Shape problem: What is the precise positioning of the target object? The position and size of the fruit in the image are not fixed.

Deep learning in fruit recognition employs mathematical algorithms to identify and categorize fruits based on input two-dimensional images. The initial concepts revolve around fruit recognition predicated on pixel intensities (Jimenez, Ceres & Pons, 2018). In order to locate the fruits, the regions of interest is extracted by using the R-CNN model (Shalini et al., 2021), while segmentation provides broader selection by defining regions of interest in the images (Hameed, Chai, Rassau, 2022).

In the fruit recognition applications, the visual object may occupy a limited portion of the image, or there may be instances of mutual occlusion among the detected targets. For instance, in the collected data, pears and apples are distinctly separated, but there may also exist densely clustered fruits on trees that pose challenges in differentiation. To address the identification of small objects, the Faster R-CNN model leverages the overlap between the ground truth and the anticipated bounding boxes (Behera, Rath & Sethy, 2021). Fruit surface disease detection (Wang et al., 2020) is intrinsically linked to fruit ripeness detection.

In our experiment, we also input fruit images with corresponding labels and bounding box information into the information of the detected target. Subsequently, we train the model to assess the degree of fruit maturity.

The feature information of tomatoes and other fruits is extracted by using the transformer model, which is also utilized for multi-granularity feature extraction through the utilization of patches (Wu, Sun & Hung, 2021). Residual networks are employed to optimize the training parameters. In addition, Jia et al. have made improvements to ResNet, resulting in the development of DenseNet, which achieved an impressive accuracy rate of 97.31% in the identification of apples (Jia et al., 2020).

To address the challenges associated with fruit recognition, the FDR (i.e., Fruit Detection and Recognition) model was introduced (Khan, & Debnath, 2019). The complex natural environment poses difficulties in object detection when it comes to fruit detection applications. In a natural orchard environment, the FDR model proves to be more effective in dealing with the issue of overlapping fruits on fruit trees. By enhancing the neural network's baseline, the FDR model mitigates the impact of background noise on the model. Simultaneously, the FDR model improves the capability of convolutional neural networks to cope with variations related to image resolution, resulting in an impressive accuracy rate 97.83%.

For the purpose of extracting feature information, the Transformer model employs enclosed encoding modules. To achieve fruit detection, a multilevel attention feature extraction module is constructed in the scope of this thesis. In contrast to the conventional CNN approach, the Transformer model demonstrates superior capability in extracting feature information from fruits and recognizing more subtle traits.

Based on the findings of fruit detection, it can be inferred that the Transformer model surpasses various models in terms of accuracy. The research work conducted by Sun et al. (2023) focuses on the application of the Transformer model in fruit detection, aiming to mitigate the influence of the orchard environment. To achieve this, Sun et al. (2023) proposed a modified Transformer model that replaces convolutional layers with focus transformer blocks, which are then integrated into the original model architecture. Unlike the issue of overlapping in orchards, the detection process also needs to account for the distinction between the color of fruits and the background color of trees. Transformer model (Sun et al., 2023) is specifically designed to address the challenge of differentiating the color of green apple peel from the color of leaves. To accomplish this, the Pascal VOC dataset is incorporated with a window-based attention mechanism into the transformer blocks to extract local features of green apples. The achieved accuracy rate is reported to be 34.2%.

By considering the labor-intensive nature of fruit picking and subsequent fruit quality classification (Xia, Nguyen, and Yan, 2022), this thesis proposes an automatic fruit recognition algorithm based on the YOLOv7 and ConvNext models (Tian, 2022). The primary objective is to develop a deep learning model capable of distinguishing between different fruit classes (Bazame, 2021) in the same fruit category, based on the degree of skin folds (Kang & Chen, 2020). The high-precision fruit (e.g., apple and pear) detection and recognition system, which relies on deep learning techniques (Wang & Yan, 2021), can be effectively utilized in everyday life or in natural environments to detect and locate fruit targets (Fu, Nguyen, & Yan, 2022). By utilizing deep learning algorithms, this system enables the detection and recognition of fruit targets through various mediums such as images, videos, and cameras. In addition, it supports the visualization of results and the exportation of inspection results in the form of images or videos (Bhargava & Bansal, 2021).

Visual object detection is characterized by the identification of the location and classification of objects in an image (Liu, Sun, Gu, & Deng, 2022). In the context of a two-dimensional image, the task of target detection involves determining the precise position of an apple in the image and further classifying it as a "ripe apple". Firstly, the dataset undergoes preprocessing, followed by the utilization of a backbone network to extract relevant features. In this process, the ELAN attention mechanism is employed. To ensure the extraction of effective features for fruit recognition, the module operates on the associated channel of the feature map. Subsequently, the model performs feature fusion to obtain semantic information and accurately locate the feature map of the pertinent information. Finally, precise detection results are achieved through category classification and prediction frame regression calculations (Gokhale, Chavan, & Sonawane, 2023).

2.3 Convolutional Neural Networks

Both one-stage and two-stage object detection methods employ supervised learning. Deep learning-based approaches are typically utilized for visual object detection. The one-stage model and the two-stage model are the two primary paradigms in visual object detection. The one-stage object detection model includes YOLO, SSD, and various key point detection algorithms. On the other hand, the two-stage object detection paradigm includes Cascade R-CNN, Fast R-CNN, Faster R-CNN, and FPN (Feature Pyramid Networks).

2.3.1 One-Stage Model

The one-stage algorithm refers to the approach that requires only one round of feature extraction to accomplish detection tasks. Therefore, the object detection speed of a one-stage algorithm surpasses that of a two-stage approach. While the early YOLO model exhibited faster speed compared to the complex Faster R-CNN model, the YOLO model we employed incorporates Transformer to achieve rapid and accurate recognition.

YOLO

YOLO (i.e., You Only Look Once) models accept the entire image as input and directly regress and predict the label with a predicted bounding box at the output layer.

SSD

The SSD (i.e., Single Shot MultiBox Detector) method integrates target localization and classification into a single step, inheriting the YOLO concept of transforming detection into regression. Simultaneously, SSD proposes a similar prior box based on the anchor in Faster R-CNN. SSD incorporates the Pyramidal Feature Hierarchy detection method, enabling the model to anticipate the target on the feature map of distinct receptive fields.

2.3.2 Two-Stage Model

Early fruit detection is accomplished by extracting features from a given image, such as SIFT, HOG, and so on.

The Deformable Parts Model, utilizing the sliding window approach, can also be employed to predict the boundary. However, these early detection models are highly time-consuming and fail to achieve high accuracy.

Subsequently, in comparison to the exhaustive sliding window method, the region proposal-based detection method exhibits a remarkable improvement in model performance while reducing computational overhead. Region proposal serves as a crucial component of the two-stage algorithm.

The region proposal method combines the convolutional neural network (CNN) to derive the R-CNN model, which is commonly referred to as the two-stage model. Figure 2.6 illustrates a simplified two-stage detection model. Faster R-CNN, SPPNet, and other network models are all based on the R-CNN framework. These models demonstrate that region proposal achieves superior precision in visual detection.

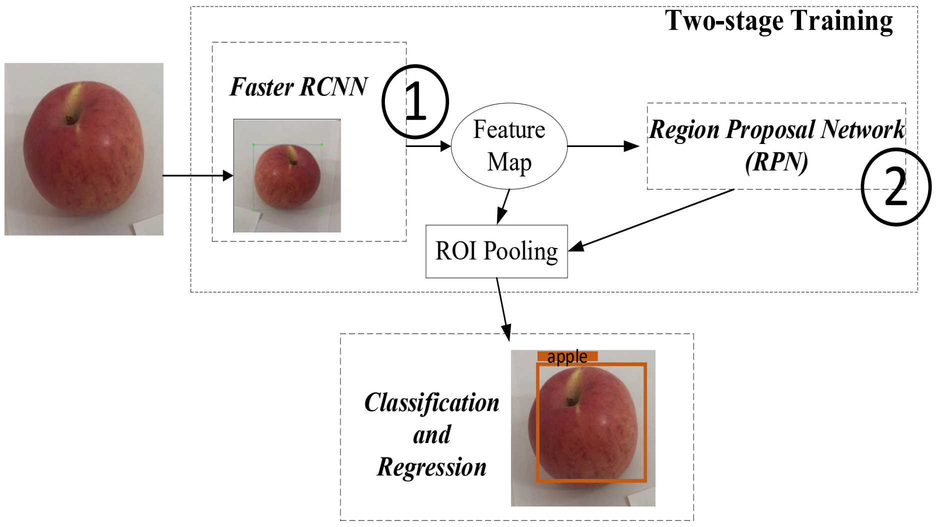


Figure 2.6: A two-stage model

Prior to the advent of the Transformer model, the superiority of the two-stage model in visual object detection was evident. However, the one-stage model has demonstrated its capability to achieve rapid real-time detection (Yao et al., 2022).

A notable characteristic of the two-stage model lies in its utilization of region proposals. Selective search generates region proposals, which are subsequently subjected to regression and classification tasks by the model.

To enhance the detection speed of Faster R-CNN, Wan and Goudos made improvements to the convolutional layer and pooling layer, resulting in an average accuracy 86.41% (Wan & Goudos, 2020).

Transfer learning emerges as another method aimed at enhancing both the accuracy of the model and the speed of model training progress. Positioned in machine learning, transfer learning, supervised learning, semi-supervised learning, and ensemble learning are interconnected concepts. These various learning methods permeate and mutually reinforce one another. Semi-supervised learning algorithms introduce the notion of ensemble learning, while supervised learning can leverage the transfer learning approach.

At its essence, transfer learning entails the extraction of target features from Dataset A, which are subsequently employed for training purposes in Dataset B. The feature information gathered from Dataset A can be effectively utilized in Dataset B, effectively serving as sample weights. Transfer learning facilitates the construction of models with enhanced generalization performance.

The notable advancements has been made to the Faster R-CNN model by incorporating transfer learning techniques and leveraging RGB color as visual features for the purpose of identifying bell peppers (Sa et al., 2016). Transfer learning primarily facilitates the extraction of identifiable characteristics pertaining to the target. The precision and recall rates for detection were reported as 0.807 and 0.838, respectively (Sa et al., 2016).

In prior studies, the Faster R-CNN model was implemented with ResNet-50 during the training phase, yielding a detection precision value of 93% (Xiao, Nguyen & Yan, 2021). In addition, the YOLOv3 model, utilizing DarkNet as the training module, achieved an impressive precision value of 99.96%.

R-CNN

The R-CNN model features unique characteristics, including the extraction of region proposals, component CNN features, and region classification. The extraction of region proposals involves the utilization of the selective search method as a replacement for the conventional sliding window selection approach, thereby contributing to expedited detection. By employing AlexNet as a backbone, CNN can be effectively employed as an image extraction mechanism.

As an object detection methodology, CNN is associated with a relatively sluggish detection speed. Each image typically contains approximately 2,000 bounding boxes, necessitating the application of CNN and feature extraction for each individual bounding box. R-CNN employs SVM as a classifier, necessitating the training of an SVM for each distinct class. Additionally, the R-CNN model necessitates the training of the final detection frame regressor. The abundance of bounding boxes results in a substantial computational burden and a laborious training process, ultimately impeding the detection speed.

Fast R-CNN

In contrast to R-CNN, Fast R-CNN incorporates the entire image into the CNN model to extract features. Followed the methods of feature extraction, the model transmits the region of interest, acquired through selective search, to the network for training. The region of interest corresponds to the region proposal in R-CNN. Therefore, Faster R-CNN necessitates only a single instance of feature map extraction, thereby reducing the computational load.

Each region of interest involves a bounding box of varying sizes. The R-CNN model resizes the bounding box and subsequently extracts features. The RoI pooling layer in Fast R-CNN ensures that the feature maps corresponding to each bounding box possess identical dimensions. Concurrently, SoftMax replaces the SVM module of R-CNN and incorporates a regressor for training purposes. Fast R-CNN attains higher accuracy while reducing computational requirements.

RoI Polling involves the mapping of a region of interest onto a feature map. The region of interest is divided into $n \times n$ segments (output dimension), with each segment undergoing max pooling operations. Finally, an $n \times n$ output matrix is derived.

Faster R-CNN

Faster R-CNN incorporates a Region Proposal Network (RPN) into Fast R-CNN, thereby extracting candidate boxes. The selective search method in Fast R-CNN is replaced by the RPN network for bounding box extraction, resulting in enhanced accuracy and computational speed. The introduction of anchors in the RPN network in Faster R-CNN reduces the computational complexity. In the Faster R-CNN framework, the RPN network is trained separately, and the training process is complicated, rendering Faster R-CNN significantly faster than Fast R-CNN. However, its implementation in industrial settings remains challenging due to computational speed constraints.

A novel visual object detection approach was proposed to address the fruit counting problem. The SSD model was employed in conjunction with MobileNet, while Faster R-CNN with Inception v2 was utilized for multi-fruit object tracking based on Gaussian estimation. Vasconez et al. (2020) achieved accuracy rate 90% by using the SSD model and accuracy rate 93% by using Faster R-CNN.

2.4 Object Detection Model

2.4.1 Transformer

The Transformer model is attributed to the attention mechanism, which was originally introduced to enhance sequence-to-sequence tasks. This mechanism empowers the model to concentrate its attention on different regions of the input data, thereby facilitating output generation. Essentially, it enables the network to "focus" on input components based on their significance. In the context of NLP tasks involving lengthy sequences, the attention mechanism significantly enhances performance by capturing long-range dependencies more effectively than RNNs.

The Transformer architecture, which completely forgoes recurrent layers and relies solely on attention mechanisms, particularly the innovative variant termed "multi-head self-attention," represents a significant advancement in the field. This architectural choice enables the Transformer to achieve a high degree of parallelizability, resulting in enhanced training speed. Additionally, the Transformer introduces the concept of positional encodings, which enables the model to consider the positional information of words in a sequence. This capability is particularly valuable given the permutationinvariant nature of the architecture. Therefore, the Transformer swiftly emerged as the new standard in various NLP benchmarks.

In 2018, Google unveiled the Bidirectional Encoder Representation from Transformers (BERT), which propelled the Transformer architecture to new heights. BERT revolutionized the field by employing pre-training on a vast corpus using a large-scale Transformer model, followed by fine-tuning for specific tasks. Subsequently, numerous variants and models based on the Transformer architecture, e.g., GPT (Generative Pre-trained Transformer), RoBERTa, T5 (Text-to-Text Transfer Transformer), and XLNet, were proposed. These models have established new benchmarks across diverse NLP tasks.

While originally designed for NLP applications, the Transformer architecture has been successfully adapted for use in other domains, including computer vision (e.g., Vision Transformer) and even protein structure prediction (e.g., DeepMind's AlphaFold).

By considering the widespread adoption of the Transformer, the increasing focus is on optimizing its performance to address challenges related to computational cost, model size, and training efficiency. This has led to the development of techniques such as knowledge distillation, pruning, and quantization, which aim to create smaller Transformer models suitable for deployment on edge devices.

To address the challenges posed by longer sequences and to optimize memory requirements, the introduction of architectural changes, such as sparse attention patterns and reversible layers, has been proposed.

Despite of the established efficacy, Transformers encounter certain obstacles. Due to the extensive parameterization, they necessitate substantial computational resources. Particularly for exceedingly lengthy sequences, the quadratic complexity associated with self-attention in relation to sequence length can present a constraining factor.

Transformer models have fundamentally revolutionized the landscape of deep learning, propelling remarkable advancements across a diverse array of applications.

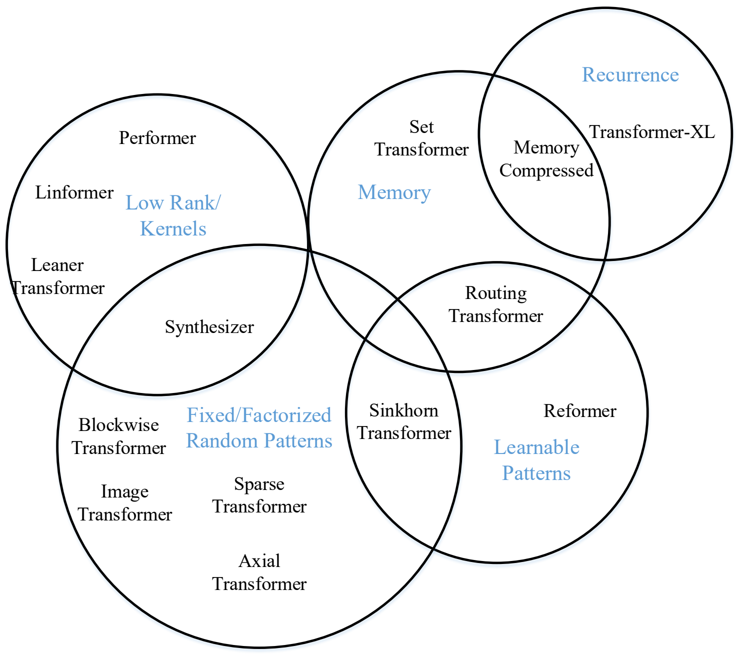


Figure 2.7: A type of Transformer models

The fundamental objective of agricultural automation resides in the utilization of computer models to govern machinery, thereby achieving automated harvesting. In agriculture, various tasks akin to fruit detection exist, such as the identification of rice diseases. A hierarchical design of the Swin Transformer (Zhang et al., 2021) has been proposed through utilizing a sliding window approach, which yielded a detection accuracy of 93.4% for rice diseases.

In addition, Transformers can be effectively employed to facilitate the implementation of a grabber mechanism (Han et al., 2021). The Transformer architecture is harnessed to extract both visual and tactile information, leveraging predefined object characteristics to enable object grasping. Additionally, a comprehensive analysis is conducted with the comparison between the CNN+LSTM model and the Transformer model.

Pertaining to visual object detection, CNN and the Transformer model possess distinct advantages and disadvantages (Arkin et. al., 2021). The CNN model exhibits a characteristic progression from local to global, whereby the convolutional layer progressively extracts information from local regions and subsequently integrates it to derive global feature representations. Conversely, the Transformer model directly captures global information and employs self-attention mechanisms in each patch of the Transformer to facilitate information transfer. It is worth noting that the Transformer model lacks bionic capabilities. Under equivalent parameter settings, the Transformer model necessitates a sufficient number of data samples to attain identical performance to that of CNN. Nevertheless, the Transformer model surpasses R-CNN in its ability to transmit features effectively, thereby reducing computational requirements and enhancing model portability.

Vision Transformer (ViT)

Throughout the stacking of convolutional layers, the CNN model gradually synthesizes global information by iteratively extracting information from local regions. Conversely, the Transformer model incorporates dependencies and directly retrieves global information.

Vision Transformer shares similarities with the CNN model, whereby an increase in network depth corresponds to an increased average attention span (Xiang et al., 2021). The absence of bionic capabilities in the Transformer model necessitates an abundance of training dataset to achieve optimal training results for the Vision Transformer. Fortunately, the Transformer model's robust scaling capabilities ensure superior feature propagation.

The detection of plant disease problems can be facilitated by employing the ViT model in conjunction with DenseNet-169 and ResNet-50v2 architectures (Alzahrani & Alsaade, 2023). Notably, the enhanced ViT model exhibits an impressive accuracy of 99.88%. In the case of multi-classification problems, the utilization of the sparse classification cross-entropy loss function enables the implementation of the detection task.

Detection Transformer (DETR)

The enhancements have been introduced to the Detection Transformer (DETR), with a primary focus on enhancing object detection for large objects (Dai et al., 2021). In this context, DETR is employed to perform object localization and preference determination while preserving the R-CNN base architecture. To facilitate multi-query localizations, this enhancement incorporates the features such as object query shuffle and attention masks (Zhang et al., 2021).

Swin Transformer

Swin Transformer model constructs a detection task by downsampling the image in a manner reminiscent of the hierarchical feature map of a convolutional neural network. Specifically, the Swin Transformer downsamples the feature map, typically by factors of 4, 8, and 16, utilizing the Windows multi-head attention method. When downsampling by a factor of four, the Swin Transformer model divides the feature map into multiple disjoint sections, with each attention mechanism exclusively employed in its designated window. In comparison to alternative models, the Transformer model effectively reduces computational complexity when dealing with large feature maps.

The local perception network of Swin Transformer is constructed by the Transformer module and the CNN module. The Spatial Attention Interleaved Execution Cascade (SAIEC) network is enhanced (Xu et al., 2021) for small object detection, resulting in a precision that surpasses the basic Swin Transformer model by 1.7%.

The multi-perceptual architecture of Transformer model outperforms residual networks in facilitating cross-channel feature transfer for visual objects (Touvron et al., 2022).

In contrast to CNN or Mask R-CNN models, Transformer models do not necessitate global modifications (e.g., convolutional, max-pooling, or global average pooling) while data augmentation or distillation is readily available (Ganesh et al., 2019).

To enhance the Swin Transformer model, one can leverage the encoder, decoder, and skip connection components (Abozeid et al., 2022). Additionally, visual object computation, stacking, and feature map upsampling using patches can be employed to mitigate the adverse effects of noise in natural environments on detection (Wang et al., 2018).

In object detection tasks, Swin Transformer is also combined with CNN models, such as weighted box fusion (WBF), non-maximum weighting (NMW), and non-maximum suppression (NMS) (Hendria et al., 2021).

2.4.2 Attention Mechanism

Visual object detection, a form of image retrieval, emulates human vision. The attention mechanism, akin to the human visual system, identifies targets of interest in complex scenes. Fundamentally, it employs semantic analysis to locate the required targets. While the human brain adeptly processes complex high-level semantic information, computer models are inadequate in achieving excessively complex image retrieval.

Currently, deep learning models predominantly rely on extracting texture, color, and other features of detected targets to accomplish classification. However, in comparison to the sophisticated and complex semantic information processed by humans, computer models still possess ample room for advancement. Transformer model mimics the human visual perception system and extracts the necessary detailed features by establishing a top-down attention mechanism. The attention mechanism of the Transformer combines local and global features, retrieves detected targets from a hierarchical perspective, and establishes a feature extraction strategy akin to human vision.

2.4.3 NLP & MLP

NLP emerges as a prominent research domain in artificial intelligence. Semantic analysis, text generation, chatbot construction, translation, and text-to-speech conversion all fall in the purview of NLP (Tang et al., 2021). Extensive datasets typically exhibit complex structures and detailed features. The multilayer perceptron model accomplishes the processing of extensive datasets through deep learning neural networks, trained on abstract data. Naturally, current vision tasks present abundant opportunities for improvement (Ünal et al., 2020).

The LSTM, GRU, and RNN series models are all based on complex neural networks and serve as prototypical NLP models. It is necessary to note that the complexity of deep neural networks does not inherently confer superior aptitude in identifying object features. However, if YOLO is combined with the LSTM model for detection tasks, a marked enhancement in precision results ensues. In addition, the issue of slow detection speed, stemming from NLP models and based on deep networks, can be ameliorated (Alkalouti & Masre, 2021).

Among the earliest and most rudimentary neural networks, to address more complicated visual tasks, the conventional artificial neural network architecture has undergone a metamorphosis, transitioning from MLP to a hybrid of CNN and RNN. The Transformer, which capitalizes on the attention mechanism, stands as a significant representative of the progress and application of NLP.

Akin to the local feature extraction function inherent in the CNN architecture, Lian et al. have adopted an enhanced AS-MLP model to encode global features. The AS-MLP model leverages the axial displacement of the feature map to construct comprehensive spatial features (Lian et al., 2022).

In previous work, an enhanced Vision Transformer model (Tolstikhin et al., 2021) is employed that is premised on MLP architecture. The Transformer is applied to segment the image into multiple non-overlapping patches. Subsequently, each patch was converted into a mixer layer in the feature embedding model. The MLP-Mixer model effectively replaces the modules in the Vision Transformer model.

2.4.4 Mask R-CNN Model Research

Regarding object detection, instance segmentation, and key point detection, Mask R-CNN emerges as a valuable approach for instance segmentation. Operating within a two-stage framework, the initial stage examines the image to generate proposals, which represent regions that potentially contain objects. The subsequent stage undertakes the categorization of these proposals while simultaneously generating bounding boxes and masks. Mask R-CNN represents an extension of the renowned Faster R-CNN object detection framework, thereby expanding its capabilities into an instance segmentation. Notably, Mask R-CNN exhibits a remarkable level of adaptability, enabling the integration of diverse branches to accommodate a wide array of tasks.

The task of detecting objects using Mask R-CNN involves a preliminary data preprocessing step, followed by the input of the processed images into a pre-trained neural network (e.g., ResNeXt) which generates a feature map. Multiple regions of interest are then generated based on this feature map. The regions of interest that do not contain the target object are filtered out by the RPN network, which performs binary classification and bounding box regression operations on the remaining regions. To classify the regions of interest, the RoIAlign procedure correlates the input image with the pixels of the feature map and aligns the feature map with a fixed feature. The FCN module of region of interest facilitates bounding box regression and Mask generation.

Similar to Faster R-CNN, Mask R-CNN is employed in conjunction with various neural networks. For instance, the Mask R-CNN model can be utilized with ResNet-50 to extract RPN for detecting wheat diseases and generating anchor points (Kumar & Kukreja, 2022). The RPN produces a binary mask for each detected object. In the context of Mask R-CNN, RPN employs the anchor box as a reference for aligning the features of the region of interest (Chen, Hsieh, & Kong, 2022). During the actual detection process, the extraction of anchor boxes may result in losses in both bounding box and mask accuracy.

To address the impact of the environment on tomato detection, the utilization of mask R-CNN has been employed (Wang et al., 2023). Demonstrating an impressive accuracy rate of 89.4%, Wang et al.'s Mask R-CNN detection model, which is founded upon the Swin Transformer model, exhibits the capability to identify the dimensions and categorization of tomatoes.

2.4.5 CenterNet & ConvNeXt

The hallmark feature of CenterNet lies in its utilization of the center point for detection. CenterNet represents a two-dimensional image of the detected object as a singular point.

ConvNeXt, derived from ResNet, undergoes a transformation process akin to the construction of the Transformer model. ConvNeXt maintains the simplicity inherent in CNN neural networks while adhering to the structural framework of the Swin Transformer model.

The CenterNet model is commonly integrated with ResNet to accomplish visual detection tasks (Jaju & Chandak, 2022). The residual network possesses the characteristics as the number of network layers increases, the capacity of model generalization improves.

ResNet is incorporated into the backbone of CenterNet, with its loss function aiding the model in achieving an accuracy rate 78.6% (Zhao & Yan, 2021). As the name implies, CenterNet focuses on feature point detection. In current two-dimensional object detection, CenterNet effectively employs the feature grid map of the target's center point. However, in the context of three-dimensional object detection, the imprecision of center point may result in the blurring of the target's direction and size, thereby leading to inaccurate detection and misinterpretation of global information. Naturally, the enhanced 3D-CenterNet can manipulate the model parameters, estimate the position of the center point through the bounding box, and subsequently capture the characteristics of the detected target (Wang et al., 2021).

2.4.6 YOLO Models

The YOLO model, a one-stage deep learning regression method, is characterized by its omission of the region proposal process. Instead of directly estimating the coordinates of the bounding box, it introduces a grid and anchor offset-based approach. In contrast to the computationally intensive Faster R-CNN, the YOLOv3 (Kuznetsova, Maleva &

Soloviev, 2020) and YOLOv5 (Tsai & Tseng, 2021) models have already demonstrated their potential in the development of fruit picking robots. Notably, the YOLOv3 model exhibited omission rate 9.2% for apples in the dataset (Kousik et al., 2021), while the YOLOv5 model failed to detect 2.8% of the apples (Mhala, Chateau & Amara, 2019).

The offset of the anchor point, determined by the fixed grid, represents the positional difference between the detected object and the anchor point. The anchor-based offset implies that the position of grids remains fixed, and the offset corresponds to the disparity between the object position and grid. YOLOv2 employs k-means clustering to identify anchors and predict the relative position of the grid.

The YOLO model has been employed for the purpose of visual object detection through the division of cells, with the number of divisions varying. Building upon the foundations laid by YOLOv2, YOLOv3 introduces the DarkNet-53 classifier to enhance the implementation of the fundamental classification network, akin to ResNet. The original single-label classification has been refined and promoted to multilabel classification. In the YOLOv3 model, each grid unit has the capability to predict three boxes, with each box containing five essential parameters $(x, y, w, h, confidence)$ associated with 80 classes. Instead of utilizing $softmax$, YOLOv3 employs multiple logistic classifiers to classify each box, thereby facilitating the attainment of multi-label classification and an enhanced detection accuracy.

In order to address the class prediction aspect, a binary cross-entropy loss based on YOLOv3 was proposed, while scale prediction was employed to enhance the precision value pertaining to the detection of small objects (Redmon and Farhadi, 2018). The understanding of YOLOv3 was predicated based on the limitations imposed by the device's memory and computing power (Mao, Sun, Liu, and Jia, 2019), leading to the redesign of the lightweight YOLOv3 network based on Darknet-53. Pointwise group convolutions and separable convolutions were harnessed to reduce the parameter size of the network. However, it should be noted that even a minor change to the network can result in a decrease in precision. To facilitate more efficient parameter training, a Ushaped structure of a multi-scale feature network was incorporated. The size of the minimal network parameters amounts to only one-sixteenth of Darknet-53, thereby enabling faster detection times.

The analysis of YOLOv3 was conducted in 2019 (Benjdira et al., 2019). In contrast to the exclusive tags utilized in previous iterations, YOLOv3 employs a multilabel classification approach. To determine potential objects with specific labels, logical classifiers such as Darknet-19 are employed. In terms of classification, the binary cross-entropy loss function was proposed. YOLOv3 utilizes various bounding boxes to assign an anchor point to each ground truth object. Additionally, YOLOv3 leverages the feature pyramid network to make predictions across different scales. A novel CNN feature extractor, inspired by ResNet, is combined with the skip connection network known as Darknet-53. Notably, it exhibits a higher level of precision compared to ResNet-152, while requiring fewer floating-point operations.

Tinier-YOLO structure based on Tiny-YOLO-V3 was proposed (Fang, Wang and Ren, 2017). This enables deep neural networks to reduce memory consumption on embedded devices. Tinier-YOLOv3 reduces the parameter size and enhances real-time performance and precision. By examining the number and placement of fire modules, the modules from SqueezeNet were incorporated to reduce the parameters and overall size of this model. Tinier-YOLO retains the first five convolutional layers from the original structure and introduces five fire-fighting modules to compress network parameters. Additionally, Tinier-YOLO merges the pass-through layer between the feature map and the first detection layer, introducing dense connections to enhance feature propagation. The scale of the Tinier-YOLO model is approximately four times smaller than that of Tiny-YOLO-V3, achieving a mean average precision of 34.0% based on the COCO dataset and 65.7% mean average precision on PASCAL VOC.

YOLOv4 incorporates novel enhancements building upon the foundation of YOLOv3. In YOLOv3, a single anchor was assigned to each ground truth. However, YOLOv4 introduces the utilization of multiple anchors for a single ground truth. While the number of anchor boxes remains unchanged, the selection of positive samples has been increased, thereby mitigating the discrepancy between positive and negative samples.

The applications of YOLOv4 exhibit remarkable diversity (Zhu, et al, 2020) of experiments employing a sound imager, utilizing sound phase as a measurement tool to evaluate the combination of sound wave and lightwave images. Essentially, the YOLOv4 model was adapted to identify error indicators and assess sound source localization results. The combination of DenseNet and clustering algorithm offers the YOLOv4 algorithm with high accuracy and facilitates the extraction of image features. The refined model achieved a precision of 96.3% during testing.

Drones are employed for real-time detection of bridge cracks (Yu, Shen, and Shen, 2021), wherein focal loss is employed to optimize the loss function. The utilization of a multiscale dataset expands the predictable range of the enhanced model, YOLOv4-FPM, and strengthens its scale robustness. The mAP of YOLOv4-FPM attains 0.976, surpassing the original YOLOv4 model by 0.064. In addition, a pruning algorithm is employed to streamline the network architecture and expedite the detection speed.

YOLOv4 offers advantages in terms of speed and accuracy, making it suitable for real-time detection. Hu et al. (2021) applied YOLOv4 to detect feed pellet consumption and water pollution in aquaculture. For fruit recognition, image collection using a mobile camera is influenced by environmental factors such as light and climate, which introduce noise. Underwater image collection poses challenges due to low image quality and the presence of extremely small particles. These challenges were addressed by utilizing a fine-grained YOLO feature map and modifying the connection mode of the feature pyramid network (FPN) and the path aggregation network (PANet). The remaining connection mode of CSPDarknets improved network performance and function reuse. De-redundancy operations reduced network complexity while ensuring detection accuracy. The experiment utilized real environment images, yielding influential results. The precision value achieved in the experiments was 92.61%, applicable to actual aquaculture environments.

Visual object detection was applied to medical image recognition. The utilization of YOLOv4 facilitated the identification of the hip joint, knee, and ankle regions in the entirety of the leg X-ray photograph (Tack, Preim, & Zachow, 2021). Subsequently, this enabled the complete alignment of the knee in the full-leg X-ray image through an automated process. The residual network was implemented to regress the coordinates of the region of interest. The detection of the hip-knee-ankle (HKA) angles served as pivotal landmarks. A dataset comprising of 2,943 medical images was amassed to assess the accuracy of the HKA angles. Through the utilization of YOLOv4 and the ResNet Landmark Regression Algorithm (YARLA), the automatic detection yielded an average discrepancy of 0.63° in the measured HKA angle, which closely approximates the proficiency of a human expert. This achievement provides a promising avenue for the automated identification of medical images.

YOLOv5, the most recent addition to the YOLO family, is predominantly based on the architecture of YOLOv3. It has garnered widespread adoption across various sectors. The customizability of YOLOv5 is evident through its adaptability to diverse datasets.

Computer vision finds extensive application in traffic scenes, comprising intelligent signal lights, traffic sign recognition, violation monitoring, vehicle classification, vehicle speed monitoring, vehicle counting, parking assistance, etc. Kasper-Eulaers et al. (2021) directed their focus towards real-time occupancy prediction of parking spaces in rest regions. The collection of experimental data for real-time monitoring of outdoor scenes is susceptible to the influence of adverse weather conditions. To address this, a dataset comprising of 580 images was enhanced through random application of horizontal mirroring, resizing (zooming and cropping), and grayscale changes. The monitoring of parking spaces directly employs the pre-training weights of YOLOv5, while the frame loss representation approach accurately detects the center and anticipates the bounding box coverage of the object.

The safety helmet monitoring system was developed due to its role in protecting workers (Zhou, Zhao, & Nie, 2021). The dataset used in the experiment consisted of 6045 images, including diverse scenes featuring various types of helmets. To conduct the experiment, the YOLOv5 model was employed with different parameters, resulting in an impressive precision of 94.7%.

Facial masks have proven to be an efficacious measure in curbing the transmission of COVID-19. Machine vision techniques have been harnessed to recognize whether individuals are wearing masks or not (Yang et al., 2020). To mitigate image noise, a smoothing filtering approach was employed to enhance the low-frequency components of the images. The image segmentation algorithm accurately identifies facial features and extracts the relevant regions of interest. The experiment yielded a remarkable success rate 97.0% (Liu, Lu, Peng, & Zhang, 2020). The *k*-means algorithm, when combined with data augmentation methods, enhances the accuracy of the mask recognition model by identifying anchor points during training.

Detecting small visual objects poses a significant challenge, particularly in oceanic environments where such objects are easily overlooked. The detection of small targets often leads to the loss of certain features, resulting in a high detection loss rate. To address this issue, improvements can be made to the YOLOv5 model to enhance the performance of weak classifiers in detecting small targets. Residual networks can facilitate the extraction of features from datasets containing small targets, thereby expediting convergence algorithms and optimizing deep networks (Dou & Yan, 2021). Notably, YOLOv5 is lighter than its predecessor, v4, and exhibits a mere 1.78% missed detection rate.

A model utilizing a YOLOv4-based convolution block was proposed to incorporate an attention mechanism for identifying the maturity of apples based on color discrimination. In order to address the practical considerations of orchard settings, including the size of fruit trees, the influence of branches and leaves on fruit, as well as the actual size and color of the apples, these factors must be taken into account in the context of real object detection. The YOLOv4 model, as proposed by Lu et al. (2022), achieved an accuracy 86.2% in fruit detection, surpassing the original YOLOv4 model by approximately 3%.

To evaluate the performance of the YOLOv4 model in real-time mango detection, a nighttime dataset was collected with images from diverse orchards, varieties, and lighting conditions. The results demonstrated an F1 score of 0.968 and an average precision of 0.983 (Koirala et al., 2019).

The YOLOv2 model was designed for predicting the location of visual objects, with YOLOv3 building upon its foundation. YOLOv3 incorporates the FPN structure to enhance the precision of multi-scale target detection. YOLOv5, in turn, is a modification of the YOLOv3 structure, employing the Cross Stage Partial Networks (CSPDarknet) as the primary framework for extracting visual information from input images.

Environmental factors pose practical considerations in deep net testing. The tests may encounter environmental noise, such as the impact of weather and lighting conditions on the surrounding details of the fruit, the presence of green leaves and background colors resembling the fruits being detected, missed detections due to overlapping fruits on trees, and the challenge of accurately identifying smaller fruits. To address these challenges, our data collection process included the inclusion of images with environmental noise, such as complex scenes featuring apples stacked on trees and pears arranged in a fruit bowl.

Numerous methodologies exist for capturing the intrinsic information pertaining to fruits (Pal et al., 2021). The RetinaNet and SSD frameworks offer viable approaches for predicting the spatial coordinates outputted by the model (Bochkovskiy, Wang & Liao, 2020). The detection task can be effectively implemented through the utilization of the Coordinate-based Anchor-Free (CBAF) technique (Tang, Zhang & Zhu, 2020). The algorithm governing the model's convergence speed plays a pivotal role in addressing the issue of vanishing or excessively minute gradients, thereby ensuring the transfer of features in the network and enhancing the detection results (Redmon et al., 2016).

YOLOv3 models were Typically employed in conjunction with neural network models. Liu et al. integrated DenseNet into the DarkNet model and employed the enhanced DarkNet53 as the backbone of YOLOv3 (Liu et al., 2022). The fusion of Darknet and DenseNet significantly bolstered the pineapple detection accuracy of the YOLOv3 model, yielding a remarkable 97.55% precision and substantially improving the feature maps' capacity to represent data.

Transfer learning was employed to streamline the computational complexity of models. YOLOv5 model was optimized by using means of layer pruning, channel pruning, and adjusting the number of detected heads in the feature map (Wang et al., 2022). The optimized YOLOv5 model achieved a 71% reduction in volume when detecting apple stems, while only incurring a marginal 1.57% decrease in average precision. The optimized YOLOv5 model was harnessed to accurately identify apple stems, achieving an impressive accuracy rate of 93.89%.

Hussain et al. (2022) introduced YOLOv7 as a novel approach for visual object detection. YOLOv7 NAS incorporates an iterative search mechanism that mines the optimal scale factor based on the resolution, width, depth, and number of feature pyramids. By using means of reparameterization, the gradient propagation path is effectively realized, enabling the reintegration of model parameters and facilitating the application of the head module for fruit detection.

To shed light on the comparative merits and demerits of the R-CNN network and YOLOv2, the work (Zhang, Huang, Jin, and Li, 2017) and (Du, 2018) was consulted. While both Faster R-CNN and YOLO employ CNN as their fundamental component, their frameworks exhibit distinct characteristics. Faster R-CNN adheres to the conventional R-CNN framework and processes the entire input image, whereas YOLO models segment the image into grids, eschewing sliding windows and region proposals. Notably, YOLOv2 incorporates 5 anchor boxes, whereas Faster R-CNN utilizes 9 anchor boxes. YOLOv2 was specifically devised to enhance accuracy through techniques such as k-means clustering, which are not supported by Faster R-CNN.

A comparison was made between YOLOv2 Darknet-19 and VGG-16 (Chang et al., 2019). It is observed that the speed of YOLOv2 is four times faster than that of VGG-16. To enhance the performance of the YOLOv2 model (Shafiee, Chywl, Li, and Wong, 2017) a fast real-time detection framework was proposed. The framework leverages the structural characteristics of deep neural networks to optimize hyperparameters using probabilistic genetic coding modeling strategies, thereby achieving network optimization through parameter reduction. The new network, compared to the original YOLOv2 framework, reduces the parameters by a factor of 2.8 and deep inferences by achieving an average 38.13%. However, the detection speed is only increased by 3.3 times.

To address the issue of unoptimized data paths and frequent off-chip accesses, an efficient Tera-OPS streaming hardware accelerator based on YOLOv2 was proposed (Nguyen, et al, 2019). This approach involves storing each network model in the on-chip memory to minimize off-chip access. In addition, binary weighting, low-level activation, and hardware-centric quantization techniques were employed to achieve the desired detection accuracy.

The Tera-OPS architecture is specifically designed to introduce YOLO-LITE (Huang, Pedoeem, and Chen, 2018), a real-time object detection model suitable for portable devices lacking a graphics processing unit (GPU). The retraining and quantization of the network are accomplished using 1-bit weights and flexible low-level activations. The model achieved mAP scores 33.81% and 12.26% based on the PASCAL VOC dataset and COCO dataset, respectively.

For the purpose of passion fruit detection, Ou et al. (2023) proposed an enhanced FSOne-YOLOv7 model. In order to detect passion fruit in challenging natural environments, novel backbone networks, namely slim-neck and ShuffleOne, were employed, along with an upgraded YOLOv7 network for the neck network. To enhance feature extraction and fusion capabilities, the FSOne-YOLOv7 model utilized gradient weighted class activation mapping. These researchers achieved an average accuracy of 94.5% with this improved model, which demonstrates its superior ability to extract features and enhance detection speed.

Similarly, a project was explored based on the potential replacement of manual picking of dragon fruits with visual inspection (Zhou, Zhang, and Wang, 2023). A PSP-Ellipse method was propounded based on YOLOv7 for classification purposes. This method involves segmenting the detection target and employing an ellipse fitting algorithm to identify the endpoints of dragon fruits in images. Subsequently, ResNet is utilized to carry out the classification task. In the PSP-Ellipse endpoint detection task, the model achieved an impressive accuracy 92% for dragon fruits.

Previous agricultural-related detections (Wu et al., 2022) were primarily based on the classification of fruit color and shape. However, traditional agricultural detection methods are susceptible to false positives in complex natural environments, and the model lacks robustness. Wu et al. conducted a study on target detection in complex environments, utilizing the module characteristics of YOLOv7 data improvement, and developed an enhanced DA-YOLOv7 model. This model was employed to detect Camellia oleifera under various interferences, including side light, backlight, slight occlusion, and heavy occlusion, thereby enhancing the model's generalization capability in complicated scenarios.

2.4.7 Regression Function

In this thesis, deep learning models are employed for fruit classification. Non-Maximum Suppression (NMS) is a widely adopted approach in detection algorithms to eliminate overlapping detection frames while preserving the optimal detection results. However, the conventional NMS algorithm possesses certain limitations. For instance, excessively stringent threshold settings may lead to missed detections, whereas loose threshold values may result in redundant detections. To address these issues, researchers proposed the Soft-NMS algorithm, which suppresses non-maximum values in a novel manner, thereby optimizing the performance of the target detection algorithm.

Moreover, Soft-NMS takes into account the overlap between the score and the border during the execution of non-maximal suppression.

In the context of NMS, the procedure involves extracting the frame with the highest score and subsequently evaluating its degree of overlap with the remaining frames. Soft-NMS, on the other hand, incorporates the Gaussian index of the obtained IOU and applies it as a weight to the original score. This weighted score is then reorganized, and the process continues iteratively.

In the NMS algorithm, any detection frame surpassing the specified IoU threshold has its score directly assigned a value of 0. Conversely, soft NMS penalizes the score of such frames, thereby attenuating it. There are two attenuation methods. The first method involves utilizing the product of the IoU and score as the attenuated value. However, this approach may disrupt the sorting order of scores when the penalty attenuation function is applied to frames that marginally exceed or fall slightly below the threshold. Therefore, a reasonable penalty function should impose a substantial penalty for high IoU values, a lesser penalty for low IoU values, and exhibit a gradual transition between the two values.

2.5 Previous Work and Our Work

Our ongoing experiments draw inspiration from prior research. The process of visual object detection involves annotating the dataset, training a detector, and subsequently applying the pre-trained detector to predict the presence of fruit. Early sliding window detection models necessitate computationally intensive calculations. While the object detection can be achieved by using these models, real-time detection remains a crucial requirement for practical applications.

The one-stage model confers significant advantages in terms of detection speed compared with the sophisticated two-stage detection approach. Given the expeditious detection capabilities inherent in YOLO models, we have adopted it as the standard one-stage model in our experimental setup. In addition, the distinctive attention mechanism of the Transformer model facilitates the transfer of feature maps in the model, thereby leading to more precise detection results.

Another two-stage detection model of interest is Mask R-CNN. We have combined the Transformer model with the mask module from Mask R-CNN and conducted ablation experiments to compare the outcomes. Concurrently, we have also selected conventional detection models, such as CenterNet, ConvNext, and others, for experimental evaluation.

In the context of agricultural-related object detection, including domains such as tomato disease detection, green apple detection, wheat disease detection, pineapple detection, apple stem detection, etc., we shall compare our experimental findings against similar agricultural detection models.

2.6 Summary of This Chapter

To commence the exploration, we delve into the intricacies of numerous visual object detection methods. These methods include a diverse array of algorithms, models, and strategies, each endowed with unique advantages and applications. By clarifying these techniques, we seek to offer a comprehensive understanding of the resources at our disposal and their potential effects on fruit recognition.

Additionally, a significant correlation between the aforementioned detection methods and the role in fruit recognition will be established. The localization and identification of fruits within photographs or videos are of paramount importance in the context of fruit detection, as it represents a practical application of visual object detection. In order to facilitate automated fruit analysis and categorization, the underlying principles of these detection techniques will be explained, as they serve as the fundamental building blocks upon which precise fruit recognition systems are constructed.

A critical evaluation of the existing work will be undertaken, alongside an exploration of the mechanics of visual object identification and its relevance to fruit recognition. The retrospective analysis of this field is essential for understanding its evolution, identifying emerging trends, and identifying areas of knowledge deficiency. By acknowledging the contributions and limitations of prior research, we are able to better appreciate the significance of our own research work in advancing the field of fruit detection.

As we progress through this chapter, we will acquire the foundational knowledge necessary to navigate the complex scenarios of visual object detection and its integration with fruit recognition. Our knowledge will serve as a robust basis for subsequent chapters, wherein our experimental methodologies and findings will be expounded upon in greater detail, ultimately enhancing our understanding of this captivating intersection between technology and agriculture.

Chapter 3 Methodology

This chapter clarifies the methodology employed in the fruit detection experiments, while also delving into the specifics of object detection research methods grounded in deep learning. These include the one-stage detection, two-stage detection, and transformer approaches. Additionally, the regression method shall be expounded upon. A comprehensive exploration of the methodological intricacies pertaining to our fruit detection experiment is undertaken, with the aim of offering a comprehensive understanding of the methodology employed in our quest for precise and efficacious fruit detection. In addition, we introduce a multitude of deep learning-based methodologies for object identification, shedding light on their unique features and applications.

3.1 Fruit Object Detection Experiment Design

Fruit detection constitutes an object detection task in computer vision. Figure 3.1 provides an exhaustive detail of a fruit ripeness detection model. The objective is to determine the presence of apples or pears in the detected image, subsequently employing pyTorch to define the boundaries of the fruit. Transformer, YOLO, and other models were employed to train the annotated images. Following the training process, the model generates a detector that encapsulates the weightings of detailed features pertaining to the detected target. This detector can be utilized to accomplish the detection task, yielding predicted results and facilitating model evaluation.

The principal steps involved in fruit ripeness identification are:

1) Data preprocessing

Prior to object detection, the dataset necessitates preprocessing to facilitate the extraction of the object's feature map. In the context of fruit detection in a simulated environment, the consideration of extreme outdoor weather conditions necessitates operations such as data image enhancement, scaling, cropping, and rotation to enhance the model's adaptability to the scene requirements.

In the early two-stage model, the functional modules of data augmentation and scaling were absent. Therefore, the methods employed for fruit classification based on Faster R-CNN must be augmented by incorporating zooming and rotation of the image data during dataset creation. Conversely, the one-stage model inherently possesses a data augmentation module, thereby obviating the need for additional rotated and scaled data samples in the dataset.

By considering the aforementioned circumstances, the data may now be directly utilized for training the one-stage model, as it no longer necessitates adjustments to the dataset size or dimensions. However, substantial preprocessing of the data images remains necessitated for the Faster R-CNN model based on the two-stage approach.

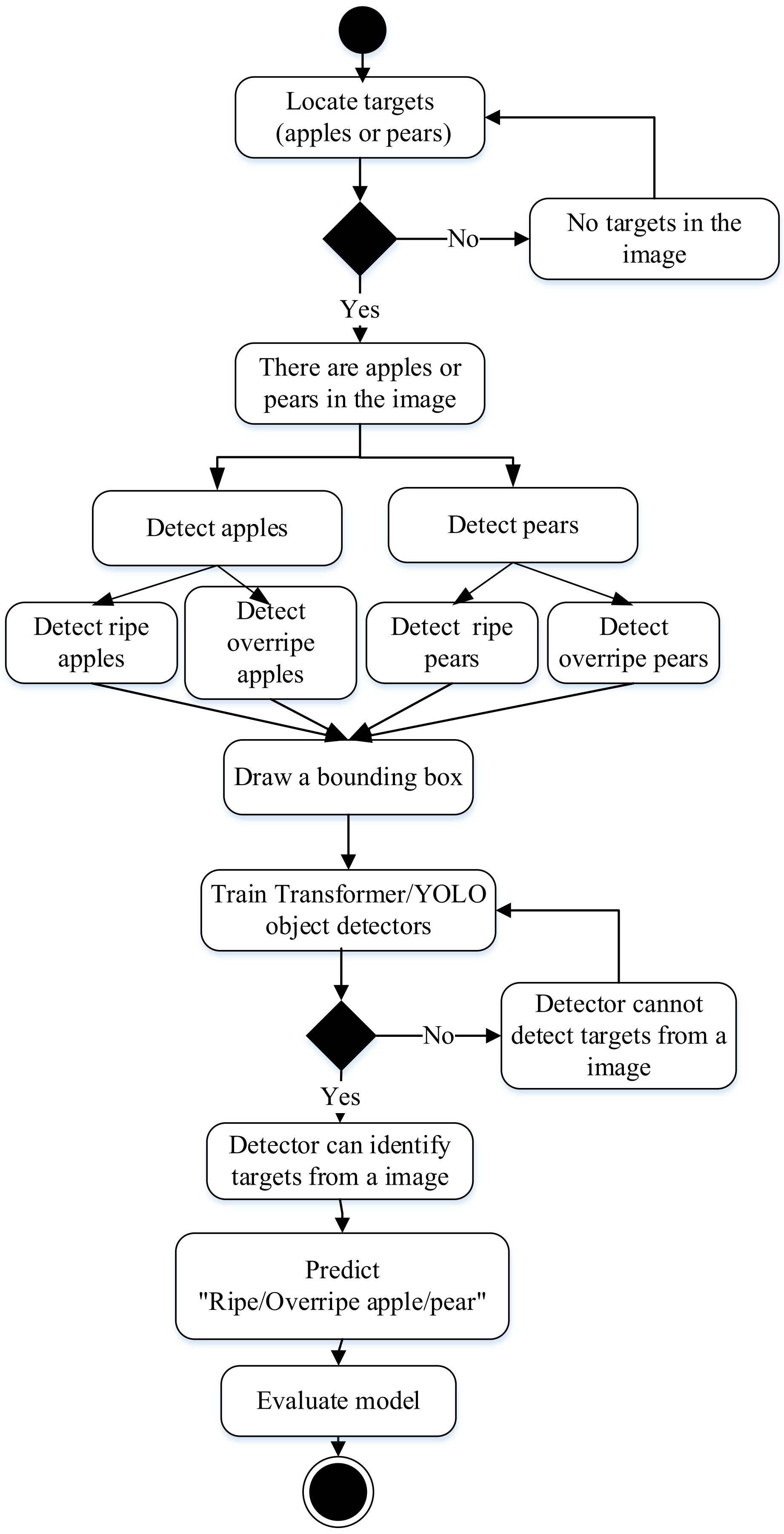

Locate targets (apples or pears)
No
No targets in the image
Yes
There are apples or pears in the image
Detect apples
Detect pears
Detect ripe apples
Detect overripe apples
Detect ripe pears
Detect overripe pears
Draw a bounding box
Train Transformer/YOLO object detectors
No
Detector cannot detect targets from a image
Yes
Detector can identify targets from a image
Predict "Ripe/Overripe apple/pear"
Evaluate model

Figure 3.1: The flowchart of fruit ripeness identification

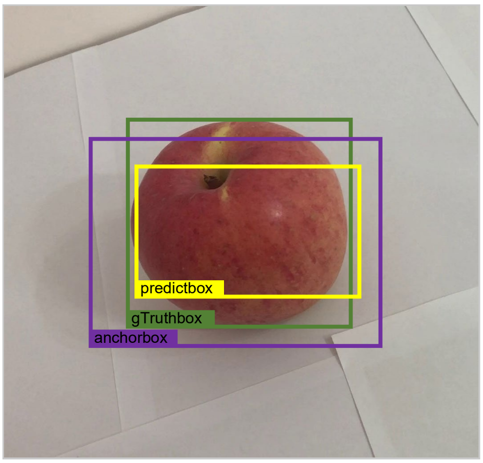


Figure 3.2: An example of the object bounding box

2) Data Labelled

Figure 3.2 reveals that manual labeling of the dataset is indispensable for fruit object detection. The labeled dataset is depicted in both Figure 1.1 and Figure 3.2. We engage in the manual labeling of dataset features, subsequently feeding the data into the model for training purposes. The bounding box of the labeled dataset is represented by the box in Figure 1.1, while the green box in Figure 3.2 signifies the ground truth box manually marked by our team.

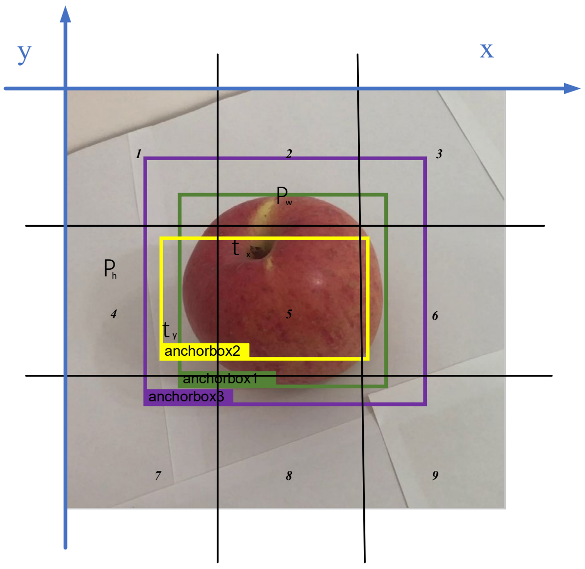


Figure 3.3: Anchor box in the image

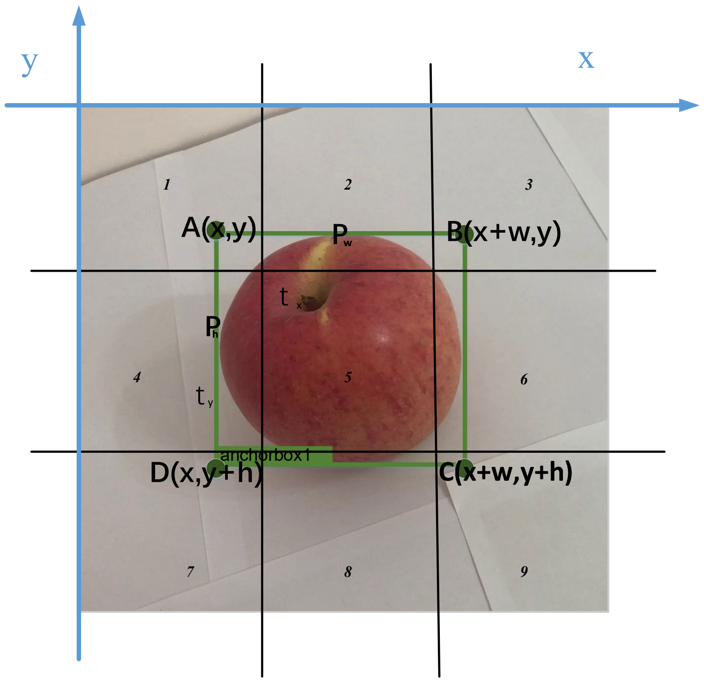


Figure 3.4: Coordinates of anchor box

The coordinates of the four corners of an anchor box are (A, B, C, D). The position is represented by (x, y), while its size is shown by (w, h). T in eq. (3.1) stands for the anchor box in the model, (x, y, w, h) is denoted as coordinates, the value is not fixed, but a relative value ranges. C_1, C_2,C_3 and C_4 represent the labels of three 4 classes: “ripe apple”, “overripe apple”, “ripe pear” and “overripe pear”, respectively. The input is the whole image and the output is a target label T. The target label T in the anchor box means a vector.

$$T = (x, y, w, h, c1, c2, c3, c4)^T \tag{3.1}$$

3) Train Object Detection Detector

In Figure 3.1, the focus of the experiments lies in training the detector. The utilization of the model is employed to execute machine learning of fruit characteristics, including color and the presence of skin wrinkling. Subsequently, the trained model possesses the capability to prognosticate the category to which an unlabeled fruit image belongs. The model's training process yields an anchor box, depicted as a purple box in Figure 3.2.

4）Evaluation

Upon completion of the detection process, the precision value is employed to assess the merits and demerits of the detection model. The model is analyzed and evaluated through a comparative analysis utilizing a line chart. The predicted outcome, referred to as the predict box, is represented by a yellow box in Figure 3.2. Our methodology is predicated on each stage of fruit detection.

3.2 Previous Research

Visual object detection involves the identification of multiple objects in a singular image. Convolutional neural networks, alongside other network architectures, are employed in various manners for object detection. In previous experiments, both supervised and unsupervised learning methodologies were employed to accomplish fruit detection.

Three predefined categories were created, and 1,000 samples are subsequently divided into these three categories. The clustering algorithm is then employed to group the 1000 apple pictures, ensuring that all apples belonging to the same category are clustered together. For instance, ripe apples are grouped in the same cluster. On the other hand, unsupervised learning is different form our model. Unsupervised learning does not necessitate manually labeling the datasets. Instead, it utilizes the widely employed clustering methodology, which finds application in various domains of statistical data analysis. Clustering, as a machine learning approach, organizes data points based on predetermined guidelines. By considering a set of data points, a clustering method is employed to assign each data point to a specific cluster. Ideally, data points in the same class should exhibit comparable qualities or traits, while those in different classes should differ significantly in these aspects.

Supervised learning, on the other hand, is employed for pattern classification using manually labeled datasets. In our previous experiments, we focused solely on apple detection. The apples were manually labeled as "ripe apple," "overripe apple," and "unripe apple." Subsequently, our models were trained using the Faster R-CNN technique.

Deep learning-based detection algorithms can be broadly categorized into two types, as previously explained: one-stage detection algorithms and two-stage detection algorithms. In the two-stage process, training redundancy is generated through the sliding window method and repeated traversals of the images. The two-stage model iteratively calculates bounding boxes that converge to the ground truth, as depicted in eq. (3.2).

$$R_1(A_1, B_1, C_1, D_1) \rightarrow R_i(A_i, B_i, C_i, D_i) \rightarrow \cdots \rightarrow R_{ground\ truth}(A_1, B_1, C_1, D_1) \qquad (3.2)$$

Table 3.1: Results of previous work

Method	Epoch	Network	Precision of ripe	Precision of overripe	Precision of unripe
Faster R-CNN	30	11 layers network	37%	47%	19%
		GoogLeNet	32%	54%	21%
		ResNet-50	36%	53%	17%
	10	ResNet-50 Transfer Learning	66%	63%	12%
	10	11 layers network	34.33%		
		ResNet-50	35.33%		
		GoogLeNet	35.67%		

In Table 3.1, due to the constraints imposed by the two-stage model, a longer duration is required for the region of proposal to traverse an image, thereby impeding the model's ability to engage in further iterative training. By traversing the image, the region of proposal can assimilate a greater array of features, thereby enabling Faster R-CNN to acquire a more comprehensive understanding of the targets. Therefore, the Faster R-CNN model has demonstrated commendable proficiency in the classification of fruits.

In contrast to the two-stage model, the one-stage model solely necessitates the submission of the entire image to the inspection model for feature extraction, obviating the need for region proposals. While the two-stage model must repeatedly undertake classification and bounding box generation, the one-stage model has already accomplished the detection process. In comparison to Faster R-CNN and Mask R-CNN, YOLO and CenterNet, as one-stage models, are better suited for real-time detection scenarios.

3.3 Evaluation Methods

In the context of multiclass classification, each individual class can be conceptualized as a curve plotting the recall and precision rates. The mean average precision (mAP) is determined by calculating the average of the individual average precisions (AP) across all categories. The average precision is represented by the area beneath this curve. In order to assess the efficacy of the proposed model, the evaluation mAP is employed.

The samples under study have been categorized into four distinct groups for the purpose of identifying fruit ripeness. This categorization is based on a comparison between the actual class labels and the anticipated ones. These groups are denoted as True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN).

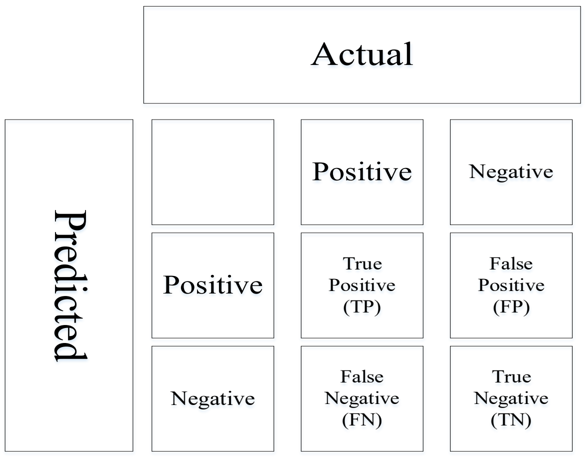


Figure 3.5 Confusion matrix

Figure 3.5 illustrates a confusion matrix comprising four cells, each of which encapsulates all conceivable results of prediction results in the context of a two-category prediction. Following the prediction of any given sample, the resultant prediction must necessarily belong to one of these four aforementioned categories.

True Positive (TP) signifies that the sample is genuinely Positive and the model accurately predicts it as Positive. This is the point at which the prediction materializes. The True Negative (TN) demonstrates that the sample is genuinely Negative, and the model correctly predicts it as Negative as well. This is the point at which the prediction materializes. The False Positive (FP) arises when a sample is genuinely negative, yet the model predicts it as positive. This constitutes an erroneous prediction, and it also represents the first type of error (Type I Error) in statistical analysis. The False Negative (FN) indicates that the sample is actually Positive, while the model predicts it as Negative. This constitutes a component of forecast error, which also represents the second category of error in statistical analysis (Type II Error). In eq. (3.3), precision is employed as an indicator to evaluate the model.

$$Precision = \frac{TP}{TP+FN} \tag{3.3}$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{3.4}$$

Accuracy refers to the proportion of correctly predicted samples out of the total number of samples in eq. (3.4). The accuracy rate serves as a relatively straightforward and intuitive means to present the prediction results and assess the accuracy of the predicted results. However, in our experiment, we not only aim to obtain a predicted result but also need to evaluate the merits and drawbacks of the entire model. Therefore, accuracy is not considered as a criterion to evaluate the experimental results.

In accordance with the findings depicted in Figure 3.6, a positive sample is defined as one wherein the Intersection over Union (IoU) between the projected bounding box and the actual bounding box is equal to or surpasses 0.5. While mAP analyses the detection performance across multiple classes, average precision (AP) evaluates the detection efficacy for a singular class.

The confidence criterion for IOU of AP0.5 is established at 0.5, and solely the preselected bounding boxes with an IOU exceeding 0.5 are employed during the computation process. mAP@.5:.95 denotes the mAP calculated over various IOU thresholds, ranging from 0.5 to 0.95, with an incremental step value of 0.05.

IoU serves as a prevalent metric for assessing the precision of recognizing associated objects within a given dataset. IoU provides a user-friendly metric for evaluating the accuracy of output bounding boxes, which function as predicted regions. In the context of object detection, IoU effectively quantifies the overlap between our projected and actual bounding boxes by expressing the ratio of intersection to union.

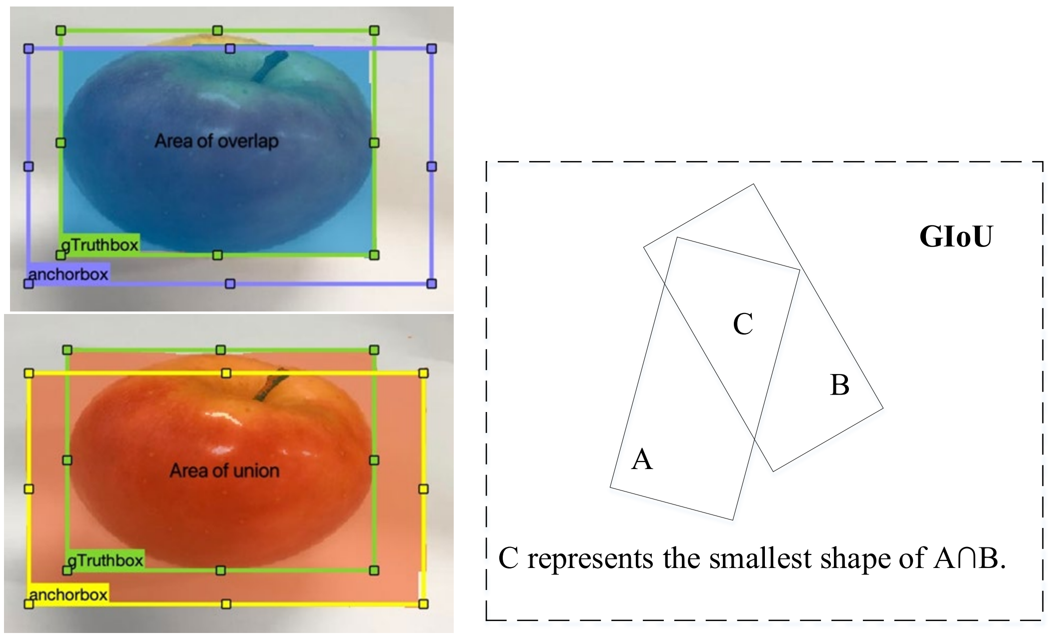


Figure 3.6 An example of IoU

In Figure 3.6, the region shaded in blue represents the union of the ground truth box and the predicted anchor box. Conversely, the region shaded in orange signifies the intersection of the ground truth box and the predicted anchor box.

In practical applications, when the two boxes do not intersect and IoU value equals 0, the model cannot reflect the distance (coincidence) between the two. Simultaneously, due to the loss value being 0, there is no gradient feedback, impeding the learning and training processes. Therefore, the accurate representation of the bounding box and predicted box through the IoU value becomes compromised.

IoU adheres to several fundamental properties, namely non-negativity, identity, symmetry, and the triangle inequality. By comparing the ground truth box with the predicted box, these properties enable IoU to effectively assess the quality of object detection. In the context of regression tasks, the scale invariance of IoU aids the model in determining the proximity between the predicted box and the ground truth box.

For any two boxes, denoted as A and B, the first step is to identify a minimum box, denoted as C, that encloses both A and B. Subsequently, the ratio of the area of $C - (A \cup B)$ to the area of C is calculated. This ratio is then subtracted from the IoU values of A and B, thus obtaining Generalized IoU, named as GIoU.

GIoU serves as both a metric and a loss function for bounding box regression. It computes the minimum circumscribed rectangle of the two boxes, thereby quantifying the distance between them. Therefore, the issue of zero gradient when the two targets do not intersect is resolved.

If the IOU is equal to zero, it signifies the absence of any intersection between A and B. In this instance, as the distance between the two boxes increases, the GIOU approaches a value closer to -1. Conversely, when the two boxes overlap, GIOU equals 1, thereby the value of GIOU is (-1, 1].

It is important to note that GIoU still assigns significant importance to IoU. However, substantial errors arise in both vertical directions, making convergence a challenging task. When the two boxes are equidistant in the horizontal and vertical directions, the corresponding area is minimized, resulting in a smaller contribution to the loss. Therefore, this leads to a poor regression effect in both the vertical and horizontal directions.

Various metrics, such as IoU, accuracy, recall, and precision, can be employed to evaluate the results of object detection. However, in order to comprehensively assess the model's impact on the detector, precision is chosen as the primary indicator.

3.4 YOLO

Figure 3.10 illustrates the division of an input image into 36 grids. The YOLO model detects whether the central point of each grid exhibits the characteristics of an apple, subsequently generating a corresponding bounding box. Figure 3.11 showcases the multiple anchor boxes generated when the apple features are localized in the grid.

Regarding real-time classification with minimal prediction errors, the YOLO model employs a deep neural network that takes an entire image as input. The arrangement of the grid is relatively sparse, in each grid, the YOLO model predicts only two bounding boxes. However, this approach leads to subpar detection performance for small items and objects that are in close proximity to one another. In addition, the pooling layer of the neural network results in the loss of detailed information, thereby reducing the accuracy and recall rate of the YOLO model when recognizing visual objects.

Data augmentation is an integral aspect of the YOLO series models. Various methods are employed to enhance the data, including geometric distortion, illumination distortion, self-adversarial training (SAT), and image occlusion techniques such as random erasure, shearing, hiding and finding, grid masking, blending, etc.

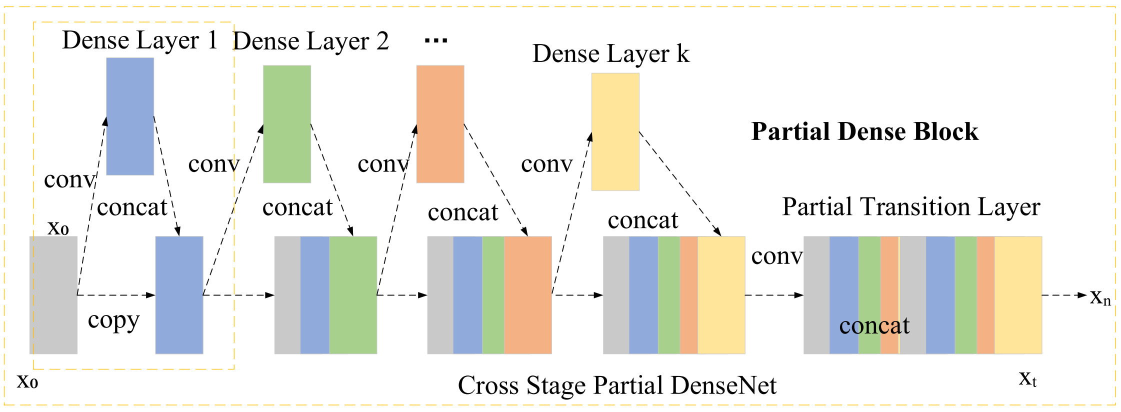


Figure 3.7: The cross stage of DenseNet

While selecting a detector, it is crucial to consider a backbone network that possesses robust image feature extraction capabilities, as the mAP evaluation index is utilized to attain a higher mAP value. If the structure of the backbone network is excessively large, it will significantly impede the detection speed. Conversely, if it is too small, the extraction effectiveness of target features will be compromised. In the YOLOv5 object detectors, the Darknet53 with CSP structure is chosen as the backbone network. The CSPNet, depicted in Figure 3.7, represents a cross-stage local network architecture.

The efficient resolution of the issue pertaining to information redundancy within the backbone network is achieved through the incorporation of YOLOv4 and YOLOv5 into the CSP structure. The CSP structure serves to optimize large-scale neural networks by significantly reducing the parameter count and floating-point operation (FLOPs) of the proposed model. Therefore, this augmentation leads to an enhancement in the inference speed of the final model.

CSPNet, a derivative of DensNet, employs a technique whereby the feature map of the base layer is duplicated and routed through the dense block, as depicted in Figure 3.8. The segregation of the feature map from the base layer enhances the capacity of our proposed model to transmit network information, thereby mitigating the issue of gradient vanishing commonly associated with deep networks.

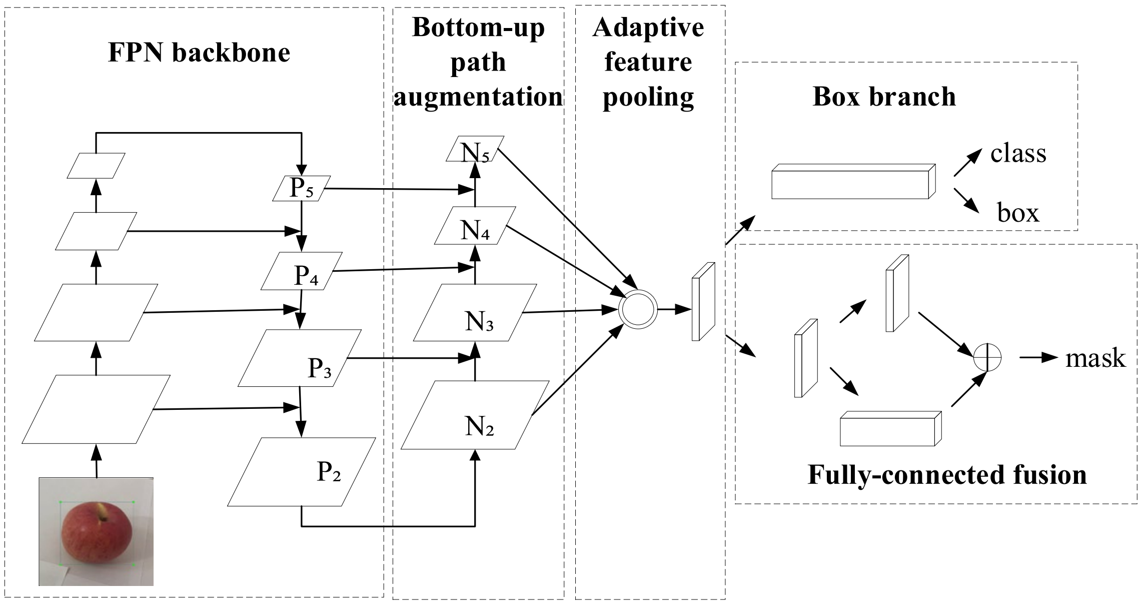


Figure 3.8: CSP network

The YOLO series models consist of three primary components: the backbone, neck, and head. The backbone module assumes the principal responsibility of extracting features from the input image. Simultaneously, the neck module undertakes the task of performing multiscale feature fusion on the feature maps and transmitting these features to the prediction layer. Finally, the head module is tasked with generating the final regression predictions for the observed objects.

The main function of the backbone network is to extract image features, thereby generating a multilayer feature map from the initial input image (i.e., the image to be identified). This feature map serves as the foundation for subsequent item detection tasks. It is worth noting that lightweight backbone networks, such as ResNet and CSPDarknet53, are employed to strike a balance between memory utilization and computational efficiency, all while maintaining a high level of detection accuracy.

Upon assessing the dataset sample depicted in Figure 4.1, it can be observed that the size and position of the fruit in different images are not fixed. To address this issue of varying target sizes in images, the feature pyramid method is employed for multi-target detection. By incorporating feature layers of different sizes into the backbone network, the feature pyramid technique effectively tackles the challenge posed by uneven target sizes.

Among the primary building blocks of the backbone network are the Conv module, C3 module, and SPPF module. The Conv module, a pivotal component of convolutional neural networks, comprises a convolution layer, a batch normalization layer (BN layer), and an activation function.

Convolutional neural networks are constructed based on the convolutional layer, which is responsible for capturing local spatial information from input features. The convolution kernels serve as the fundamental constituents of the convolution layer, with each convolution layer consisting of multiple convolution kernels. In a two-stage model, the convolution process can be likened to a sliding window technique. As the window traverses the feature map, convolution operations are performed on the feature values within the window, resulting in the generation of output features. The output size and receptive field are influenced by various characteristics associated with the convolution kernel, including stride, size, padding method, and others. Each convolution kernel corresponds to an output channel. Essentially, the convolutional network functions as a feature extractor.

The standardization of the eigenvalue distribution in neural networks is achieved through the utilization of the BN layer. This layer plays a crucial role in determining the mean and variance of the feature values for each channel, thereby facilitating the process of standardization. Additionally, the BN layer incorporates a learnable affine transformation (stretch and offset) operation, to restore the normalized features. Therefore, the output of the BN layer is obtained.

Artificial neural networks possess the ability to capture complex interactions owing to the presence of activation functions, which endow them with nonlinear transformation capabilities. Activation functions such as ReLU, LeakyReLU, ELU, sigmoid, and others inherently exhibit nonlinearity. Depending on the specific requirements of different detection models, various activation functions may be adopted. The selection of an appropriate activation function can significantly impact a model's capacity to adapt to the data distribution within a defined range.

During the confidence prediction process in YOLO, the network simultaneously predicts the width, height, and category information associated with the prediction box. In cases where parallel prediction occurs, the network has assumptions regarding the size and shape of the target. However, it remains unable to precisely determine the exact location of the center of the prediction box. Therefore, multiple predicted boxes may correspond to the same target, resulting in redundancy. To address this issue, non-maximum suppression techniques can be employed to eliminate video frames with low confidence and high overlap.

A crucial component in the framework of a convolutional neural network is the Spatial Pyramid Pooling (SPP) module. The output size of the conventional pooling layer (e.g., maximum pooling or average pooling), is inherently dependent on the input size. However, when implementing classification, the fully connected layer necessitates the specification of the fully connected input. To achieve this, the image is subjected to distortion through the utilization of the resize function. The SPP module, in turn, empowers the neural network to obtain an output of fixed dimensions at a specific layer.

The integration of SPP into a neural network enhances its capacity to recognize objects, thereby playing a pivotal role in boosting the spatial and positional invariance of the input data. By performing pooling operations on feature maps of varying sizes, the

SPP module generates feature maps of diverse dimensions. This method enables the utilization of multiple receptive fields during image processing, thereby capturing feature information at various scales. Finally, the fully connected layer reduces the dimensionality of the feature map, resulting in a fixed-size feature vector.

While SPP and SPPF share a common objective in terms of shape, there are subtle differences in their structures. The transition from SPP to SPPF leads to a reduction in the computational volume of the model, accompanied by an increase in calculation speed.

The Neck module of YOLO is typically employed to combine feature maps from different levels, thereby generating feature maps imbued with multi-scale information. This integration serves to enhance the accuracy of detection.

In Figure 3.8, the feature pyramid structure of FPN combines feature maps of varying levels via up and down sampling operations, thereby forming a multi-scale feature pyramid. The uppermost section of this multi-scale feature pyramid effectuates the fusion of features at various levels through upsampling and fusion with coarser-grained feature maps.

Ordinarily, the Neck module in YOLO serves the purpose of combining feature maps from multiple levels, thereby generating multi-scale feature maps that enhance the accuracy of target detection. This particular functionality is executed by using the PANet feature fusion module in YOLOv4 and YOLOv5.

The bottom-up pathway facilitates the movement of low-level data to the uppermost layers of high-level features. Therefore, smaller feature maps exhibit larger receptive fields as a consequence of feature fusion. In contrast to the FPN approach, PANet significantly reduces the number of feature maps required for feature transfer. The FPN algorithm may necessitate a multitude of layers to transfer underlying features to the apex of the feature map. By eliminating low, medium, and high-level feature maps from the original network, PANet accomplishes both top-down and bottom-up feature fusion, thereby expediting the transfer of feature information.

The Head module of YOLO series is dedicated to the detection of feature pyramids. It involves convolutional layers, pooling layers, and fully connected layers. In the case of the YOLOv5 model, the Head module is responsible for multi-scale object detection, utilizing feature maps obtained from the backbone network. This module consists of three primary components. In order to significantly enhance the accuracy of detection, YOLOv5 incorporates the Mish activation function, GIoU loss, and multi-scale training approaches.

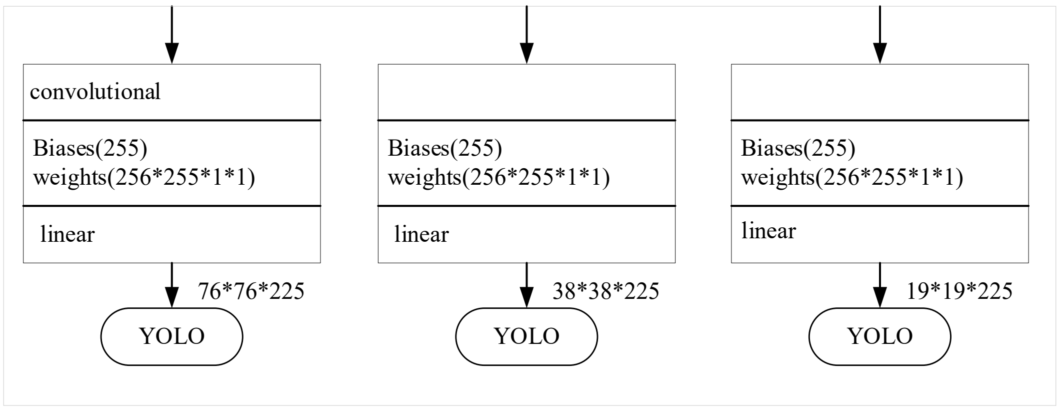


Figure 3.9: Multiple scales applied to the Head of YOLO model

The efficacy of the YOLO model is notable, as evidenced by the adjustment made to the loss function. The original loss function of YOLO, as depicted in eq. (3.5), is observed.

$$
\begin{aligned}
Loss = {} & \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{obj} \left(x_i - x'_{\ i}\right)^2 + \left(y_i - y'_{\ i}\right)^2 \\
& + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{obj} \left(\sqrt{w_i} - \sqrt{w'_{\ i}}\right)^2 + \left(\sqrt{h_i} - \sqrt{h'_{\ i}}\right)^2 \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij} \left(c_i - c'_{\ i}\right)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{noobj} \left(c_i - c'_{\ i}\right)^2 \\
& + \sum_{i=0}^{S^2} I_{ij} \sum_{c \in class} \left(p_i(c) - p'_{\ i}(c)\right)^2
\end{aligned}
\tag{3.5}
$$

where x, y, w, h, c and p represent predicted values. I_{ij}^{obj} indicates that the object is fallen into the j_{th} bounding box of grid i. YOLO model utilizes $\lambda_{coord} = 5$ to correct coordError while calculating the loss. For IoU errors, whether the grid covers the detected objects has an influence on the loss. If the centre point (centroid) of an object is in the grid, YOLO adopts $\lambda_{noobj} = 0.5$ to correct iouError.

The training input is derived from the labelled data, which is carefully selected based on its proximity to the output in terms of weights and deviations. The degree of similarity between the training input and the output is quantified. The loss function incorporates the variance as one of its components, while also considering the normalization of contrast to enhance it. Moreover, the existence of visual objects in the region is represented by using current position, width, and height of the image. The loss function is defined as eq. (3.6):

$$E^N = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{C}(t_k^n - y_k^n)^2 \tag{3.6}$$

where t represents input label, y shows the actual output.

3.4.1 YOLOv5

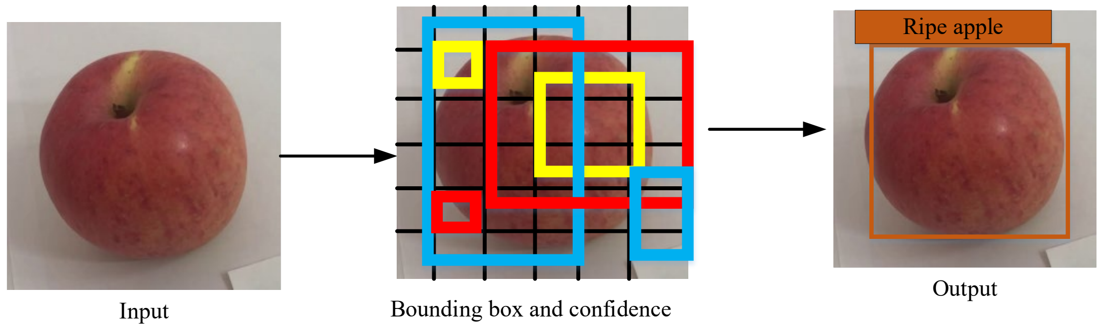


Figure 3.10: The pipeline of YOLO model in fruit detection

The output of the bounding box in Figures 3.3 and 3.4 is introduced. The coordinates of the bounding box are denoted as (x, y), while its size, expressed by its width and height, is represented as (w, h).

Confidence is a crucial parameter yielded by each bounding box. The concept of confidence can be interpreted from two perspectives.

1) Whether an object has been detected in the current box.

The probability that there is an object in the current box is $P_r(Object)$. When analyzing the object in the current box, it is unnecessary to differentiate between an apple and a pear. The sole requirement is to differentiate the presence of fruit in the bounding box.

2) The current box has objects.

Confidence reflects the potential disparity in value between the actual box of the object and the projected box. The confidence level essentially conveys the accuracy of the model's labeling for the object, as there is no actual ground truth box available for the object.

From these two perspectives, we deduce that the confidence level is a probability value, constrained in the range of [0, 1.00]. The degree of confidence is indicated by the certainty that fruit exists in the bounding box and the certainty that the bounding box includes all the features of the complete object. In eq. (3.7), confidence is determined by multiplying the probability of an object in a bounding box by the IOU_{pred}^{truth} (the ratio of union to intersection between the predicted box and the ground truth box). C_i^j indicates that the j_{th} box in i image is responsible for the currently detected target.

$$C_i^j = P_r(Object) \times IOU_{pred}^{truth} \tag{3.7}$$

Our development of the YOLO series models comprises a series of modifications to the fundamental components. YOLOv5 employs mosaic data augmentation. The images undergo mosaic data enhancement, wherein four images are randomly scaled, cropped, and arranged. Subsequently, these processed images are merged into a single image for training purposes. Data augmentation offers the advantage of enabling the model to extract a greater number of features. The mosaic data enhancement method facilitates the extraction of smaller clipped targets in the image for feature extraction. The detection of small targets can potentially result in an imbalance in the proportion of features. However, random scaling aids in balancing the data and enhancing the robustness of this proposed model. The utilization of mosaic method for training, which stitches four images together, has the potential to expedite model computation. Nevertheless, the Mosaic data augmentation function is unsuitable for detecting small targets. A plethora of small target samples fails to provide sufficient features for detection, thereby swiftly diminishing the model's generalization capacity.

In YOLO algorithm, the model determines the initial dimensions for anchor boxes tailored to diverse datasets. During the model training process, the network generates predicted boxes based on these initial anchor boxes. Subsequently, the model calculates the disparity between these predicted boxes and the actual ground truth boxes. The network parameters are then backpropagated to refine the predictions of our proposed model. YOLOv5 employs k-means clustering to dynamically generate the appropriate starting anchor box configuration for distinct training datasets during each training iteration. In this process, k-means clustering computes the distances between each bounding box and all cluster centers. It assigns each bounding box to its nearest cluster center based on category and subsequently computes the category center for the subsequent iteration by averaging the locations of the cluster members.

The computation of classification loss in YOLOv5 is accomplished through the utilization of the cross-entropy loss function ($BCEclsloss$). Additionally, the confidence loss is computed by using the cross-entropy loss function, specifically the BCEcls loss. The YOLOv5 model employs GIOU Loss to calculate the bounding box loss. As depicted in eq. (3.8), the comprehensive loss function of YOLOv5 is the aggregate of three constituent elements: Classification loss ($classError$), localisation loss ($coordError$), and confidence loss ($iouError$).

$$loss = coordError + iouError + classError \tag{3.8}$$

During the training process, YOLOv5 substitutes $softmax$ function with binary cross-entropy when computing the classification loss. This substitution serves to reduce computational complexity and enhance the capacity for calculating labels that are specific to the input.

In instances where the detected object possesses a smaller size, the model assigns a correspondingly smaller anchor, resulting in a reduced feature map size. Therefore, the downsampling rate is increased, thereby enlarging the receptive field.

Upon examining Figures 3.3, 3.4, and Figure 3.10, it becomes apparent that the position of the anchor box remains fixed. However, the position of the fruit (bounding box) undergoes a translation relative to the anchor point position (anchor box). The difference between these two parallel boxes represents the offset that necessitates calculation.

$$t_x = log\left(\frac{bbox_x - c_x}{1-(bbox_x - c_x)}\right) \tag{3.9}$$

$$t_y = log\left(\frac{bbox_y - c_y}{1-(bbox_y - c_y)}\right) \tag{3.10}$$

$$t_{\mathrm{w}} = \log\left(\frac{\mathrm{gt_w}}{\mathrm{p_w}}\right) \tag{3.11}$$

$$\mathrm{t_h} = \log\left(\frac{\mathrm{gt_h}}{\mathrm{p_h}}\right) \tag{3.12}$$

where (x, y, w, h) denotes the coordinate. In eq. (3.9) - (3.12), $(c_x,\ c_y)$ depicts the bounding box coordinate. (t_x, t_y, t_w, t_h) represents the offset of the bounding box and anchor box. (b_x, b_y, b_w, b_h) indicates the final results predicted by the YOLOv5 model.

The entire process is facilitated by using a singular neural network, thereby enabling an end-to-end framework. Upon inputting an image into the YOLO model, it is segmented into a multitude of ss grids. Each of these grids is subsequently subjected to processing in order to yield bounding boxes and confidence scores. The bounding boxes, marked in blue, yellow, and red boxes shown in Figure 3.11, represent the result of this process. The conditional class probabilities, as predicted by each grid, are compounded with the confidence ratings associated with each bounding box. Subsequently, a threshold is established to eliminate low scores, followed by the application of non-maximum suppression, which finally offers the final prediction boxes.

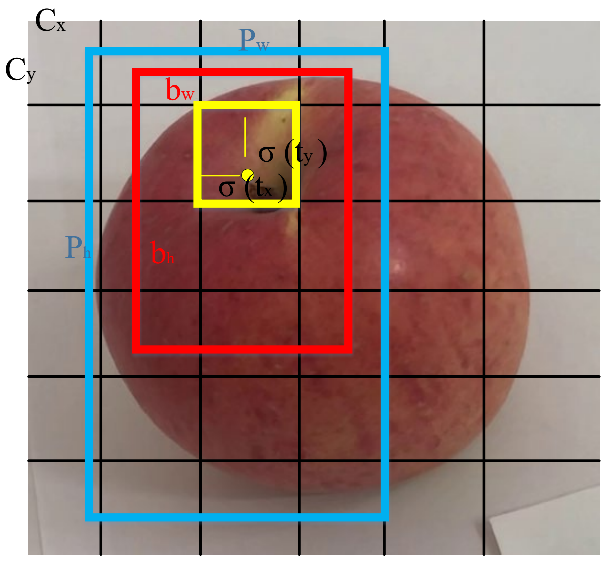


Figure 3.11: The prediction of bounding boxes.

In Figure 3.11, the array (b_x, b_y, b_w, b_h) describes the predicted box, while C_x and $C_y$denote the distance from the top-left corner of the grid to the top-left corner of the image. P_w and P_h represent the width and height of the anchor box, respectively, with σ denoting the sigmoid function. $t_x, t_y, t_w, t_{h,}$ are employed in the calculation of the bounding box and confidence. The center of the predicted box corresponds to the point situated in the yellow box, as illustrated in Figure 3.11. Subsequently, a set of equations is presented to facilitate the calculation process, where b_x, b_y, b_w, b_h represent the final results obtained by using the proposed model, respectively.

$$b_x = \sigma(t_x) + C_x \tag{3.13}$$

$$b_y = \sigma(t_y) + C_y \tag{3.14}$$

$$b_x = P_w e^{t^w} \tag{3.15}$$

$$b_x = P_h e^{t_h} \tag{3.16}$$

The confidence and maximum suppression are:

$$Confidence\ score = P(Object) \times IoU_{truth\_pred} \tag{3.17}$$

and

$$Class - specific\ confidence\ scores = Confidence \times P(Class|Object) \tag{3.18}$$

3.4.2 YOLOv7

In the YOLOv7 model, a decoupled architecture was proposed for training a multi-branch model, enabling its conversion into a single-channel model for deployment. This approach allows for the utilization of the high performance offered by the multi-branch architecture while also benefiting from the fast inference capabilities of the single-branch model.

Figure 3.12 illustrates the merging of all Conv and BN layers in YOLOv7. The fused Conv layer is then transformed into a 3×3 Conv layer. Similarly, a 1×1 Conv layer is converted into a 3×3 Conv layer by employing the center weight, which is equivalent to the 1×1 Conv layer that merges with the branch's 3×3 Conv layer. Therefore, the weights and bias of the convolution kernels from all branches are combined to create a new 3×3 Conv layer. The YOLOv7 model finally incorporates an identity mapping branch, referred to as a RepVGG block. The mapping network of YOLOv7 resembles the residual network of ResNet, wherein a branch is added at a specific layer.

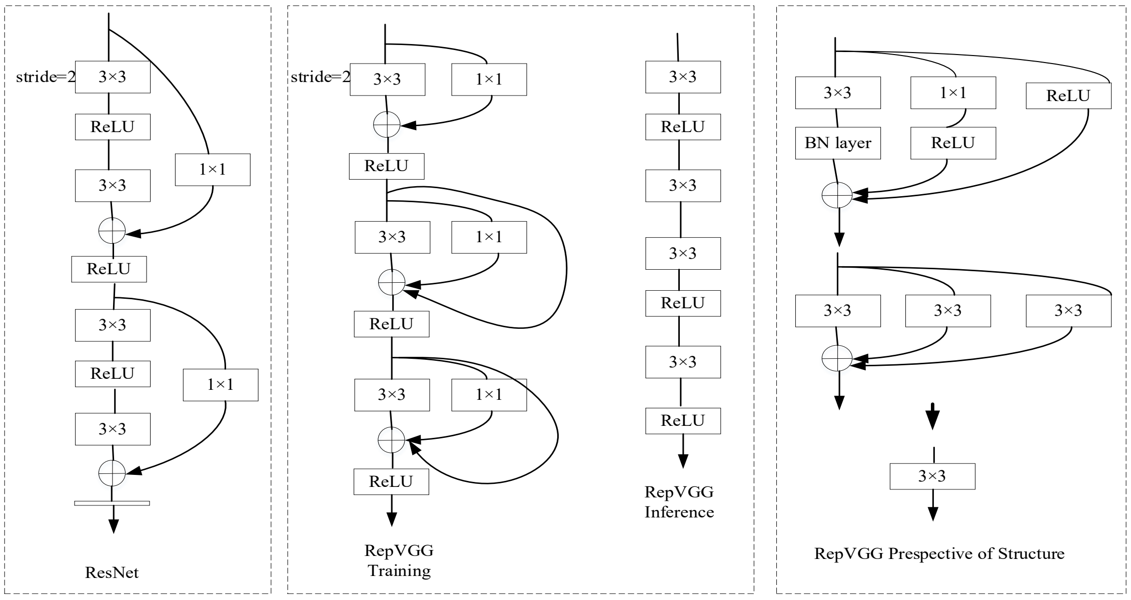


Figure 3.12: Reparameterization process of RepVGG

The parameter fusion process is:

$$W_i' = \frac{\gamma_i}{\sigma_i} W_i \tag{3.19}$$

and

$$b_i' = -\frac{\mu_i \gamma_i}{\sigma_i} + \beta_i \tag{3.20}$$

where μ_i, σ_i, γ_i, and $\beta_i$are the mean, variance, scale factor and offset factor of BN, respectively, W_i is the original convolution weight. Equation for each calculation in Conv is,

$$Conv(x) = W \times X \tag{3.21}$$

$$BN(x) = \gamma_i \left(\frac{x-\mu_i}{\sqrt{\sigma_i^2+\varepsilon}}\right) \tag{3.22}$$

where ε is equal to the minimum. The fused Conv and BN is,

$$\left(\frac{W \times x - \mu_i}{\sqrt{\sigma_i^2+\varepsilon}}\right) + \beta_i = \frac{\gamma_i}{\sqrt{\sigma_i^2+\varepsilon}} W_i X - \frac{\mu_i \gamma_i}{\sqrt{\sigma_i^2+\varepsilon}} + \beta_i \tag{3.23}$$

Ignoring the minimum value ε, the new convolution is,

$$Conv(x) = W' x + \beta' \tag{3.24}$$

In comparison to the calculations performed after fusion, the process essentially involves a linear convolution operation.

Figure 3.13 highlights the significance of scale in the YOLO model. The YOLOv7 model employs a composite scaling method, which involves modifying the depth factor and determining the corresponding change in the transfer layer. By scaling the model and adjusting the width in accordance with the depth scaling factor, the optimal state of the model can be maintained.

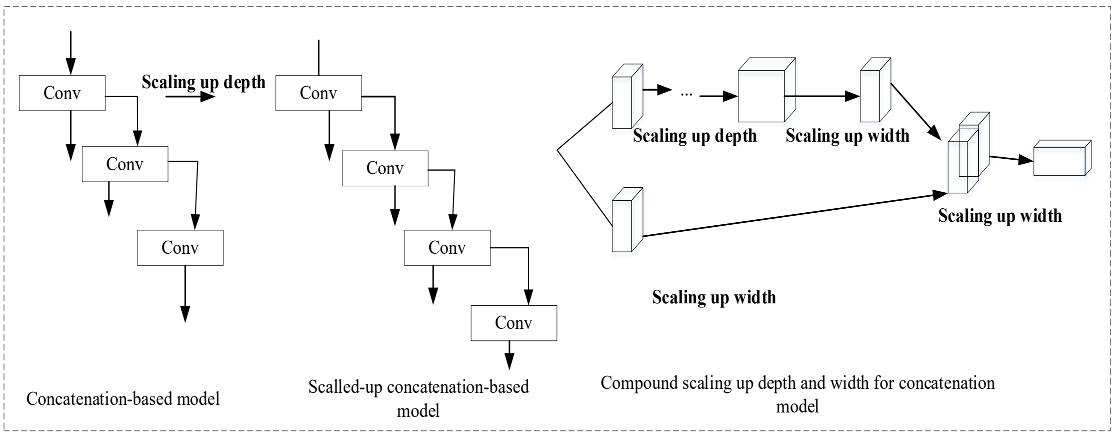

Conv
Scaling up depth
Conv
Conv
Conv
Conv
Conv
Conv
Concatenation-based model
Scalled-up concatenation-based model
...
Scaling up depth
Scaling up width
Scaling up width
Scaling up width
Compound scaling up depth and width for concatenation model

Figure 3.13: Stitch-based model scaling

The YOLOv7-X model employs a compound scaling method to adjust the depth and width of the stack in the Neck module. The YOLOv7-E6 weights are subjected to E-ELAN. The activation function used in YOLOv7, YOLOv7-X, and YOLOv7-E6 is SiLu,

$$SiLu(x) = x \times sigmoid(x) \tag{3.25}$$

The SiLU function, an abbreviation for sigmoid weighted linear unit, serves as the activation function. In contrast to other activation functions (e.g., tanh and sigmoid), SiLU is not a monotonically increasing function. It possesses self-stabilizing properties and acts as an implicit regulariser on the weights. The YOLOv7-tiny weight architecture is designed for edge GPUs. Leaky addresses the zero-gradient problem for negative values by introducing a small linear component to the negative input x.

In Figure 3.14, the YOLOv7 model segregates the auxiliary and dominant heads, assigning labels based on the respective predictions and ground truths. Deep supervision information is incorporated to provide additional guidance to the model, thereby enhancing its performance. The leading head corresponds to the feature map responsible for the final output, while the auxiliary head represents an additional training branch introduced for auxiliary training.

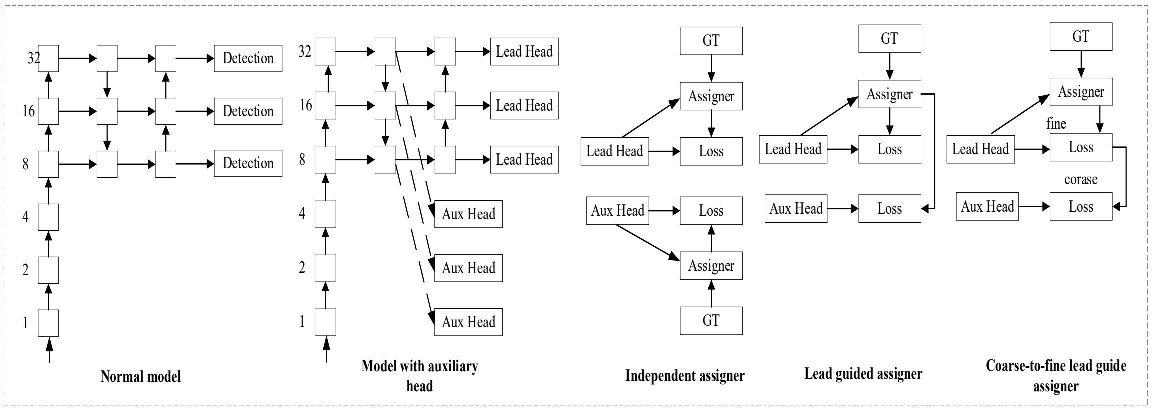


Figure 3.14: Auxiliary head and leading head perform label assignment process using prediction results and ground-truth values

ELAN, an efficient long-range attention network, employs convolutions to extract local structure information from complex images. It utilizes a grouped multi-scale self-attention (GMSA) module to compute on non-overlapping feature sets at various window sizes, thereby accelerating the operation of this module through a shared attention mechanism.

The E-ELAN module of YOLOv7 adopts a feature aggregation and feature transfer process akin to ELAN. E-ELAN employs a group convolution similar to ShuffleNet, an expansion module, and a shuffling module in the calculation module. Finally, it fuses features through the aggregation module. The E-ELAN module enables the acquisition of more diverse features while enhancing parameter calculation and utilization efficiency.

3.4.3 YOLOv8

YOLOv8 represents a significant advancement over its predecessor, delivering a faster, more accurate, and user-friendly model. Building upon the foundation of YOLOv5, YOLOv8 retains the CSP module, as illustrated in Figure 3.15, and introduces the C2f module for visual feature extraction. Notably, in YOLOv8, the CBS 1×1 convolution structure in the PAN-FPN up-sampling stage from YOLOv5 has been eliminated, and the C3 module has been replaced with the C2f module. Another noteworthy improvement in YOLOv8 is the introduction of a decoupled head, which employs two separate convolutional layers for classification and regression tasks, harnessing the concept of

DFL to enhance performance.

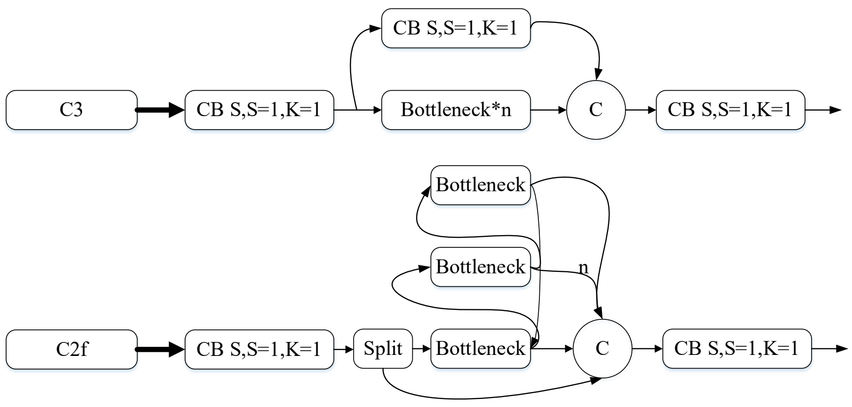


Figure 3.15 C2f block of YOLOv8 model

The most remarkable enhancement in the YOLOv8 model is the implementation of an anchor-free method, coupled with task alignment learning, to synchronize the classification (cls) and regression (reg) tasks. This alignment ensures the accurate positioning of aligned anchors. YOLOv8 introduces a novel anchor alignment metric by multiplying the cls score with the IoU between the predicted anchor box and the ground truth box.

The integration of this alignment metric into the sample allocation and loss function enables dynamic optimization of each anchor's prediction. YOLOv8 employs the VFL loss for classification and a combination of the VFL loss, DFL loss, and CIoU loss for regression.

$$\mathrm{VFL}(p,q)=\begin{cases}-q(q\log(p)+(1-q)\log(1-p)),\ q>0\\ -\alpha p^{\gamma}\log(1-p)\ ,\ q=0\end{cases} \tag{3.26}$$

While FL (Focal Loss) and QFL (Quality Focal Loss) exhibit symmetrical properties, VFL (Volumetric Focal Loss) incorporates an asymmetric weighting process. To address the imbalance between positive and negative samples, VFL is utilized. In eq. (3.26), p represents the label, q is calculated using the $norm\_align\_metric$ when a positive sample is selected, and p is set to 0 when a negative sample is chosen. The $norm\_align\_metric$ weighting scheme serves to accentuate the significance of the samples.

DFL (Distribution Focal Loss) is a technique that transforms the single value of coordinate regression into $n + 1$ output values, wherein each value represents the probability distribution for the respective regression distance. The integration of these probabilities yields the final regression distance. By employing DFL, the network is able to prioritize and concentrate on labeled target y values that are in close proximity to the desired outcome, thereby enhancing the probability of accurate predictions.

$$\mathrm{DFL}(S_i,S_{i+1})=-((y_{i+1}-y)\log(S_i)+(y-y_i)\log(S_{i+1})) \tag{3.27}$$

The essence of DFL lies in its objective to optimize the probabilities of the two positions nearest to the label y (one to the left and one to the right) through the utilization of a cross-entropy formulation. This optimization approach facilitates the network in efficiently directing its attention towards the distribution of the neighboring region surrounding the target position.

3.5 Transformer

In contrast to RPN employed in the earlier Faster R-CNN model, the Transformer model exclusively relies on self-attention mechanisms. The inherent intricacy of the Transformer model contributes to its superior accuracy when compared to the R-CNN neural network.

The attention mechanism emulates human cognitive processes, enhancing efficiency and accuracy by selectively focusing on and processing pertinent information during perception, cognition, and behavioral decision-making.

The attention mechanism shares similarities with CNN in various aspects. It performs computations at each location, combining the information from that location with information from other locations, while disregarding a majority of the surrounding irrelevant information.

In contrast to CNN, the attention mechanism in this study is not confined to a "local" scope. The current position of the attention mechanism is not predetermined or fixed. The attention mechanism does not adhere to fixed "windows" akin to those employed in CNN, where the window size is specified as 3×3. Instead, the attention mechanism is dynamically computed based on the entirety of the input. The Transformer model, which serves as the architectural framework with attention as its core functional unit, allows for the stacking of attention layers in a manner similar to the stacking of layers in a CNN or RNN. As depicted in Figure 3.16, the individual "blocks" or "layers" of the Transformer model execute the proposed operations.

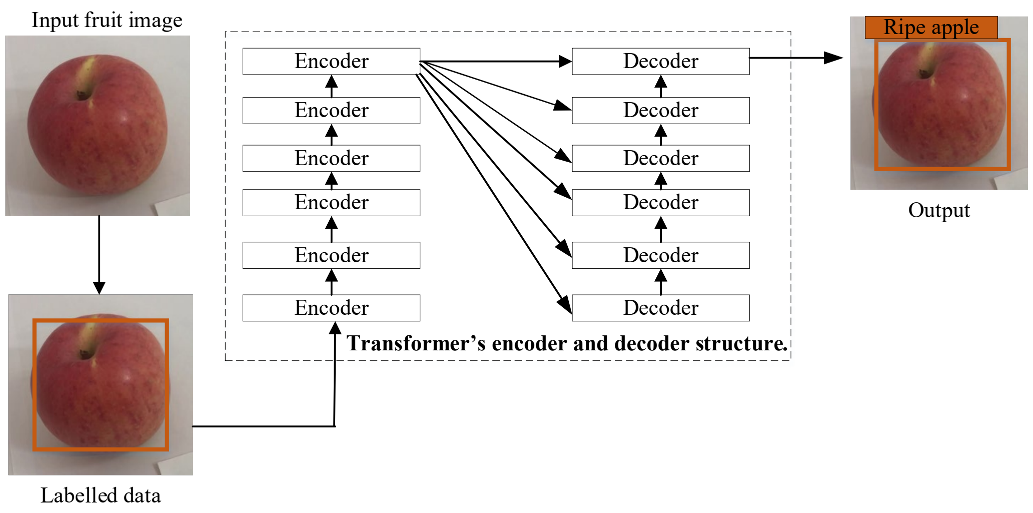


Figure 3.16 Transformer model encoder and decoder block

3.5.1 Swin Transformer

The Transformer model heavily relies on the attention mechanism. By constructing a mask through a series of procedures, as illustrated in Figure 3.17, this model evaluates the relevant characteristics of the fruits that necessitate identification. The first step in the procedure involves embedding the input data, which subsequently serves as the input for the encoder layer. The self-attention layer processes the data before transmitting it to the feedforward neural network. The computation of the feedforward neural network can be parallelized to generate the output, which then becomes the input for the subsequent encoder.

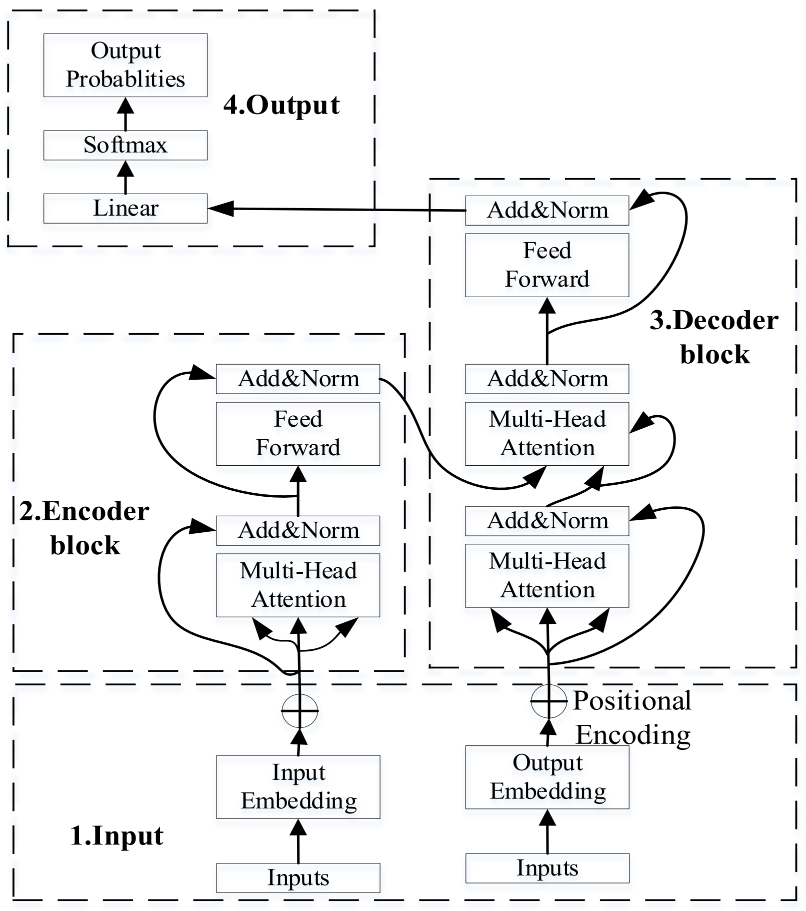


Figure 3.17 The structure of Transformer model

Natural language processing constitutes a mathematical model for vector processing. Voice, images, text, and other forms of information are transformed into sequences of vectors for computational purposes. Therefore, our dataset represents vectorized information resulting from data processing. The self-attention module operates on the entire sequence and employs three matrices of Q (query), K (key) and V (value) to process the dataset.

$$\mathbf{Q} = \mathbf{X} \times \mathbf{W_q} \tag{3.28}$$

$$\mathbf{K} = \mathbf{X} \times \mathbf{W_k} \tag{3.29}$$

$$\mathbf{V} = \mathbf{X} \times \mathbf{W_v} \tag{3.30}$$

where Q represents a query vector in eq. (3.28). K denotes a vector that encapsulates the correlation between query information and other information in eq. (3.29), V indicates a vector of queried information in eq. (3.30).

The attention weight assigned to the information X is directly proportional to that of Q and K, and it corresponds to V. Considering that X itself determines the attention weight, this connection essentially embodies a self-attention mechanism.

Correspondingly, $\mathrm{W_q}$, $\mathrm{W_k}$, and $\mathrm{W_v}$ matrices undergo updates and adjustments in accordance with specific job objectives, thereby ensuring the adaptability and efficacy of the self-attention mechanism. The inner product of vectors includes Q, K, and V, which represent the angle formed by two vectors and the projection of one vector onto the other. A greater projection value indicates a higher correlation between the two vectors.

$$Attention(\mathbf{Q},\mathbf{K},\mathbf{V}) = Softmax\left(\frac{\mathbf{QK}^T}{\sqrt{d}} + \mathbf{B}\right)\mathbf{V} \tag{3.31}$$

The objective of $Softmax$ function is to normalize these values. eq. (3.31) is employed to demonstrate the outcome of the inner product operation between each vector and itself, as well as other vectors. This equation performs the inner product operation with the row vector K^T, the vector itself, and two additional row vectors. Where B denotes a bounding box location, and T signifies matrix transpose. In the Swin Transformer model, both Q and K incorporate relative position coding.

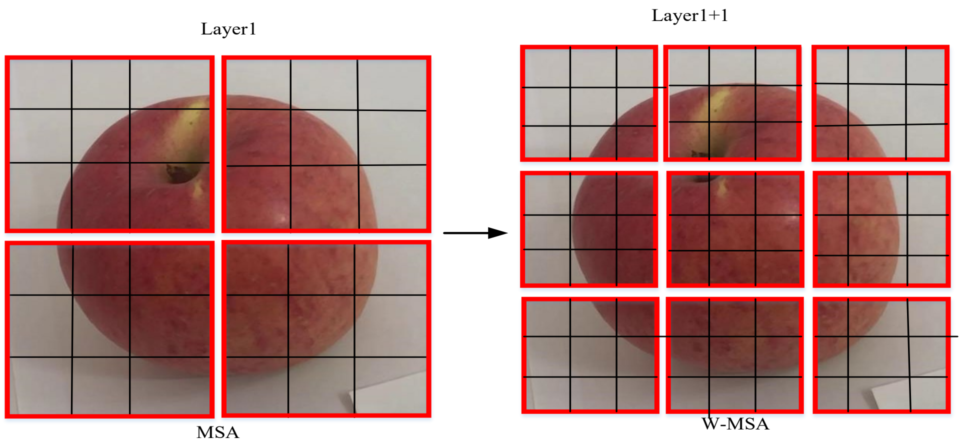


Figure 3.18 MSA and W-MSA

The input image is subjected to a convolutional layer in the Swin Transformer model, wherein each small block is transformed into a pixel to enhance the channel dimensionality. The feature map generated is received by the two transformer layers in the stage. Amidst these steps, a pooling procedure is employed to reduce the data quantity, as the emphasis transitions from initially extracting local information to subsequently extracting global information.

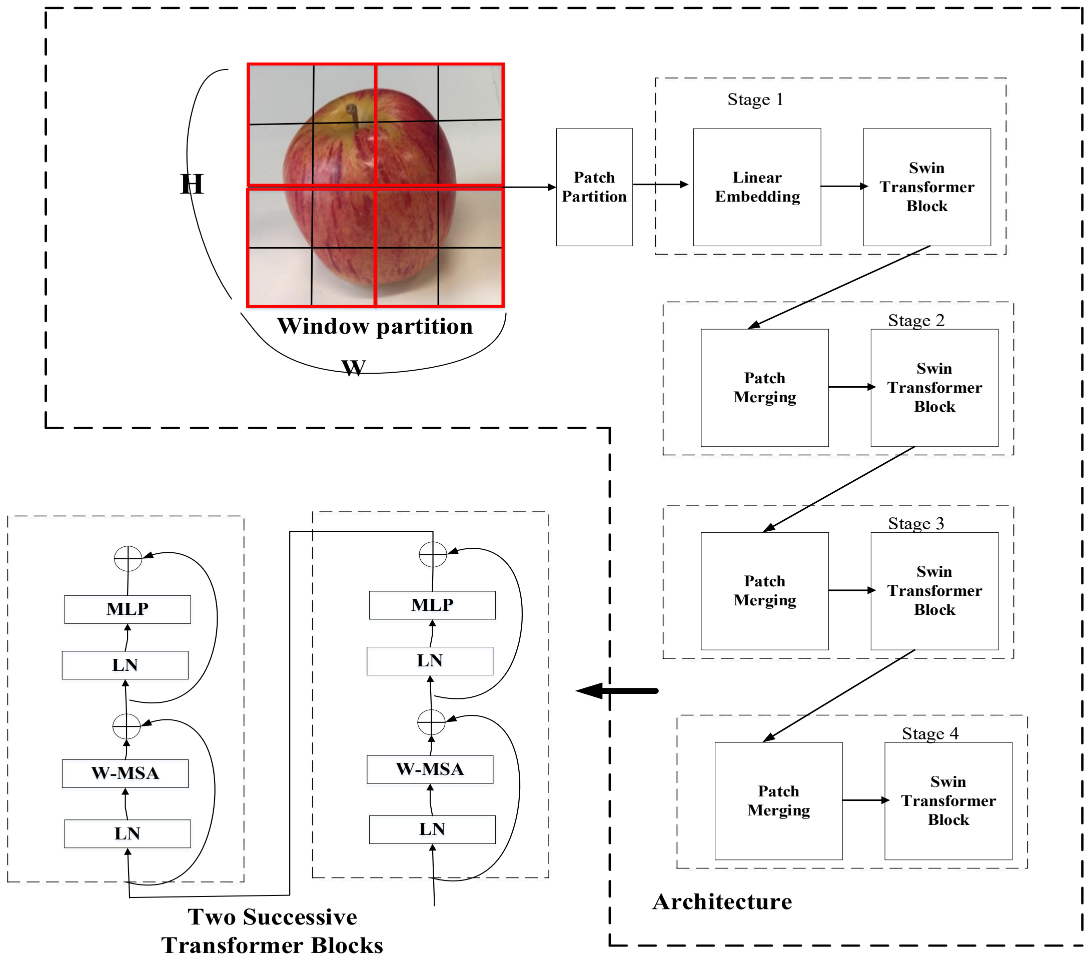


Figure 3.19 The blocks of Swin Transformer

In Figure 3.19, the red boxes indicate a window wherein self-attention is practiced, while the black boxes represent patches. Window attention is utilized to segment the image into different windows based on a desired size. The attention mechanism of the Transformer solely computes interactions occurring inside the window. Both the Swin Transformer and ResNet exhibit a hierarchical structure, with specific emphasis placed on various tiers of data processing. The lower layers of the network concentrate on increasingly localized data, whereas the higher layers handle a reduced dataset, prioritizing semantic information. Notably, the Swin Transformer primarily relies on the Transformers module for information extraction, whereas ResNet employs convolutional kernels for this purpose.

The Swin Transformer architecture comprises four stages, each characterized by a similar structure. In Figure 3.19, the windows where self-attention is applied are denoted by red boxes, while the individual patches are represented by black boxes. The dimensions of the input images in the Swin Transformer are denoted as $W \times H \times C$. Subsequently, these images are organized into a collection of patches at $\frac{H}{4} \times \frac{W}{4}$, with each patch measuring 4×4.

The initial stage of the Swin model involves linear embedding, which converts the features of the input patches into a vector of dimension C. Subsequently, these embeddings undergo processing through a Swin Transformer block. This pattern persists through stages 2 to 4, wherein neighboring patches are merged before being fed into the subsequent Swin Transformer block. The merging of patches facilitates feature down-sampling.

The original feature size of the Swin Transformer is denoted as $[H_1, W_1, C_1]$. The process of window partitioning is determined by reshaping the features based on their original size, resulting in a new size:

$$Reshape\ Size = [\frac{H_1 \times W_1}{window_{size} \times window_{size}}, window_{size}, window_{size}, C_1] \quad (3.32)$$

In Figure 3.20, the distinctive characteristic of the Swin Transformer block lies in its utilization of a shift window, which serves as a substitute for the conventional Multi-head Self-Attention (MSA) module.

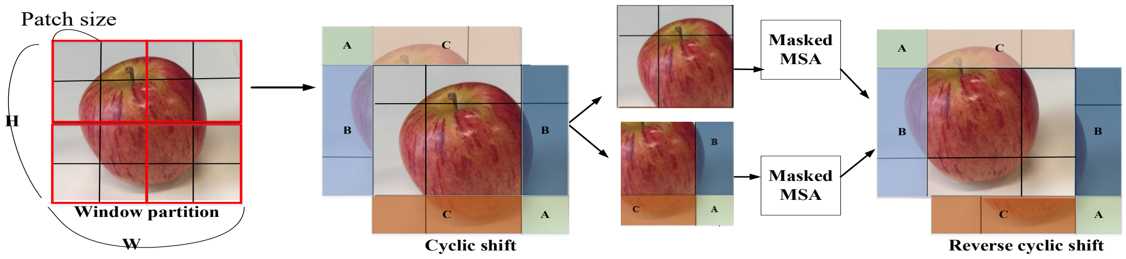


Figure 3.20 The shift window dependent on MSA in Swin Transformer

Mask R-CNN represents a two-stage detection model. Akin to Faster R-CNN, Mask R-CNN generates region proposals. Subsequently, the Mask R-CNN model performs classification, produces bounding boxes, and generates masks for the proposals. To mitigate the degradation of the training process, Mask R-CNN incorporates FPN.

The FPN architecture, characterized by a top-down structure and lateral connections, serves to facilitate the fusion of feature maps across different levels, ranging from the lowest to the highest. This design choice effectively enables rapid connectivity and the extraction of information across various scales. The process of feature extraction takes place in the backbone network, with the resulting feature map subsequently being inputted into RPN for the purpose of selecting sub-networks. Functioning akin to a sliding window with a fixed size, FPN operates in this manner.

To enhance the transfer of features, the Swin Transformer incorporates a 3×3 shift window in addition to the pre-existing 2×2 window, thereby allowing for the intersection of upper window partitions in each block. Considering the differing sizes of these shift windows, a shift operation is employed. As depicted in Figure 3.19, cyclic shifting is utilized to modify the window size from 3×3 to 2×2 during the initial stage. Subsequently, a reverse cyclic shift is applied based on the attention model, resulting in the generation of the shift window attention. This process involves shifting the 3×3 window feature map to align with a 2×2 window, while the actual computations continue to be performed in the context of 3×3 windows. Therefore, the results of 9 attentions are computed with the assistance of masks.

The self-attention mechanism of the Transformer model is equipped with a dedicated mask, which restricts the computation to the relevant portion in a given window, while masking out the remaining areas. The original approach to attention calculation can be customized according to specific requirements. As depicted in Figure 3.20, the shaded region B represents the portion that necessitates masking.

The Window-based local self-attention (W-MSA) partitions the input image into distinct non-overlapping windows and performs separate self-attention computations in each window. If an image consists of $h \times w$ patches, with each window containing $M \times M$ regions, then the computational complexities for both the MSA and W-MSA can be expressed as follows:

$$\Omega(MSA) = 4whC^2 + 2(hw)^2C \tag{3.33}$$

and

$$\Omega(W - MSA) = 4whC^2 + 2M^2hwC \tag{3.34}$$

3.5.2 Detection Transformer (DETR)

The Detection Transformer (DETR) has been employed for tasks such as visual object detection and panoramic segmentation. It distinguishes itself as a pioneering object detection framework that seamlessly integrates the Transformer as the core component of the detection model. In contrast to previous detection approaches, DETR effectively eliminates the need for manually crafted elements such as non-maximum suppression algorithms and anchor point generation.

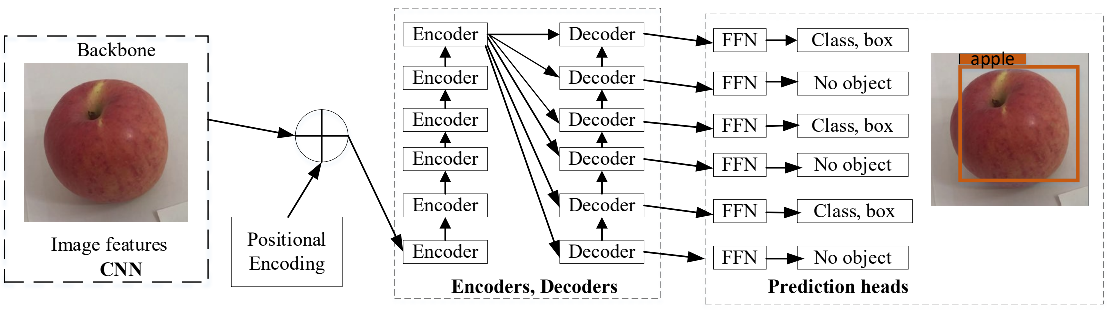


Figure 3.21 The structure of Detection Transformer

The Detection Transformer model accepts a 3-channel RGB image as input, with a CNN backbone responsible for extracting features. To obtain the detection results, these features are combined with positional data and fed into the encoder and decoder components of the Transformer model. Each output is displayed as a bounding box, with each box being a tuple that comprises the class of the fruits and the coordinates specifying the placement of the bounding box.

3.5.3 Vision Transformer (ViT)

The utilization of attention mechanism is pervasive across various domains, including attention modules such as the Se module and CBAM module, which have demonstrated their efficacy in enhancing network performance. Transfer learning proves to be fitting for the ViT model, as it showcases the potential to achieve commendable image classification results without resorting to conventional CNNs. In the ViT model, the original image is segmented into blocks and subsequently fed into the encoder of the conventional Transformer model. Finally, image classification is executed through the utilization of a fully connected layer.

Figure 3.22 illustrates the fundamental constituents of the ViT model, which primarily comprise three components: 1) Linear projection (the embedding layer for patches and their positional information); 2) Transformer encoder (responsible for processing the patches); 3) MLP head (the classification layer).

In the ViT framework, each individual image block is treated as a token, akin to words in NLP. The model computes correlations between these tokens to facilitate the execution of detection tasks.

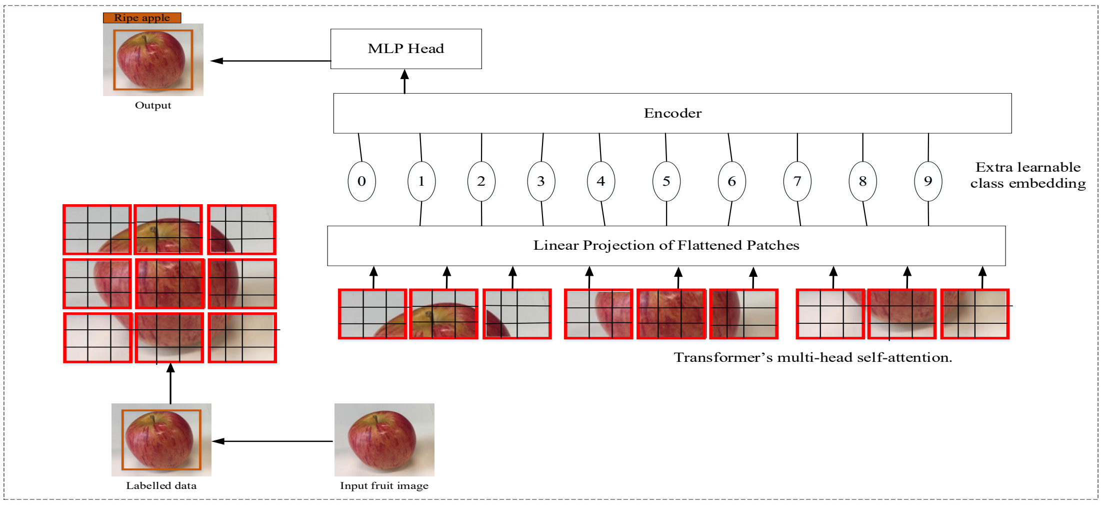

Ripe apple
Output
MLP Head
Encoder
0
1
2
3
4
5
6
7
8
9
Extra learnable class embedding
Linear Projection of Flattened Patches
Transformer's multi-head self-attention.
Labelled data
Input fruit image

Figure 3.22 Vision Transformer flowchart

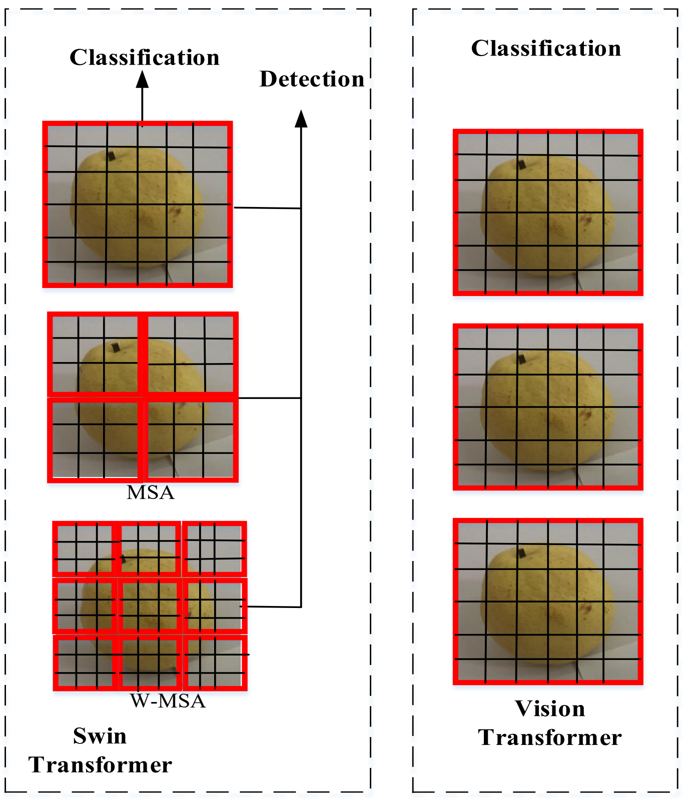


Figure 3.23 Self-attention of regions

The purpose of relative encoding in the Swin Transformer, as illustrated in Figure 3.23, is to tackle the challenge of self-attention organization invariance. The encoding module ensures that input tokens (irrespective of their order), yield consistent results. Merely segmenting the image into patches, as done in the ViT model, proves to be inadequate. In the encoder module, the dimensions of the vector are $[num_{token}, token_{dim}]$. However, the initial shape of the image data, denoted as $[H, W, C]$, fails to conform to the input requirements. Therefore, the encoder assumes a crucial role in transforming the image into tokens through the process of embedding.

Similar to the Swin Transformer, the Vision Transformer also generates $Q, K \text{ } and \text{ } V$. $Q, K \text{ } and \text{ } V$ are actually patches obtained by embedding the context or image using the model. W_q, W_k and W_v are obtained by multiplying and mapping the three matrices that need to be learned. The entire Attention process can be divided into four steps:

1) The patches obtained from embedded operation are subjected to matrix multiplication with learnable matrices W_q , W_k and W_v to generate Q,

K and V.

2) Q and K are used to compute similarity scores S.

3) S is processed through a softmax function to calculate probabilities representing candidate words P. In the Scaled Dot-Product Attention variant, S is divided by a scaling factor before applying softmax function.

4) The attention is derived by performing a weighted sum of P.

The Transformer encoder module comprises multiple stacked encoder blocks and is primarily composed of the following components:

1) Layer normalization

The Transformer encoder module consists of multiple stacked encoder blocks and primarily comprises the following components:

1) Begin by defining the input encoding matrix.

2) Apply layer normalization to this matrix to obtain a normalized version.

3) Utilize this matrix to execute a linear transformation, resulting in another transformed matrix.

2) Multi-head attention

$$MultiHead(Q,K,V) = Attention(QW_i^Q,\ KW_i^K,\ VW_i^V) \tag{3.35}$$

In Figure 3.23, the Vision Transformer employs self-attention to illustrate the connections between individual patches and other patches.

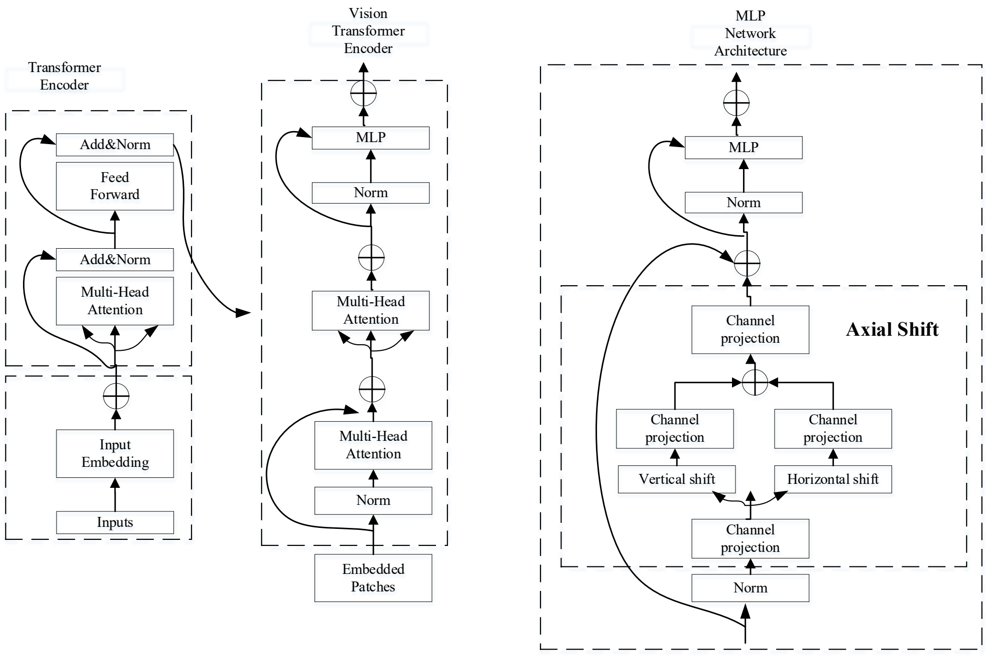


Figure 3.24 Transformer and MLP blocks

The ViT generates query, key, and value vectors into multiple heads (num_heads) and subsequently performs separate self-attention operations on each head. The results from all heads are then merged together. The utilization of multi-head self-attention helps isolate parameters and enables more effective focus on related features during training.

3) MLP block

Calculating the cross-attention value between the encoder and decoder block is unnecessary in the ViT since there is no decoder module. The shape of the output and input remains unchanged after passing through the Transformer Encoder. Before the MLP Head, ViT first extracts $class\,tokens$ for classification from the feature sequences ($tokens$). Then, the MLP Head outputs the final prediction result.

While both the ViT and MLP models apply the MLP block in Figure 3.24, the processes of the two models differ.

3.6 MLP

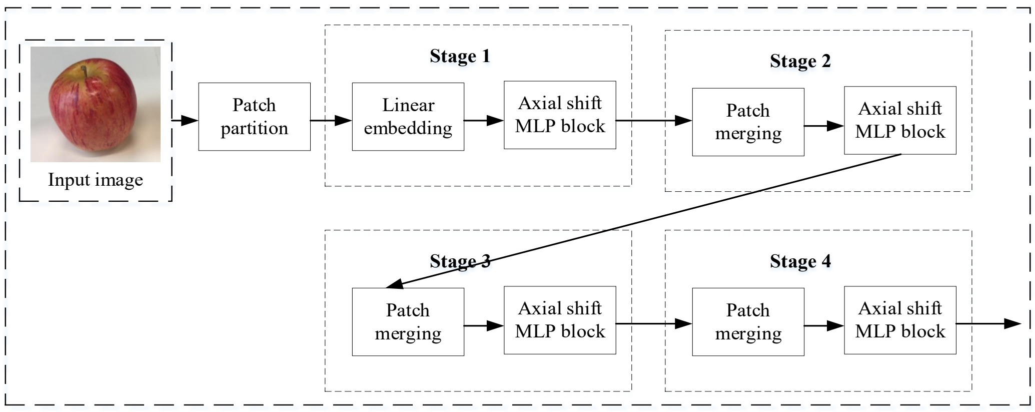


Figure 3.25 MLP Horizontal displacement

From Figures 3.20, 3.22, and 3.25, we can observe the similarities between the MLP model and the Transformer series models. Both models segment the detected images into patches for training.

MLP is a neural network that employs multilayer perceptron for the purpose of feature segmentation and transmission. Comprising an input layer, multiple hidden layers, and an output layer, the MLP neural network combines multilayer perceptron. Each neuron in the MLP is interconnected with the input and output layers, thereby exhibiting similarity among the neurons. Additionally, each neuron transmits the same features to multiple connected output neurons.

MLP networks possess a minimal inductive bias, rendering them effective in tasks such as visual object detection. The structural diagram depicted in Figure 3.26 illustrates the process of axial displacement for MLP-based object detection.

In Figure 3.26, the MLP divides the image into patches, facilitating the extraction of local information by the model.

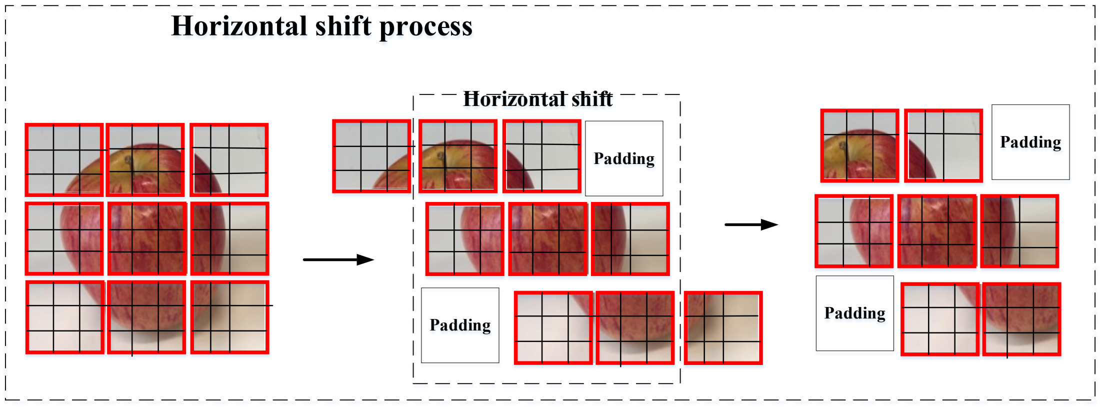


Figure 3.26 MLP Horizontal shift process

3.7 CenterNet

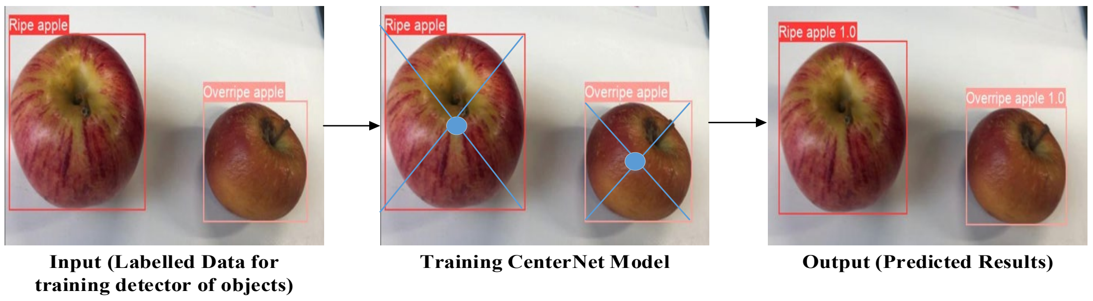


Figure 3.27 The flowchart of CenterNet detection

As the name suggests, the CenterNet model returns the characteristics of the detected target based on its center point. In Figure 3.27, the bounding box with the apple as the center point is detected. The CenterNet model does not require the extraction of RoI, but rather directly feeds images into the model for training. It exemplifies a typical end-to-end model.

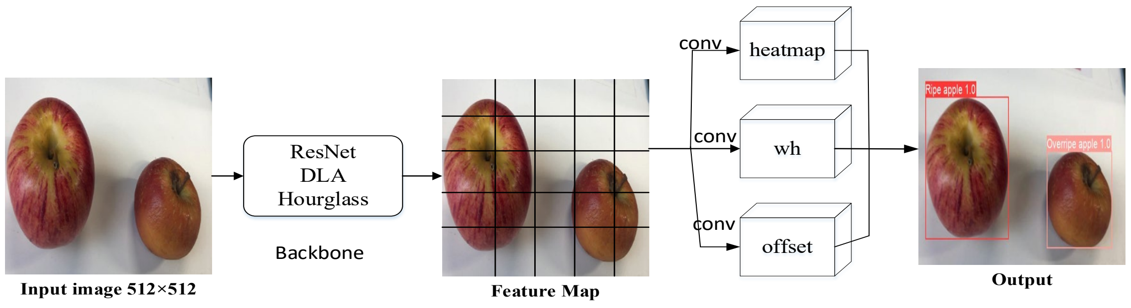


Figure 3.28 CenterNet model

CenterNet detects the target as a point, utilizing the center point of the target box to represent the object. It predicts the offset, width, and height of the target's center point to obtain the actual object box. The heatmap, on the other hand, represents classification information, with each class having its own heatmap. If a center point of an object exists at the coordinates of a particular point on the heatmap, a key point (represented by a Gaussian circle) is generated at that coordinate.

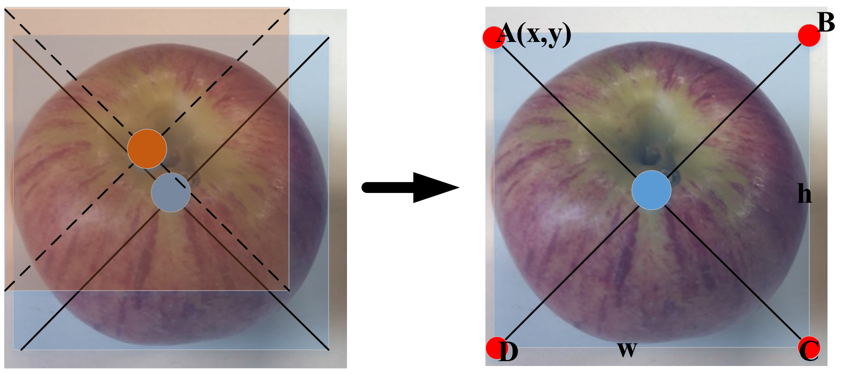


Figure 3.29 CenterNet bounding box

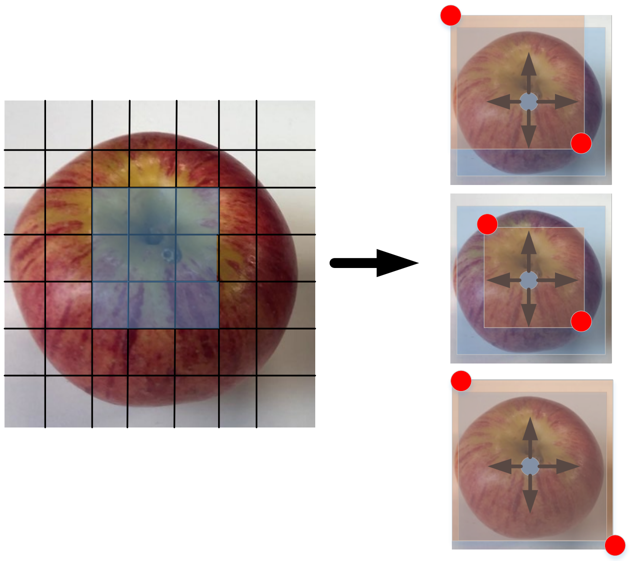


Figure 3.30 Heatmap of the ground truth box

In Figure 3.29, when the network predicts the central point of the heatmap, it does not simply assign a value of 1 to that point and 0 to the other points. Instead, it employs a Gaussian distribution, resembling the shape of a mountain peak. As long as the predicted central point is within the area defined by this Gaussian distribution, it is considered a valid prediction. The blue points in the figure represent the heatmap of a ground truth box, while the orange points can also surround the apple. Moving to Figure 3.30, if the predicted corners area with radius *r* of the top-left or bottom-right point (orange point), the output label of the orange point is not 0, indicating the presence of a detected target. CenterNet utilizes a Gaussian scattering kernel to exclude orange center points with lower probability and retain the blue center points.

The prediction module of CenterNet consists of the prediction of the center point heatmap, offset prediction, and object size prediction. L_{dat} indicates the loss function of CenterNet. L_k displays the centre point of the heatmap. L_{off} expresses offset of the loss of centre point. L_{size} represents the loss of width and hight. Y_{xyc} serves as the ground truth input. The prediction loss function is defined in eq. (3.36).

$$L_{dat}=L_k+\lambda_{size}L_{size}+\lambda_{off}L_{off}\ (\lambda_{size}=0.1, \lambda_{off}=1) \tag{3.36}$$

$$L_K=\frac{-1}{N}\sum_{xyc}\begin{cases}(1-\check{Y}_{xyc})^{\alpha}\log(\check{Y}_{xyc}), \text{ if } Y_{xyc}=1\\(1-Y_{xyc})^{\beta}\log(1-\check{Y}_{xyc}), \text{ otherwise}\end{cases} \tag{3.37}$$

The enhancement of the heatmap loss function, as depicted in eq. (3.37), has been achieved through the incorporation of elements derived from the focal loss. I In this context, α and β serve to strike a balance between the treatment of arduous and straightforward samples. Additionally, the variable N signifies the count of key points, thereby signifying its significance.

It is worth noting that the output of the feature map possesses a resolution that is one-fourth the size of the input image, thereby potentially introducing a substantial margin of error. Therefore, the offset center point loss function, as expounded in eq. (3.38), employs the L_1 loss function to effectively compute the offset loss for positive sample blocks. In this equation, $\hat{O}_{\tilde{p}}$ represents the network's predicted offset value, p denotes the coordinates of the image centre point, R signifies the scaling factor of the heatmap, and $\tilde{p}$ indicates the approximate integer coordinates of the centre point after scaling. It is crucial to acknowledge that each pixel on the output feature map corresponds to a 4×4 region of the original image.

$$L_{off}=\frac{1}{N}\sum_{p}\left|\hat{O}_{\tilde{p}}-\left(\frac{p}{R}-\tilde{p}\right)\right| \tag{3.38}$$

$$L_{size}=\frac{1}{N}\sum_{k=1}^{N}\left|\hat{S}_{pk}-s_k\right| \tag{3.39}$$

The loss functions pertaining to width and height are presented in eq. (3.39). where N denotes the number of key points, s_k represents the actual size of the target, $\hat{S}_{pk}$ indicates the predicted size. The entire computation is performed using the $L1$ loss function.

To enhance the precision of the model and decrease detection speed, non-maximum suppression is omitted by CenterNet. CenterNet employs ResNet as its underlying architecture. The residual network possesses the advantage of variable convolution kernels, which augment the overall scale and effectively diminish the number of channels and computational workload.

3.8 CornerNet

CornerNet is also a model predicated on key point detection. CornerNet utilizes two symmetrical points, situated in the upper left and lower right corners, as key points. Subsequently, corner pooling is employed as a pooling layer to process the identified targets.

CornerNet leverages the Hourglass network as its foundational structure, in conjunction with two prediction models to forecast the upper-left and lower-right corners of bounding boxes individually. In CornerNet, each heatmap channel essentially functions as a binary mask, responsible for predicting the precise location of key point pairs associated with objects. In addition, each corner point generates a distance prediction relative to other corner points, thereby forming an embeddings vector. Similar embeddings indicate a closer proximity between the corresponding corners. In instances where a box is relatively diminutive, an anchor box is generated, leading to the final predicted outcome.

3.9 ConvNeXt & ResNet

All ConvNeXt models are pre-existing structures and methodologies, with a transformation process akin to that of constructing Transformers. The starting point of ConvNeXt stems from ResNet, which employs refined training methodologies to enhance the performance of the ResNet-50 model. The ConvNeXt network architecture reduces several components, including macro design, ResNeXt, inverted bottleneck, large kernel size, and various micro designs, with layers serving as the fundamental building blocks.

The ConvNeXt network exhibits the capability to modify the stacking times of each stage of ResNet, transitioning from the original configuration of (3, 4, 6, 3) to (3, 3, 9, 3). This adjustment, while enhancing accuracy, necessitates an increase in computational scales. In Swin Transformer net, the stem layer is comprised of a convolutional layer with a convolution kernel size of 4 and a stride of 4. Conversely, the stem layer of ResNet50 includes a convolutional layer with a kernel size of 7 and a stride of 2, accompanied by a maximum pooling layer with a kernel size of 3 and a stride of 2. ConvNeXt, as a combination of Transformer networks and ResNet models, replaces the stem layer with the identical convolution layer found in the Swin Transformer network, featuring a convolution kernel size of 4 and a stride of 4. This substitution yields a marginal improvement in accuracy.

In contrast to the conventional ResNet architecture, the ResNeXt network has achieved a harmonious equilibrium between computational operations (FLOPs) and accuracy.

ResNeXt employs group-wise convolution within the convolution blocks, thereby establishing a parallel structure within the block. As the ResNet network expands in size, the block widens at both ends while narrowing in the middle.

Figure 3.31 illustrates the utilization of depth-wise convolution in the ConvNeXt network's construction of its convolution block. Depth-wise convolution effectively reduces the number of parameters, albeit at the expense of a fraction of accuracy.

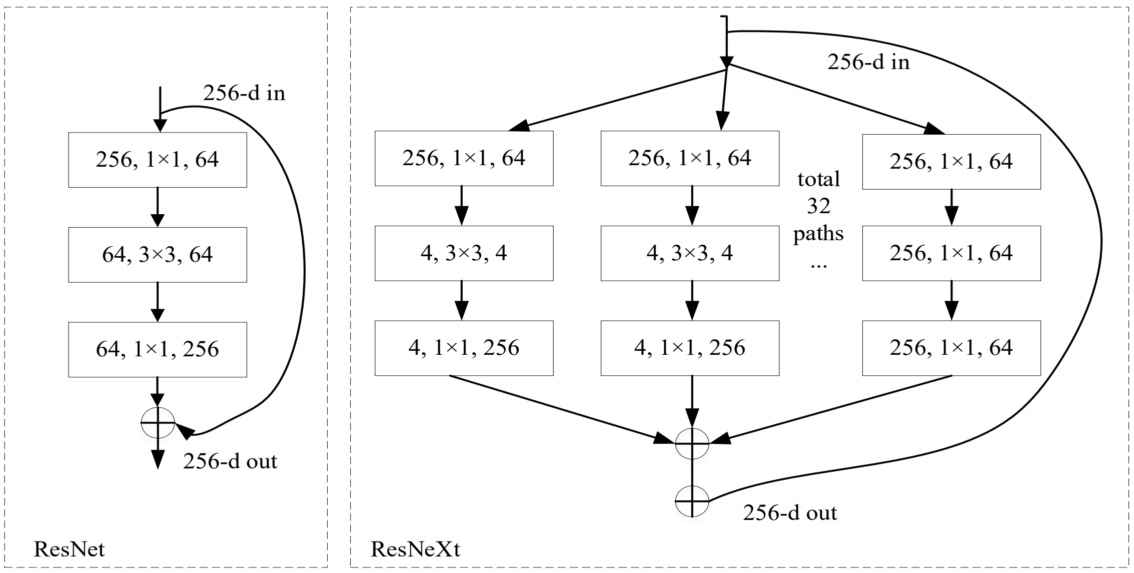


Figure 3.31 ResNet and ResNeXt blocks

In Figure 3.31, the stem layer of Swin Transformer network exhibits an output feature channel count of 96, whereas the ResNet network's stem layer output is limited to a mere 64 dimensions. To ensure conformity with the Swin Transformer, the ConvNeXt network augments the output dimensions to align with those of the Swin-T network. This augmentation significantly enhances the network's accuracy, albeit at the expense of an inevitable escalation in model parameter scale.

The ConvNeXt network designs a similar inverted bottleneck structure, which serves to partially diminish the model's parameter size while concurrently bolstering its overall performance, albeit with a slight improvement in accuracy.

In the ConvNeXt network, the size of convolution kernel is modified from 3×3 to 7×7, mirroring the Swin Transformer's approach, thereby optimizing accuracy. Presently, the prevailing convolutional neural networks predominantly employ a 3×3 window size. However, such a window size yields a smaller receptive field. By utilizing a larger convolution kernel, ConvNeXt effectively expands the receptive field, thereby enabling the acquisition of a greater extent of information to a certain degree.

ConvNeXt replaces the conventional activation function ReLu with GELU and employs a reduced number of activation functions. An activation function or fully connected layer is typically appended to a convolutional neural network following each convolutional layer. However, ConvNeXt does not consistently incorporate an activation function subsequent to a module. Simultaneously, ConvNeXt utilizes a reduced amount of Normalization. The normalization layer in the ConvNeXt block solely retains the normalization layer subsequent to the depthwise convolution. Batch Normalization (BN) expedites network convergence and mitigates overfitting in the convolutional neural network. The downsampling operation of ResNet is executed at the onset of each stage through a 3×3 convolution with a stride of 2 and a 1×1 convolution with a direct stride of 2. Conversely, ConvNext performs independent downsampling between distinct stages, employing a 2×2 convolution with a stride of 2 for spatial downsampling. This modification engenders unstable training; hence, a layer-based normalization is appended prior to the downsampling operation, subsequent to the Stem operation, and the global pooling layer to stabilize the training process.

3.10 Data Augmentation

Data augmentation is a conventional method employed to artificially expand datasets by generating novel data images from existing data. Its primary objective is to enlarge the training dataset, enhance its diversity, and improve the generalization capability of the trained model. Notably, existing models such as Transformer and YOLO already possess their own data augmentation techniques, which can be readily invoked through corresponding interface functions. These techniques involve perturbing the data in various directions or utilizing deep learning methods to generate new data images in the latent space of the original data, thereby artificially augmenting the dataset.

It is important to differentiate augmented data from synthetic data. Synthetic data refers to artificially generated data that does not rely on real-world images. GAN or Diffusion Model can be employed to generate synthetic data. On the other hand, augmented data is derived from the original images and undergoes minor geometric transformations (e.g., flipping, translating, rotating, or adding noise) or color transformations (e.g., adjusting brightness, contrast, saturation, or shuffling channels).

These transformations serve to increase the diversity of the training set. In practical applications, datasets may suffer from data loss and image blurring (due to environmental conditions and other factors) during the collection, cleaning, and labeling processes. By leveraging data augmentation techniques, the issue of the model's inability to learn features from abnormal scenes in the data is addressed, thereby enhancing the model's performance.

CNN exhibits invariance to size, translation, lighting, and viewpoint, rendering them capable of accurately classifying objects across various orientations. In deep learning, CNN excels human vision by conducting convolution operations on input images to acquire diverse features embedded within. In light of the introduction of ViT, a series of models have gained widespread adoption. Nonetheless, the efficacy of both CNNs and Transformers hinges upon the availability of data. Particularly, when confronted with limited data, CNN is susceptible to overfitting, while Transformers struggle to acquire optimal representations.

Vision Transformers have emerged as the dominant approach across natural language processing (NLP) tasks. ViTs have demonstrated remarkable advantages in computer vision tasks, particularly in image classification. This methodology combines local feature extraction with global representation, wherein the ViT model establishes a network architecture comprising a branch dedicated to convolutional neural networks, a visual converter branch, and a startup module. This dual-network structure effectively preserves and enhances both local features and global representation in a targeted manner.

For the dataset pertaining to visual object detection, when the sample size becomes excessively large, the training of the model becomes redundant, thereby consuming a substantial amount of time and resources. However, in the general scenario or when confronted with an insufficient number of samples, the detection process is susceptible to overfitting due to the inadequacy of training data. The influence of background images on feature extraction is often overlooked during the collection of datasets, resulting in the attainment of excessively large parameters for training purposes. Data augmentation serves as a potential solution to address the extent to which the pixels in the background of an image impact the label space.

To mitigate the issue of overfitting during the training of deep networks, the mixup data augmentation module is introduced to enhance the learning ability of the proposed model. Mixup represents a successful technique for image blending, wherein an augmented image is obtained through the pixel-weighted combination of two global images. The subsequent variants of Mixup can be categorized as follows:

1) Global image mixing, such as: ManifoldMixup and Un-Mix.

2) Regional image mixing, such as: CutMix, Puzzle-Mix, Attentive-CutMix and Saliency-Mix.

The utilization of the mixup data augmentation method facilitates the redistribution of ground truth labels in images, thereby enabling the attainment of more precise classification labels through the guidance of attention. Mixup constitutes an algorithm employed in computer vision to blend and enhance images. By blending distinct images, mixup effectively expands the training dataset.

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j \tag{3.40}$$

$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j \tag{3.41}$$

$$\lambda = Beta(\alpha, \beta) \tag{3.42}$$

$$img_{mix} = img1 \times \lambda + img2 \times (1 - \lambda) \tag{3.43}$$

where (x_i, y_i), (x_j, y_j) represent two randomly selected samples and their corresponding labels. Among them are image x_i (labelled y_i) and image x_j (labelled y_j), while λ denotes a number randomly sampled from the beta distribution.

Among them, λ represents the mixing coefficient calculated by the beta distribution whose parameters are α and β. The img_1 and img_2 denote Mixup data augmentation samples. Irrespective of the chosen values for α and β, it is expected that $\frac{\alpha}{\alpha+\beta}$ value constantly approximates to 0.5.

Mixup data augmentation is predicated on the random numbers sampled from the beta distribution, which are then utilized to blend the images in accordance with the ratio of the random numbers. Simultaneously, the labels are also mixed in proportion to their corresponding ratios. This process yields the mixed new images and labels.

3.11 Activation Function

Vanishing and explosion are classical challenges that cause learning instability in deep learning models. As the neural network grows deeper, the stability of the model tends to deteriorate.

The bias parameter is not taken into consideration, and the activation functions of all hidden layers are set to be identity maps.

$$(Identify\ mapping)\varphi(x) = x \tag{3.44}$$

Given an input *X*, the output of layer l of a multilayer perceptron.

$$H(I) = XW(1)W(2)\ldots W(I) \tag{3.45}$$

At this time, if the number of layers, denoted as l, is substantial, the computation of *H*(*I*) may experience attenuation or explosion. Similarly, when the number of layers is large, the calculation of gradients is more susceptible to attenuation or explosion.

The multiplicative effects arising from excessively deep network layers during gradient backpropagation are the root causes of gradient explosion and gradient disappearance, which are fundamentally synonymous phenomena.

The weights of the hidden layer adjacent to the input layer exhibit a sluggish or static update pattern due to the absence of the gradient, thereby resulting in a training process that can only be likened to the learning capabilities of the subsequent few layers of the shallow network.

The occurrence of gradient explosion is a frequent phenomenon when the deep network and weight initialization parameters are excessively high. This explosion of gradients leads to network instability, rendering it incapable of updating weight values and thus impeding the acquisition of knowledge from the training data.

In deep learning networks, owing to the large number of network layers and the considerable discrepancy in the speed of learning across these layers, the network is able to acquire a greater number of features in proximity to the output layer, while the learning process near the input layer proceeds at a markedly slower pace. As the network grows deeper, prolonged training leads to weight values that closely resemble their initialization counterparts. The input information undergoes layer-by-layer processing through backpropagation, finally reaching the output layer. This sequential transmission of information culminates in the gradient of the objective function with respect to the weight vector, thereby giving rise to the phenomenon of gradient disappearance and explosion.

Upon training the model, a chain of models is derived. The length of this chain increases as one approaches the input layer, resulting in a greater computational burden and, therefore, a slower processing time. Simultaneously, the derivative problem associated with the nonlinear function Sigmoid is prone to inducing gradient disappearance.

Taken the simplest network structure depicted in Figure 3.32 as an illustration, three hidden layers are added, the number of neurons in each layer is 1. Among them, C is the loss function; the input of each layer is z, and the output is a, where $z = w \times a + b$ (w can also be compared).

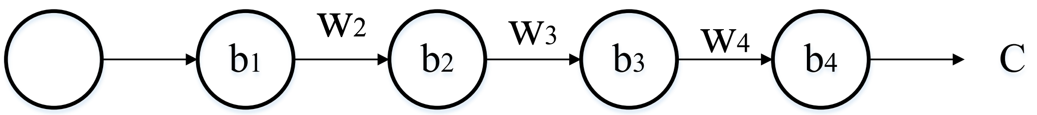


Figure 3.32 An example of artificial neural networks

Suppose the objective is to update the parameter b_1; in such a case, it is necessary to identify the derivative of the loss function with respect to b1. By virtue of the chain rule, the corresponding formula is presented in eq. (42).

$$Sigmoid(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \tag{3.46}$$

$$Sigmoid^{'}(x) = Sigmoid(x)(1 - Sigmoid(x)) \tag{3.47}$$

Upon deriving the Sigmoid function, as indicated in eq. (43), it is observed that the sigmoid derivative attains a maximum value of 0.25 at 0. As the number of network layers deepens, the resultant product experiences an exponential decay, thereby leading to gradient dispersion.

Both gradient disappearance and gradient explosion are observed in the presence of ReLU. As the number of network layers increases, the activations tend to gravitate towards larger and smaller values. During the gradient calculation process, each layer of backpropagation necessitates multiplication by the activations pertaining to that specific layer. Therefore, the deep gradient of the network layer undergoes an explosive growth, while the shallow gradient of the network layer reduces. The impacts of these two issues are most pronounced in the form of the arduous convergence of deep networks.

The utilization of the unsupervised layer-by-layer training method entails the utilization of the output derived from the hidden node of the preceding layer as input during the training process. This input is then employed as the input for the hidden node of the subsequent layer, thereby facilitating a layer-by-layer "pre-training" procedure. Following the completion of the pre-training phase, the entire network undergoes a process of "fine-tuning" to address the gradient predicament that arises due to the network's depth. Additionally, the issue of exploding gradients can be mitigated through the implementation of gradient clipping. By establishing a threshold for gradient clipping, the gradient is constrained in a predetermined range during the model training process. In addition, weight regularization is employed through the utilization of L_1 and L_2 regularization functions. When the network invokes the layer, the regularization loss is directly computed, thereby mitigating the problem of overfitting caused by gradient explosion.

The activation function, which operates on the neurons in a neural network, assumes the responsibility of mapping the input of a neuron to its corresponding output. The ReLU activation function, characterized by a derivative of 1.0, effectively eliminates the issue of gradient disappearance explosion, ensuring that each layer of the network undergoes updates at a consistent pace. Therefore, the ReLU function resolves the problems associated with gradient disappearance and explosion, expediting the network's training process and facilitating more efficient network computations. However, due to the derivative being 1, the negative portion of the function is consistently evaluated as 0, thereby impeding the transmission of certain neuron features.

ELUs, or exponential linear units, are activation functions that aim to expedite the learning process by reducing the mean value of the activation function towards zero. Moreover, by identifying positive values, ELUs can effectively mitigate the issue of gradient fading. Empirical studies have demonstrated that ELUs exhibit superior classification accuracy compared to ReLUs. However, it is worth noting that ELUs require more computational time for evaluation when compared to Leaky ReLUs.

The Leaky ReLU function serves as a viable solution to the limitations of the ReLU function interval, while including all the advantages associated with ReLUs. The ReLU function sets all negative values to zero, whereas the Leaky ReLU assigns a non-zero gradient or slope to all negative values in eq. (3.48).

$$y_i = \begin{cases} x_i, if\ x_i \geq 0 \\ \frac{x_i}{a_i}, if\ x_i < 0 \end{cases} \tag{3.48}$$

In the Leaky ReLU function, the parameter a_i remains fixed in the interval $(1, +\infty)$.

A variant of Leaky ReLU, known as RReLU, introduces a randomization element to the slope of negative values during the training process.

During training, the value of a_{ji} is randomly drawn from a uniform distribution U.

$$y_{ji} = \begin{cases} x_{ji},\ if\ x_{ji} \geq 0 \\ a_{ji}x_{ji},\ if\ x_{ji} < 0 \end{cases} \tag{3.49}$$

where $a_{ji} \sim U(l, u), l < u \; and \; l, u \in [0,1)$.

The difference between ReLu and Sigmoid:

(1) The sigmoid function yields values in the range of [0,1], while the ReLU function yields values in the range of [0,+ ∞].

(2) The gradient of the sigmoid function increases or decreases and eventually vanishes as *x* increases, whereas the ReLU function does not exhibit this behavior.

The residual network depicted in Figure 3.33 incorporates the input into a specific layer, ensuring that during the derivation process, a value of 1 is consistently present to prevent the vanishing gradient problem.

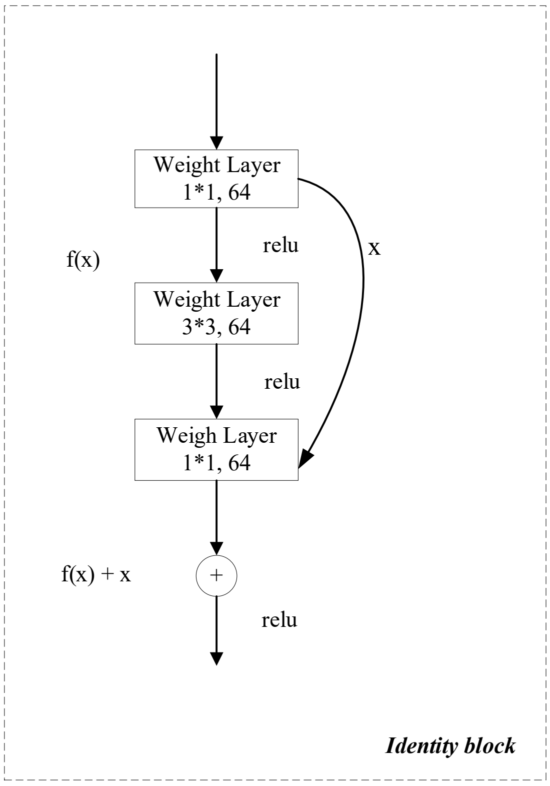


Figure 3.33 Residual network structure

3.12 Summary of This Chapter

A pseudocode is employed to concisely outline the iterative procedure of the YOLO comprehensive model. The complete fruit image is fed into the model to extract feature maps, followed by regression to predict the fruit category.

The pseudocode for fruit detection in digital images essentially includes a loop-based process of detection. Irrespective of any changes made to the model, fruit detection necessitates a sequence of operations, involving dataset processing, detector training, predicted box generation, and model evaluation.

Based on our analysis of the preceding model, it is necessary to ensure swift detection when applying the object detection model to fruit detection. Both the YOLO series and the Transformer model employ a series of modules to enhance the capacity to transfer detailed features and reduce computational requirements. The fifth chapter of this thesis can provide a more intuitive analysis of how the model operates in the context of fruit detection.

```
Input: An apple image
Initialization;
for $x$ in interval (0, width)
for $y$ in interval (0, height)
For $w_c$ in interval (1, 2000)
For $h_c$ in interval (1, 2000)
        For a feature map $img_c$
        $img_c$ = $img[x-w_c/2:x+w_c/2, y-w_c/2:y+w_c/2]$;
        confidence = classifier ($img_c$)
        If class == "ripe apple" $detect_{result}$ ([confidence])
        Endfor
Endfor
Endfor
Endfor
Endfor
Output: Apple detection result.
```

Figure 3.34 Pseudocode of the algorithm for fruit detection

To commence, a comprehensive explanation of each constituent of the process is provided prior to delving into the crux of the fruit identification experiment. The technique employed for data collection, the methodologies employed for annotation, and the selection of pertinent characteristics for fruit detection are all expounded upon in detail in this tutorial. We go above and beyond in our efforts to offer a clear and complete understanding of the experimental procedure.

In addition, an exploration into the functionality of transformers for object detection in the era of transformer-based architectures is undertaken. These models harbor the potential to enhance fruit detection, as they have evinced remarkable capability across a multitude of computer vision tasks. The mechanics of transformer-based item identification techniques are described, and a discussion ensues regarding their potential to enhance the precision and efficacy of fruit recognition systems.

Finally, an analysis of regression methodologies in the context of object detection is undertaken. These methodologies are indispensable for refining the localization of identified items and ensuring that the bounding boxes fully encapsulate the instances of fruit. The intricacies of regression approaches are illuminated, underscoring their paramount importance in attaining highly accurate fruit detection results.

In conclusion, a comprehensive understanding of the fruit identification experiment, the underlying object detection paradigms, and the pivotal role of regression approaches is provided in this chapter. This knowledge forms the bedrock upon which subsequent chapters are constructed, facilitating an exploration of the experimental findings and broader ramifications of our work in the field of fruit recognition.

Chapter 4 Experiment Settings

This chapter delves into our experiments comprehensively and their corresponding evaluation methodologies. A thorough investigation of data collection and experimentation is presented, accompanied by the introduction of more efficacious model evaluation techniques. Our work, including not only a description of the experimental content but also the presentation of enhanced model evaluation procedures that enhance the accuracy and comprehensiveness of our study.

4.1Data Collection and Experimental Environment

4.1.1 Data Description and Preprocessing

In our experiments, a dataset was captured. Our 4,000 images are divided into four datasets, classified in accordance with the experimental requisites. Figure 4.1 showcases a total of 20,000 labels for the 4,000 images, revealing a relatively uniform distribution of labels.

In Section 3.2 of this thesis, a previous two-stage model was introduced for object detection. The dataset, denoted as Dataset I, II, and III, underwent augmentation techniques such as rotation and Gaussian blur. During the training process, it was discovered that a substantial number of datasets did not contribute to the model's ability to learn detection features effectively. Instead, these additional datasets introduced redundancy, resulting in model overload. Simultaneously, considering the characteristics of the data augmentation module in the one-stage model; finally, the study selected a dataset, denoted as Dataset IV, consisting of 2000 samples. Notably, the models such as YOLO and Transformer possess the capability to accept arbitrary data inputs, automatically resizing the image dimensions to 640×640.

Table 4.1: Dataset description

Datasets	Category	Number of apple images	Number of apple labels	Number of pear images	Number of pear labels	Image Size
I	Ripe	144	552	----	----	224×224
	Overripe	111	452	----	----	
	Unripe	92	409	----	----	
II	Ripe	1,325	5,416	----	----	
	Overripe	1,083	4,475	----	----	
	Unripe	1,040	3,976	----	----	
III	Ripe	4,149	12,564	----	----	
	Overripe	3,713	10,812	----	----	
IV	Ripe	200	200	200	99	Random resize to 640×640
	Overripe	200	200	200	101	

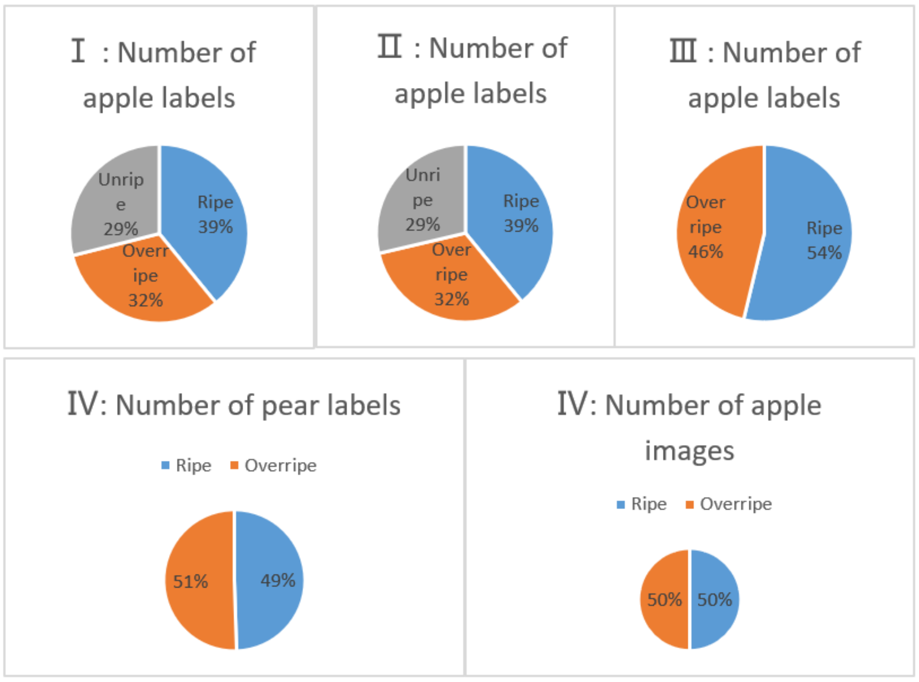


Figure 4.1: Dataset label distribution

Figure 4.1 presents a pie chart illustrating the distribution of the dataset. It is observed that the label classification in the dataset is well-balanced, ensuring that the model can effectively learn the fruit features across all categories.

4.1.2 Original Data & Labelled Data

The ripeness of a fruit is determined by the smoothness of its skin, while various fruits are classified into different classes. The term "Ripe" denotes a fruit with a smooth peel, whereas the term "Overripe" refers to a fruit with a crumpled or decaying surface.

In order to evaluate the learned parameters, we utilized our phone cameras to acquire datasets of apples and pears. Subsequently, a comparative analysis was conducted between the learned parameters and those of authentic models. Figure 4.2 showcases an input image from our acquired dataset.

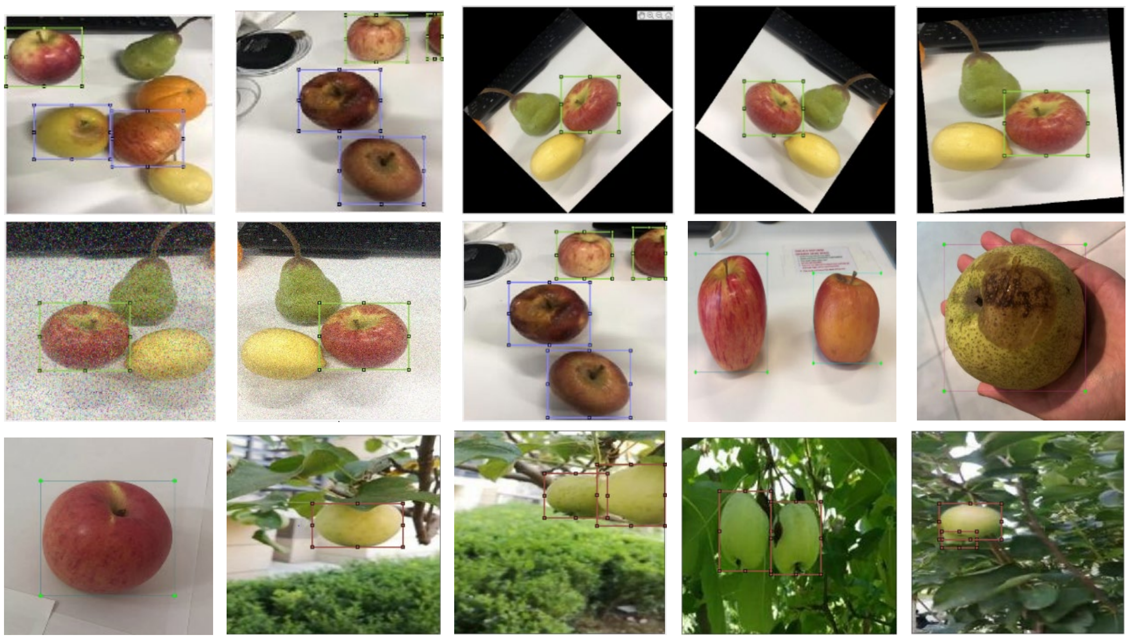
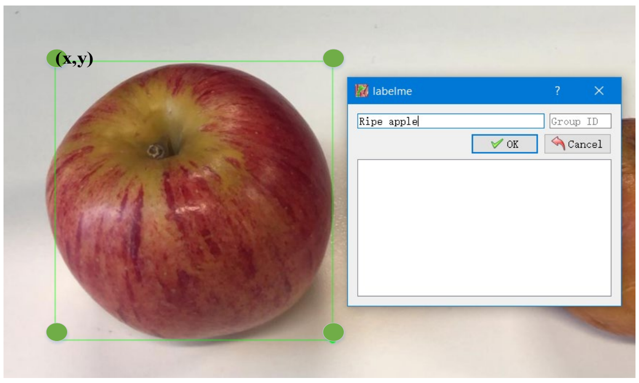
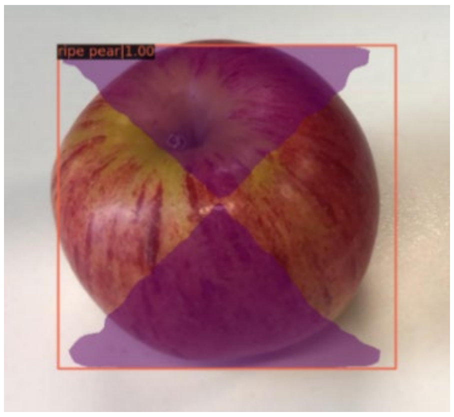


Figure 4.2: Samples of image data in our dataset.



a) A sample of input images.



(b) A sample of MLP methods output images.

Figure 4.3:Image samples of our dataset

The dataset that was labeled is depicted in the image presented on the left-hand side of Figure 4.3. The location and class of apples were manually designated by means of a red bounding box. To create the training dataset for fruit classification, the ground truth and the coordinates for the placements of the bounding boxes were manually annotated onto a collection of photographs.

4.2Experimental Design

In this project, PyTorch and MATLAB were employed as the experimental platforms. The fruit datasets were labeled using the Labelimg tools.

Table 4.2: Training parameters

Parameter	Value
$optimizer$	SGD
$batch$	2
$mask\_ratio$	4
box	7.5
cls	0.5
$weight\_decay$	0.0005

To ensure the utmost quality of our experimental data, explicit images of apples and pears were captured in a well-illuminated environment, as exemplified in the input image. Fruits were categorized based on the characteristics of their skins, which served as an indicator of their maturity. A smooth surface of the fruit was regarded as a sign of maturity, while any apples exhibiting signs of decay were deemed overripe.

The primary objective of our experimentation is to assess various models for fruit detection and identify the most effective one. The number of epochs serves as a critical experimental parameter that is employed for this purpose. Merely running the neural network once with the complete dataset is insufficient. The entire dataset must be traversed multiple times. By varying the epoch value, we ascertain the number of model iterations, thereby enabling us to evaluate the merits and drawbacks of each model. In addition, we set the $weight\_decay$ to 5.00×10^{-2} and adjust the betas in the range of

(0.90, 0.99).

With respect to supervised learning, the study first employed a higher learning rate, which was subsequently reduced in a gradual manner as the number of iterations increased, eventually settling on 0.01. Throughout the training process, the entirety of the data was fed into the neural network, and the gradients were computed. However, due to the significant difference in gradient values, the application of a global learning rate posed a formidable challenge. Therefore, the batch value was set to 2 in order to minimize the potential for a catastrophic increase in RAM usage.

4.3Summary of This Chapter

A comprehensive explanation is provided regarding the modifications made to the variables, the sources of data utilized, and the specific methodologies employed in our research.

Elaborate details are provided regarding certain aspects of the data set division, including the platforms and software employed in the studies, as well as the methodologies employed to collect the data sets. In Chapter 3 of the thesis, we also expound upon how the Faster R-CNN experiment served as a source of inspiration for our current studies, particularly in terms of how data sets influence experiments and models.

The intricacies of data acquisition are navigated, with a comprehensive elucidation of the sources, sampling techniques, and data preprocessing steps that form the foundation of the research being undertaken.

Chapter 5 Result Analysis and Discussions

In this chapter, a thorough analysis of all experimental results is conducted, accompanied by a comparative analysis. The advantages and disadvantages of various methods are assessed through ablation experiments. This chapter comprises the entirety of the compiled experiments and their corresponding outcomes, representing the collective research. To present the findings of the scientific investigation, a combination of graphs and informative tables is employed.

5.1 Results & Analysis

Two distinct variations of ViT model were utilized: $Base$ and $Large$. While $vit\_large\_patch\_32$ signifies the utilization of the ViT large model with a patch size of 32×32, while $vit\_base\_patch\_16$ refers to the ViT base model with a patch size of 16×16. The performance of ViT model is unsatisfactory, as evidenced by the results presented in Table 5.1. It appears that the ViT model encounters challenges when confronted with the trade-off between small and large datasets. Increased iterations do not yield improved outcomes. ViT models are typically employed as pretrained models. In scenarios involving limited datasets, CNNs frequently outperform ViT models.

Table 5.1: ViT results

Model	Epoch	Weights	AP@0.5:0.95
Vision Transformer	10	$vit\_base\_patch\_16$	0.4560
		$vit\_base\_patch\_32$	0.4060
		$vit\_large\_patch\_16$	0.4560
		$vit\_large\_patch\_32$	0.4120
	20	$vit\_base\_patch\_16$	0.4310
		$vit\_base\_patch\_32$	0.4310
		$vit\_large\_patch\_16$	0.3560
		$vit\_large\_patch\_32$	0.4250
	30	$vit\_base\_patch\_16$	0.4000
		$vit\_base\_patch\_32$	0.4190
		$vit\_large\_patch\_16$	0.3880
		$vit\_large\_patch\_32$	0.3880
	50	$vit\_base\_patch\_16$	0.3810
		$vit\_base\_patch\_32$	0.3940
		$vit\_large\_patch\_16$	0.4060
		$vit\_large\_patch\_32$	0.4000

Despite of the relatively modest precision observed in Table 5.1, the ViT model offers notable advantages in terms of efficient resource utilization during the training process. Prior to the final optimization, the ViT model implements structural pruning on the Transformer model, followed by quantization. It is worth noting that the pruning procedure of the ViT model necessitates additional training, thereby imposing certain limitations on its applicability. While ViT model pruning leads to reduced execution time and memory consumption, it does come at the cost of model fidelity.

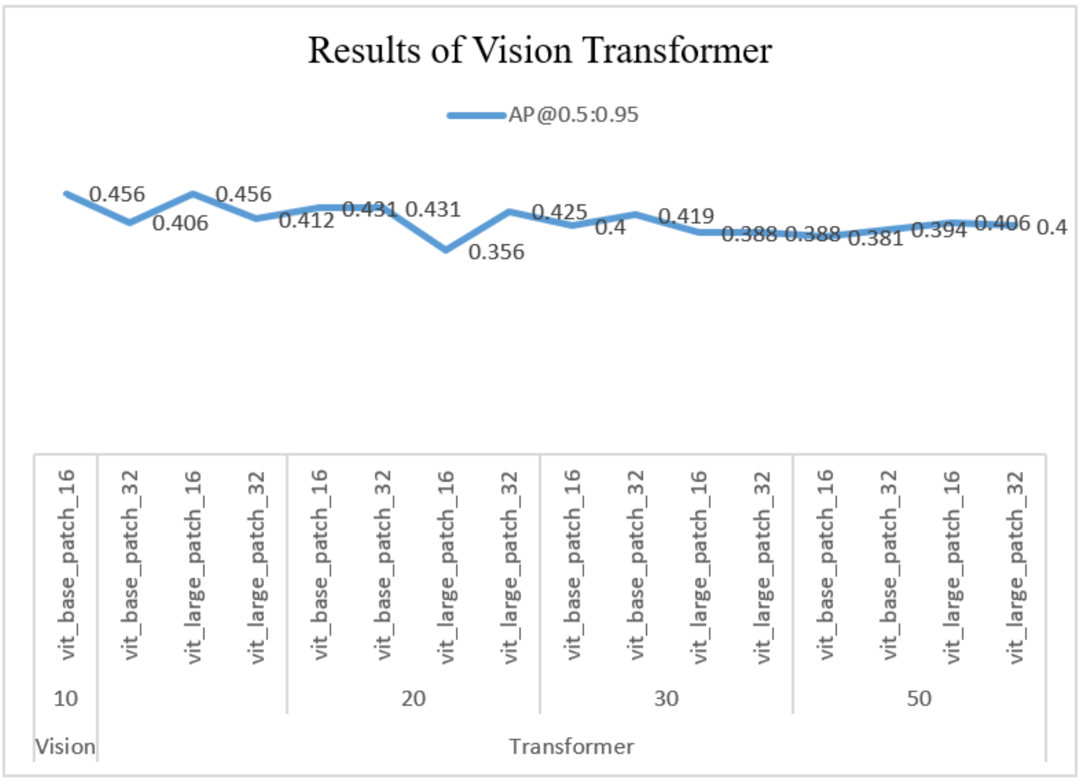


Figure 5.1: The result of ViT transformer

The ViT model progressively acquires both local and global features from its shallow layers. While the fruit detection model did not yield a substantial amount of informative data, the stability of the line chart depicted in Figure 5.1 suggests that ViT's self-attention mechanism effectively facilitates the transfer of learned features.

In Table 5.2, the MLP object detection model demonstrates commendable performance, particularly in the case of smaller models. The MLP model places significant emphasis on local feature extraction. However, as the generalization capacity of this model is significantly augmented, the emergence of overfitting issues becomes apparent. These overfitting issues can impede the extraction of features by the MLP model.

The ViT self-attention mechanism incorporates an MLP block, which operates on sequences, ensuring that each position in the sequence shares the same set of MLP parameters. Subsequently, a weighted averaging operation is performed in a novel space.

The MLP model is regarded as a nonlinear mapping. Tables 5.1 and 5.2 unveil the exceptional ability of the MLP model to capture the inherent characteristics.

Table 5.2: MLP with Mask R-CNN (small) results

Model	Epoch	Weights	AP50	AP@0.5:0.95	Average inference time (seconds)
MLP using Axial shift MLP block	10	*mask_rcnn_small_patch4_1x*	0.9450	0.8310	0.5850
	30		0.9560	0.8430	0.5370
	50		0.9600	0.8470	0.5400
	10	*mask_rcnn_tiny_patch4_1x*	0.9330	0.8270	0.3820
	30		0.9550	0.8440	0.3850
	50		0.9580	0.8440	0.3760

It is evident from Table 5.2 that, in terms of detection, the MLP structure exhibits a reduced inductive bias, enabling it to achieve, and in some cases surpass, the performance of CNN (as demonstrated in the results of Faster R-CNN in Table 3.1) and Vision Transformer.

The characteristics of long-term dependence of MLP filter and weight the feature vector of deep network, highlight the key information, and mitigates information loss attributable to the pooling layer.

The axial displacement strategy of MLP architecture facilitates spatial shifting of features in both horizontal and vertical directions. This strategy aligns features from different spatial locations to the same position, thereby enabling a straightforward and effective combination of these features using an MLP. Therefore, the model acquires more localized dependencies, leading to enhanced performance. Additionally, the axial displacement strategy aids in determining the kernel size and expansion rate of the MLP, akin to a convolution kernel.

The network model under consideration is characterized by an MLP block, which, in the context of the axial displacement strategy, has assimilated a more extensive repertoire of features pertaining to fruits. The MLP block, operated in accordance with the axial shift paradigm, is primarily concerned with the extraction of local features, thereby striving to minimize the inference time associated with the model.

Diverging from the ViT and MLP architectures, the Swin Transformer exhibits a unique self-attention calculation mechanism predicated on a mobile window. This particular approach ensures that the model is capable of capturing a broader spectrum of visual object attributes.

In order to train the Swin Transformer model, we employed three distinct weight configurations. It is worth emphasizing, as outlined in both Table 5.3 and Table 5.4, that a single patch has the capacity to cover four windows following the process of displacement. However, it is not practically feasible for a patch to independently traverse each window. To circumvent this limitation, a mask is employed to establish connections and facilitate attention computation within each window. Therefore, the Swin Transformer model operates as a hierarchical representation, thereby enabling it to effectively perform intricate linear calculations.

Table 5.3: Swin Transformer with Mask R-CNN (small) results

Model	Epoch	Weights	AP50	AP@0.5:0.95
Swin Transformer	10	*mask_rcnn_small_patch4_1x*	0.9390	0.8210
	20		0.9400	0.8230
	30		0.9480	0.8300
	50		0.9510	0.8390
	10	*mask_rcnn_small_patch4_3x*	0.8340	0.6810
	50		0.9460	0.8350

Table 5.4: Swin Transformer with Mask R-CNN (tiny) results

Model	Epoch	Weights	AP50	AP@0.5:0.95
Swin Transformer	10	*mask_rcnn_tiny_patch4_1x*	0.9360	0.8090
	20		0.9380	0.8230
	30		0.9450	0.8220
	50		0.9450	0.8230

Through the shifted windows from the Swin Transformer, self-attention calculations are confined to non-overlapping windows, while cross-window connections are leveraged to enhance computational efficiency. As illustrated in column chart 5.2, the performance of the Swin Transformer model exhibits remarkable stability.

The original image is divided into smaller patches by ViT in order to ensure that the sequence length falls within the acceptable range for the Transformer. In the case of larger targets, ViT can adjust the sequence length by modifying the patch size, thereby enabling the Transformer to handle larger images. Therefore, the Vision Transformer may not be optimal for detecting small objects such as fruit. This observation is evident from the analysis presented in Fig. 5.1 and Fig. 5.2, where it is demonstrated that Swin Transform yields superior results for small target detection.

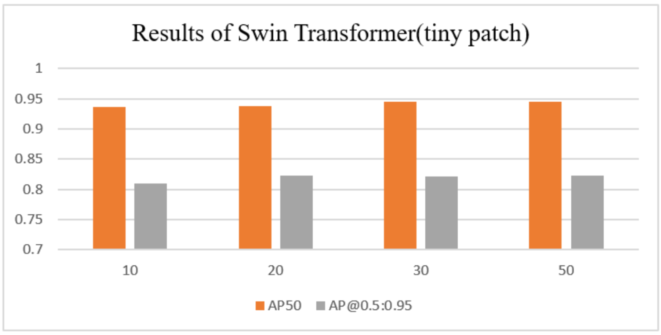


Figure 5.2: Results of Swin Transformer

Table 5.5: Swin Transformer + YOLO with Mask R-CNN

Model	Weights	Epoch	Class	mAP	Average inference time (seconds)
YOLOX + Swin Transformer	*mask_rcnn_small_patch4_3x*	10	Ripe apple	0.0000	0.1217
			Over apple	0.2200	
			Ripe pear	0.0200	
			Overripe pear	0.4600	
		20	Ripe apple	0.0000	0.1210
			Over apple	0.0060	
			Ripe pear	0.0860	
			Overripe pear	0.4340	
		30	Ripe apple	0.7867	0.1205
			Over apple	0.4687	
			Ripe pear	0.8404	
			Overripe pear	0.8416	
		50	Ripe apple	0.8889	0.1212
			Over apple	0.6695	
			Ripe pear	0.8856	
			Overripe pear	0.9127	

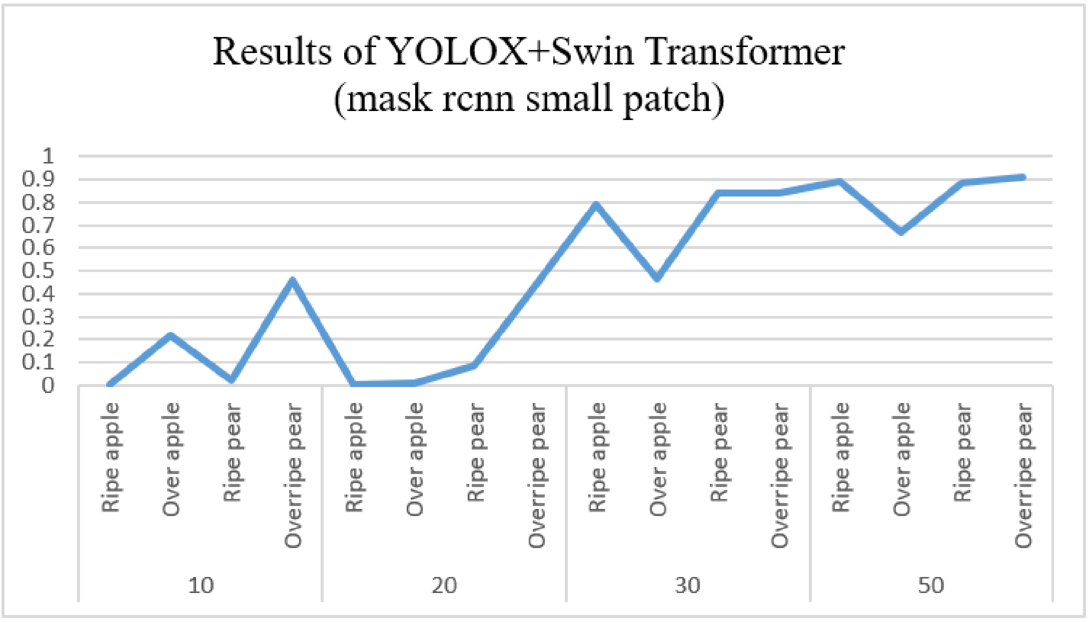


Figure 5.3: The results of YOLOX + Swin Transformer using Mask R-CNN small patch

As an object detection algorithm operating in a single stage, the YOLO model exhibits remarkable flexibility, efficiency, and generalization capabilities. YOLOX introduces several novel techniques, including anchor-free, decoupled, head, mosaic data enhancement, and SimOTA sample matching methods, which are built upon the foundation of YOLOv3. The anchor-free method liberates YOLO from its dependence on anchors and establishes a novel end-to-end framework.

Similar to YOLO, YOLOX is structured into three components: Backbone, Neck, and Head. In YOLOX, the CSODarknet from the YOLO series is employed as the backbone extraction network consisting of Focus, CBS, CSP1, and SPP. The Neck component adopts a combined architecture that incorporates FPN and Pyramids Attention Network (PAN). FPN facilitates the fusion of high-level and low-level information through upsampling, enabling the transfer of features from top to bottom. PAN, on the other hand, facilitates the transfer of information from lower layers to higher layers, thereby achieving integration and transmitting positional information from the bottom to the top.

Table 5.5, Table 5.6, and Table 5.11 are compared in order to assess the performance of different models. Specifically, the utilization of YOLOX in conjunction with the Swin Transformer yields superior detection results compared to the Swin Transformer employing the conventional YOLO module. The incorporation of the FPN+PAN network structure facilitates the improved transmission of object characteristics in the model. Figure 5.3 provides a visual representation of the YOLOX and Swin Transformer's ability to capture detailed features subsequent to training.

Simultaneously, YOLOX replaces the original Head of YOLO with a decoupling head, thereby achieving expedited model convergence. In the context of end-to-end tasks, the YOLO+Swin Transformer architecture proves more conducive to the integration of downstream tasks.

Table 5.6: Swin Transformer with Mask R-CNN (tiny)

Model	Weights	Epoch	Class	mAP	Average inference time (seconds)
YOLOX + Swin Transformer	*mask_rcnn_tiny_patch4_3x*	10	Ripe apple	0.1978	0.1288
			Over apple	0.0100	
			Ripe pear	0.0000	
			Overripe pear	0.0061	
		20	Ripe apple	0.4142	0.1300
			Over apple	0.1392	
			Ripe pear	0.1833	
			Overripe pear	0.6463	
		30	Ripe apple	0.8702	0.1320
			Over apple	0.8270	
			Ripe pear	0.8322	
			Overripe pear	0.8426	
		50	Ripe apple	0.8791	0.1334
			Over apple	0.8292	
			Ripe pear	0.8909	
			Overripe pear	0.8981	

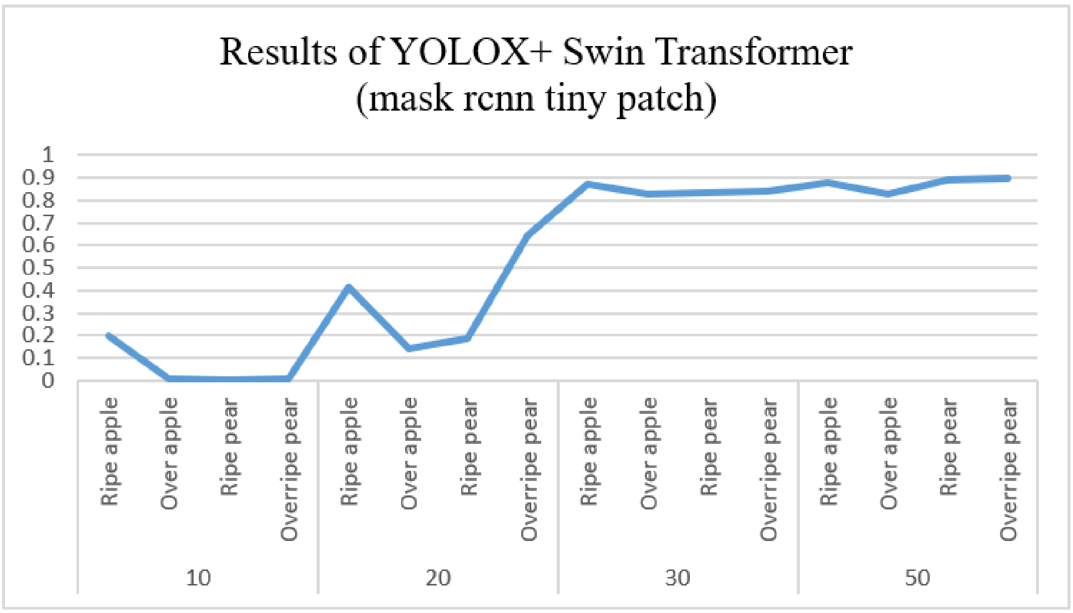


Figure 5.4: The results of YOLOX + Swin Transformer using Mask R-CNN tiny patch

Tables 5.5 and Table 5.6 show that the enhanced YOLOX combined with the Swin Transformer model utilizes the Swin Transformer as a replacement for the backbone (CSPDarknet) in YOLOX. This substitution reduces the activation function and normalization layer of the Neck and Head components in YOLOX, thereby enhancing feature extraction capabilities and optimizing the network structure. Experimental results substantiate that the improved YOLOX+Swin Transformer model simultaneously attains the speed advantage of the YOLO model and the accuracy of the Swin Transformer model.

The shifted window segmentation in the Swin Transformer results in an increased number of windows, thus increasing the computational demands associated with fitting smaller windows into larger ones. Conversely, the shifted window approach achieves identical calculation results while preserving the same number of windows as window attention through the utilization of a well-designed mask. Figure 5.4 demonstrates that the Swin Transformer achieves superior training results by employing this strategy. Notably, higher weight values and an increased number of training iterations contribute to enhanced model performance.

MLP places great emphasis on the concept of feature transfer. In order to establish local dependencies, the MLP model places the spatial positions in the same location using axial displacement. The performance of the model can be considered comparable to that of the Transformer model.

The computational requirements for MSA and W-MSA can be understood through eq. (3.33) and (3.34) respectively. The complexities of W-MSA and MSA are linked to $(h{\times}w)^2$ and $(h{\times}w)^2$, respectively. Consequently, the computational demands for W-MSA remain relatively modest.

The advantage of W-MSA becomes increasingly apparent when confronted with large original images. Resultantly, it is observed in Table 5.2, Table 5.5, and Table 5.6 that experiments utilizing the Swin Transformer module can yield expedited processing speeds.

Various frameworks were employed to implement the Swin Transformer, as elaborated in Table 5.5 and Table 5.6. The training results for classification exhibit improvement and enhanced stability as the number of iterations increases. In traditional Transformers, pre-normalization is employed at the inception of each residual branch. This technique normalizes the input size without imposing constraints on the output. Pre-normalization involves the accumulation of output activation values from each residual branch into the main branch, resulting in an amplification of the magnitude of the main branch as the depth increases.

On the contrary, the Swin Transformer employs a residual-post-normalization method. This method involves relocating the normalization layer from the beginning to the end of each residual branch. This adjustment guarantees that the output of each residual branch undergoes normalization prior to its reintegration into the primary branch. Consequently, as the network's depth increases, the accumulation of magnitude in the primary branch is alleviated.

Ranging from tiny to large, four distinct YOLOv5 weight configurations were utilized. The evaluation of this model at each epoch during the training process was conducted using the validation dataset, including the computation of both loss and precision. Therefore, the progress could be monitored throughout its training. Subsequently, the test dataset was employed to assess the model's overall precision after its development and training.

Upon testing the YOLOv5 model, it was observed that larger weight files resulted in lengthier training periods, despite the identical number of epochs. YOLOv5 customizes the depths and widths of specific sub-modules in accordance with the $depth\_multiple$ and $width\_multiple$ parameters provided from the YAML file. The YOLOv5 weight files maintain a consistent overall architecture across different sizes, namely $s, m, l, and\ x$. These variables ensure that the YOLOv5 variations are downsampled by a factor of 32 and contain three prediction feature layers.

The issue of data redundancy was addressed in the conducted experiments. The choice was made to train the model with the minimum number of epochs, as presented in Tables 5.7 to 5.10. Insufficient training using fruit image features was observed when the number of model iterations was too low, impeding the successful propagation of feature maps through the deep neural network. Conversely, it was discovered that balancing the number of iterations resulted in an enhanced model performance.

Table 5.7: YOLOv5x results

Model	Epoch	Classes	AP50	AP@0.5:0.95
YOLOv5x	30	Ripe apple	0.9995	0.9640
		Overripe apple	0.9995	0.9230
		Ripe pear	0.9995	0.9390
		Overripe pear	0.9995	0.9320
	50	Ripe apple	0.9995	0.9760
		Overripe apple	0.9996	0.9180
		Ripe pear	0.9995	0.9750
		Overripe pear	0.9995	0.9770

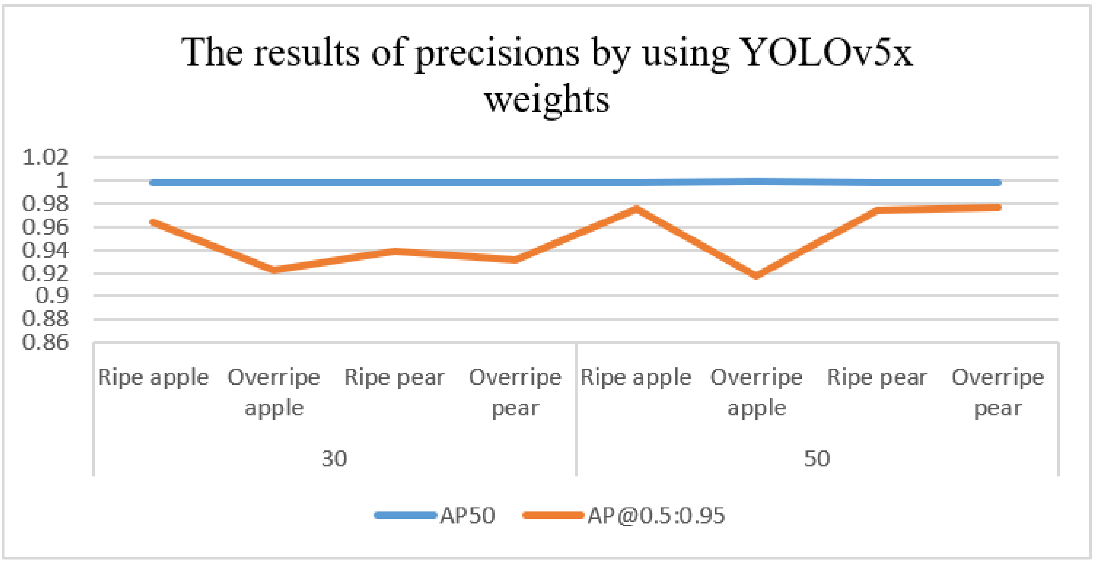


Figure 5.5: The results of YOLOv5x model

Table 5.8:YOLOv5l results

Model	Epoch	Classes	AP50	AP@0.5:0.95
YOLOv5l	30	Ripe apple	0.9995	0.9550
		Overripe apple	0.9995	0.9580
		Ripe pear	0.9995	0.9370
		Overripe pear	0.9995	0.9320
	50	Ripe apple	0.9994	0.9850
		Overripe apple	0.9996	0.9340
		Ripe pear	0.9994	0.9820
		Overripe pear	0.9994	0.9790

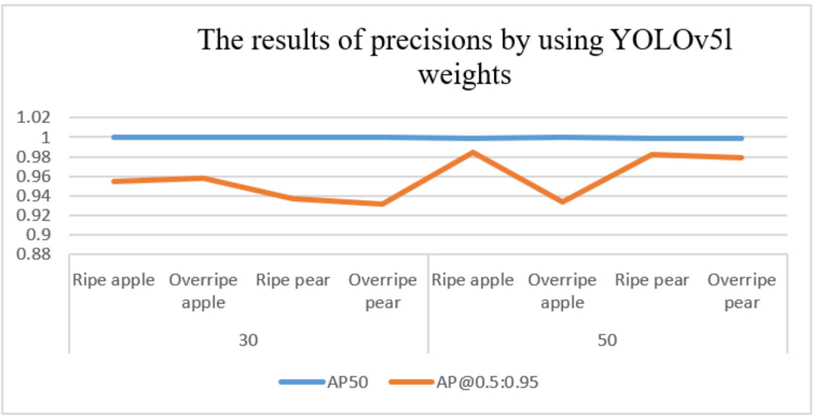


Figure 5.6: The results of YOLOv5l model

Table 5.9: YOLOv5m results

Model	Epoch	Classes	AP50	AP@0.5:0.95
YOLOv5m	30	Ripe apple	0.9991	0.9540
		Overripe apple	0.9995	0.9260
		Ripe pear	0.9995	0.9020
		Overripe pear	0.9995	0.8980
	50	Ripe apple	0.9996	0.9370
		Overripe apple	0.9995	0.8960
		Ripe pear	0.9995	0.9440
		Overripe pear	0.9996	0.8620

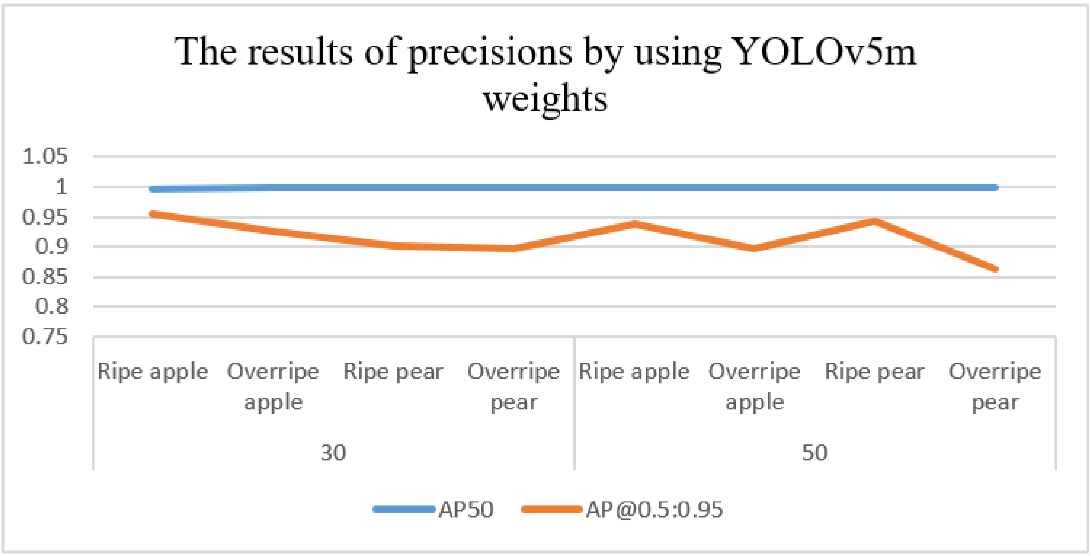


Figure 5.7: The results of YOLOv5m model

Table 5.10: YOLOv5s results

Model	Epoch	Classes	AP50	AP@0.5:0.95
YOLOv5s	30	Ripe apple	0.9994	0.8390
		Overripe apple	0.9995	0.9390
		Ripe pear	0.9995	0.8970
		Overripe pear	0.9995	0.8970
	50	Ripe apple	0.9994	0.9060
		Overripe apple	0.9995	0.8800
		Ripe pear	0.9995	0.9230
		Overripe pear	0.9995	0.9220

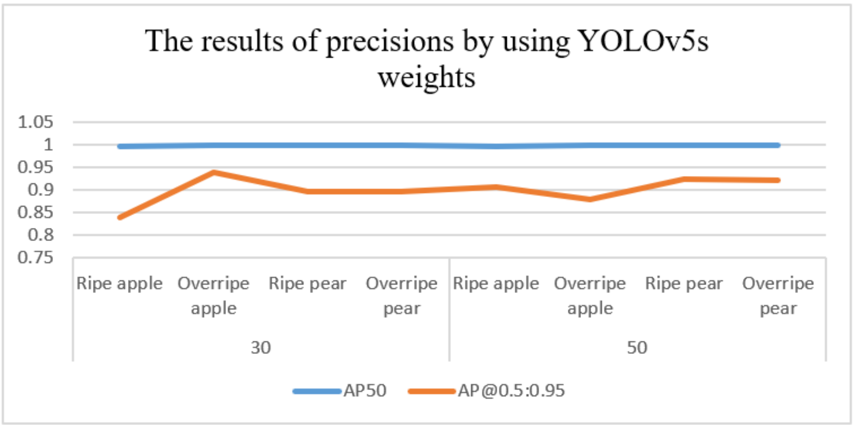


Figure 5.8: The results of YOLOv5s model

From Fig. 5.5 to Fig. 5.8, the tables exhibit the commendable performance of the YOLOv5 model. Notably, the performance of the YOLOv5x model was significantly improved owing to substantial weight and parameter adjustments. In the three-class classification task, Faster R-CNN was tested, revealing improved results for one class while relatively poorer outcomes for the other two classes. The discrepancy in categorization results can be attributed to the distribution of data within the training set.

The complexity of the deep neural network escalates with an increase in the number of training epochs. The diversity of the image dataset may influence the number of epochs required for training. Therefore, it is necessary to continually modify the number of epochs based on the specific characteristics of the fruit that the model aims to identify.

Table 5.11: YOLO using Swin Transformer results

Model	Epoch	AP50	AP@0.5:0.95
Swin + YOLO	10	0.1980	0.6360
	20	0.2050	0.5400
	30	0.5150	0.9950
	50	0.1850	0.5200

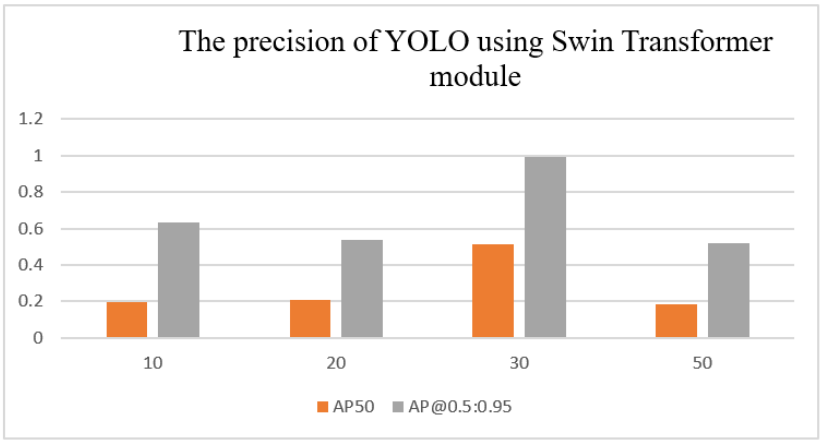


Figure 5.9: The results of YOLO + Swin Transformer model

The mean average precision rates exhibit a gradual decline as the weights are systematically reduced, with one class displaying a slightly inferior rate compared to the others. We have extended the sample size of the dataset and refined the weights during the training process to minimize the cost function and address the challenges encountered in their studies. In addition, they have explored the adaptation of YOLOv5 parameters through the utilization of weight decay techniques.

Figure 5.9 shows the utilization of a more potent deep neural network for further exploration, taking into account both the dataset at hand and the results of the conducted tests. To achieve a more streamlined model while maintaining the same level of accuracy in future efforts, we intend to enhance the architecture of the YOLO models and eliminate redundant parameters.

However, in Table 5.11, the pairing of YOLO model with Swin Transformer model failed to yield satisfactory results. This hybrid model retains the advantageous characteristic of faster training exhibited by YOLO models. Nevertheless, the introduction of the Transformer decoder for direct prediction of target object bounding boxes has led to prolonged convergence periods and subpar tracking performance.

Table 5.12: DETR with ResNet-50 results

Model	Epoch	Classes	AP@0.5:0.95	Average inference time (seconds)
DETR + ResNet-50	10	Ripe apple	0.7268	0.002
		Overripe apple	0.7374	
		Ripe pear	0.7958	
		Overripe pear	0.7714	
	20	Ripe apple	0.6819	0.004
		Overripe apple	0.7327	
		Ripe pear	0.9663	
		Overripe pear	0.7046	
	30	Ripe apple	0.7953	0.004
		Overripe apple	0.7959	
		Ripe pear	0.6832	
		Overripe pear	0.8560	
	50	Ripe apple	0.7296	0.002
		Overripe apple	0.7764	
		Ripe pear	0.7427	
		Overripe pear	0.7450	

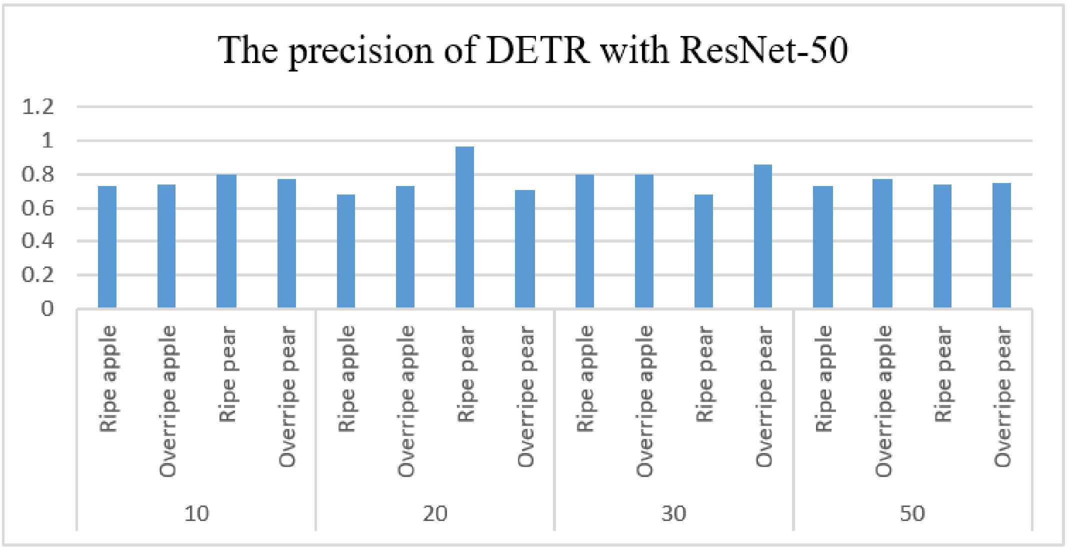


Figure 5.10: The results of Detection transformer using ResNet50

The results of the detection transformer studies are presented in Table 5.12. However, the DETR model did perform deficiently in the field of fruit ripeness detection tasks. We infer that this deficiency can affect the diminutive size of the fruit items. Therefore, complex network optimizations such as the incorporation of FPN or BiFPN components are deemed unnecessary.

From Figure 5.10, it is evident that the information pertaining to each fruit category is captured in a balanced manner by the detection transformer model.

The limitations of the detection transformer become apparent during the testing phase, where it encounters difficulties in accurately positioning objects. Nevertheless, DETR compensates for the shortcomings commonly observed in anchor-free algorithms. DETR proves to be particularly advantageous in scenarios involving overlapping objects. The detection transformer model excels in the domain of large object detection.

Table 5.13: Swin Transformer + Mask R-CNN results

Model	Epoch	AP50	AP@0.5:0.95	Average inference time (seconds)
Swin + Mask R-CNN	10	0.9360	0.8220	0.042
	20	0.9360	0.8360	0.048
	30	0.9580	0.9580	0.042
	50	0.9670	0.9670	0.044

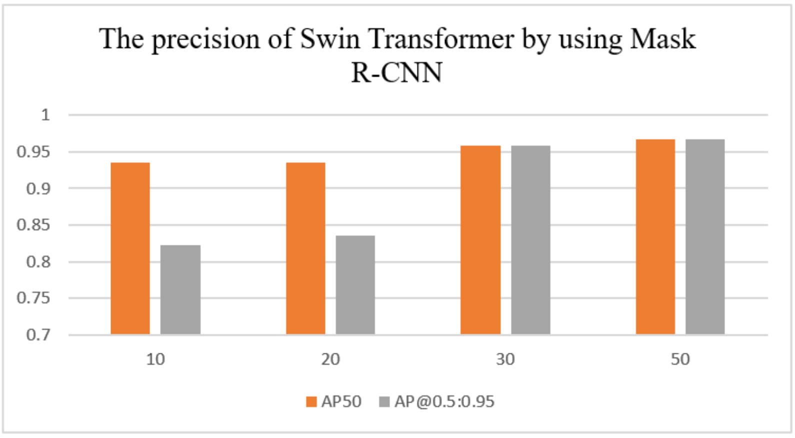


Figure 5.11: The results of Swin Transformer model

In contrast to Table 5.11, Table 5.13 combines the Swin Transformer and Mask R-CNN to yield a sequence-to-sequence model feature. This integration enhances the generalization ability of this model to seamlessly incorporate multimodal inputs, thereby providing greater flexibility in constructing network structures. Notably, as depicted in Figure 5.11, the model exhibits improved performance as the number of iterations increases.

Table 5.14: CornerNet results

Model	Epoch	AP@0.5:0.95	Average inference time (seconds)
CornerNet+ Hourglass	10	0.718	0.60
	20	0.245	0.58
	50	0.789	0.56

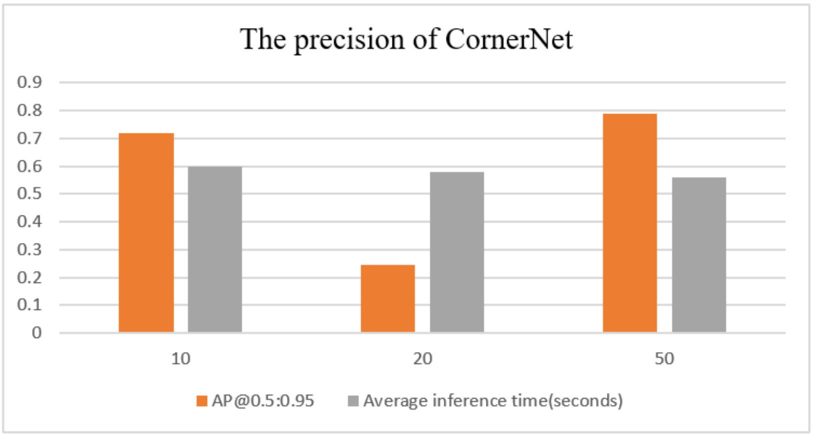
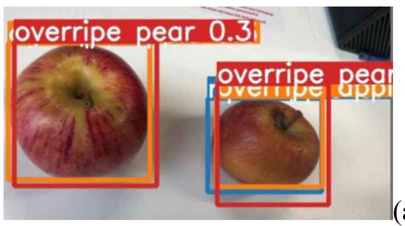


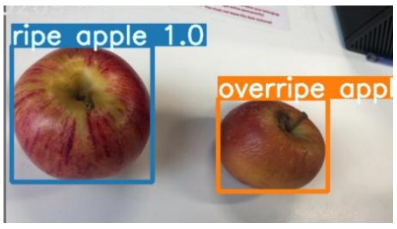
Figure 5.12: The of results of CornerNet model

If the accuracy rate is similar, the CornerNet, with its identical anchor-free approach and backbone, necessitates a longer average inference time, as illustrated in Table 5.12 and Table 5.14. This indicates that CornerNet falls short in achieving rapid detection speeds in practical applications. In addition, the detection results of CornerNet are inferior to those of other models, as evident from the comparison between Figure 5.11 and Figure 5.12. In contrast, both the Swin Transformer and DETR demonstrate faster detection speeds, as shown in Tables 5.12 and 5.13. Additionally, while maintaining a similar level of computational speed, the Swin Transformer outperforms DETR in terms of performance.

The accuracy of fruit detection was significantly enhanced by the incorporation of a bag-of-freebies in the YOLOv7 model. The utilization of the scaling method in the YOLOv7 model effectively mitigated the loss of visual information. By employing adaptive image scaling, the model was able to deepen its understanding of visual features while ensuring consistency in overall image transformation, thereby effectively leveraging the information from the receptive field. The substitution of the reparametrized module and the allocation of dynamic label assignments facilitated the computation of prediction results and ground truth values, thereby endowing the dominant leader with robust learning capabilities throughout the optimization process.



(a)



(b)

Figure 5.13: (a) and (b) are the images including fruits and the predicted boxes

Table 5.15: YOLOv7 results

Model	Weights	Epoch	Class Synchronous	AP@.5	AP@.5:.95
YOLOv7	yolov7	10	Ripe apple	0.468	0.427
			Overripe apple	0.885	0.764
			Ripe pear	0.169	0.115
			Overripe pear	0.209	0.151
		20	Ripe apple	0.427	0.419
			Overripe apple	0.995	0.932
			Ripe pear	0.673	0.622
			Overripe pear	0.967	0.944
		30	Ripe apple	0.993	0.974
			Overripe apple	0.996	0.948
			Ripe pear	0.542	0.534
			Overripe pear	0.995	0.932
		50	Ripe apple	0.996	0.992
			Over apple	0.996	0.979
			Ripe pear	0.996	0.981
			Overripe pear	0.996	0.991
		100	Ripe apple	0.996	0.992
			Overripe apple	0.996	0.988
			Ripe pear	0.996	0.995
			Overripe pear	0.996	0.990

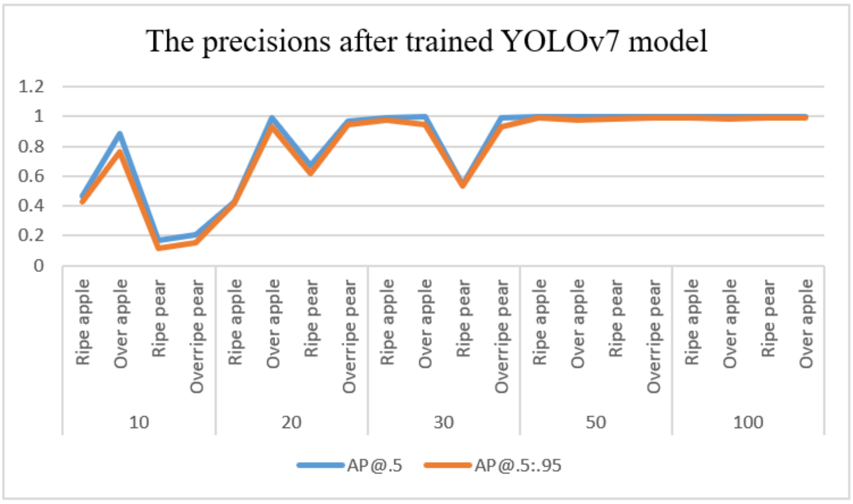


Figure 5.14: The results of YOLOv7 model

Table 5.16: YOLOv7-tiny results

Model	Weights	Epoch	Class	AP@.5	AP@.5:.95
YOLOv7	yolov7-tiny	10	Ripe apple	0.660	0.259
			Overripe apple	0.144	0.041
			Ripe pear	0.075	0.031
			Overripe pear	0.056	0.018
		20	Ripe apple	0.470	0.336
			Overripe apple	0.730	0.526
			Ripe pear	0.859	0.697
			Overripe pear	0.165	0.113
		30	Ripe apple	0.665	0.608
			Overripe apple	0.651	0.593
			Ripe pear	0.778	0.691
			Overripe pear	0.492	0.370
		50	Ripe apple	0.995	0.935
			Overripe apple	0.995	0.905
			Ripe pear	0.995	0.898
			Overripe pear	0.880	0.757
		100	Ripe apple	0.995	0.958
			Overripe apple	0.995	0.963
			Ripe pear	0.995	0.930
			Overripe pear	0.995	0.953

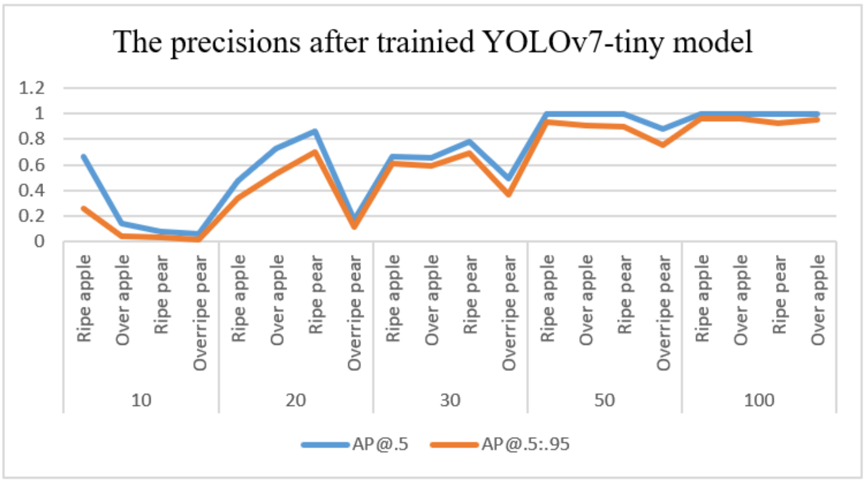


Figure 5.15: The results of YOLOv7-tiny model

Table 5.17: YOLOv7-X results

Model	Weights	Epoch	Class	AP@.5	AP@.5:.95
YOLOv7	yolov7-X	10	Ripe apple	0.439	0.393
			Overripe apple	0.843	0.718
			Ripe pear	0.344	0.224
			Overripe pear	0.436	0.396
		20	Ripe apple	0.918	0.906
			Overripe apple	0.996	0.949
			Ripe pear	0.390	0.352
			Overripe pear	0.517	0.470
		30	Ripe apple	0.996	0.978
			Overripe apple	0.996	0.960
			Ripe pear	0.996	0.919
			Overripe pear	0.996	0.959
		50	Ripe apple	0.996	0.991
			Overripe apple	0.997	0.983
			Ripe pear	0.996	0.994
			Overripe pear	0.996	0.989
		100	Ripe apple	0.996	0.992
			Overripe apple	0.996	0.989
			Ripe pear	0.996	0.996
			Overripe pear	0.996	0.993

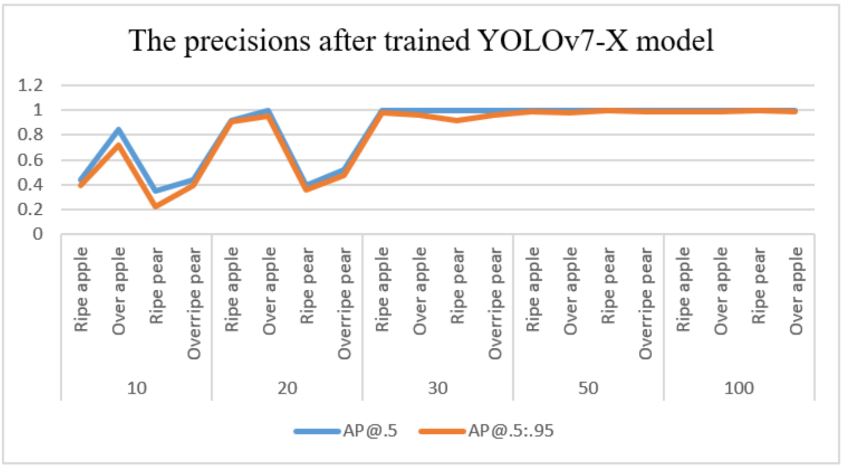


Figure 5.16: The results of YOLOv7-X model

For our experimental analysis, we selected four weight variants of YOLOv7 model, namely, YOLOv7, YOLOv7-X, YOLOv7-E6, and YOLOv7-tiny. We loaded the pretrained weights and compared the backbone network, including the pretrained weights, to determine the extent of layer similarity.

Table 5.15, Table 5.16, and Table 5.17 demonstrate that YOLOv7 outperforms other variants in terms of fruit positioning. In Figure 5.13, panel (a) illustrates that although the model may not accurately determine the fruit category, it excels in precisely localizing the fruit. Once the model has acquired sufficient knowledge of features, Figure 5.13 (b) showcases the detection results achieved by the model.

In Table 5.18, it was observed that the E-ELAN module, when equipped with YOLO-E6 weights, facilitates the efficient convergence of the deep network by regulating the shortest and longest gradient paths in the same number of iterations. The YOLO-tiny weight employs the utilization of the LeakyReLU function to address the issue of parameter stagnation following the acceptance of outlier range inputs by the neural network. During the backpropagation process, a substantial gradient is generated due to the continuous multiplication of derivatives, thereby impeding parameter updates. This phenomenon gives rise to the vanishing gradient problem. By assigning negative values to inputs below 0 in the LeakyReLU function, a small gradient is introduced, effectively circumventing the issue of aliasing in the gradient direction. As depicted in Table 5.19, with an increasing number of epochs, the number of iterations for weight updating rises, resulting in the transition of the curve from an initial state of poor fitting to an optimal state of fitting.

Table 5.18: YOLOv7-E6 results

Model	Weights	Epoch	Class	AP@.5	AP@.5:.95
YOLOv7	yolov7-E6	10	Ripe apple	0.334	0.277
			Over apple	0.437	0.358
			Ripe pear	0.156	0.088
			Overripe pear	0.714	0.538
		20	Ripe apple	0.377	0.342
			Over apple	0.365	0.319
			Ripe pear	0.234	0.204
			Overripe pear	0.589	0.511
		30	Ripe apple	0.993	0.897
			Over apple	0.995	0.921
			Ripe pear	0.227	0.209
			Overripe pear	0.995	0.971
		50	Ripe apple	0.994	0.988
			Over apple	0.996	0.969
			Ripe pear	0.995	0.930
			Overripe pear	0.995	0.923
		100	Ripe apple	0.995	0.991
			Over apple	0.996	0.986
			Ripe pear	0.995	0.995
			Over pear	0.995	0.992

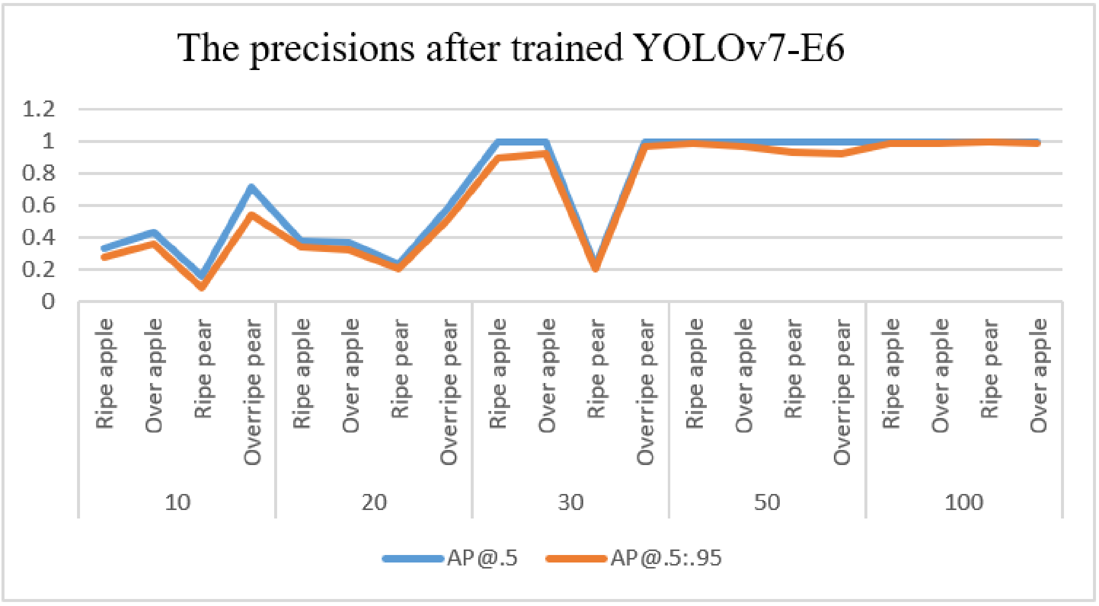


Figure 5.17: The results of YOLOv7-E6 model

Table 5.19: Mean average precisions of YOLOv7e (mAP)

Model	Weights	Epoch	AP@.5	AP@.5:.95	Average inference time (millisecond)
YOLOv7	YOLOv7	10	0.433	0.364	6
		20	0.766	0.730	6
		30	0.882	0.847	6
		50	0.996	0.986	6
		100	0.996	0.991	6
	YOLOv7-tiny	10	0.234	0.087	8
		20	0.556	0.418	8
		30	0.644	0.565	7
		50	0.966	0.874	7
		100	0.995	0.951	6
	YOLOv7-X	10	0.515	0.433	9
		20	0.705	0.669	9
		30	0.996	0.954	9
		50	0.996	0.989	9
		100	0.996	0.993	10
	YOLOv7-E6	10	0.410	0.315	13
		20	0.391	0.344	13
		30	0.803	0.749	13
		50	0.995	0.952	13
		100	0.995	0.991	12

Table 5.20: ConvNeXt results

Model	Weights	Epoch	AP@.5	AP@.5:.95	Average inference time (millisecond)
ConvNeXt	ConvNext + Mask R-CNN	10	0.848	0.719	5
		20	0.948	0.678	13
		30	0.926	0.669	37
		50	0.844	0.617	58
	ConvNext + Mask R-CNN Transfer Learning	10	0.500	0.701	14
		20	0.487	0.695	4
		30	0.483	0.694	5
		50	0.483	0.695	5

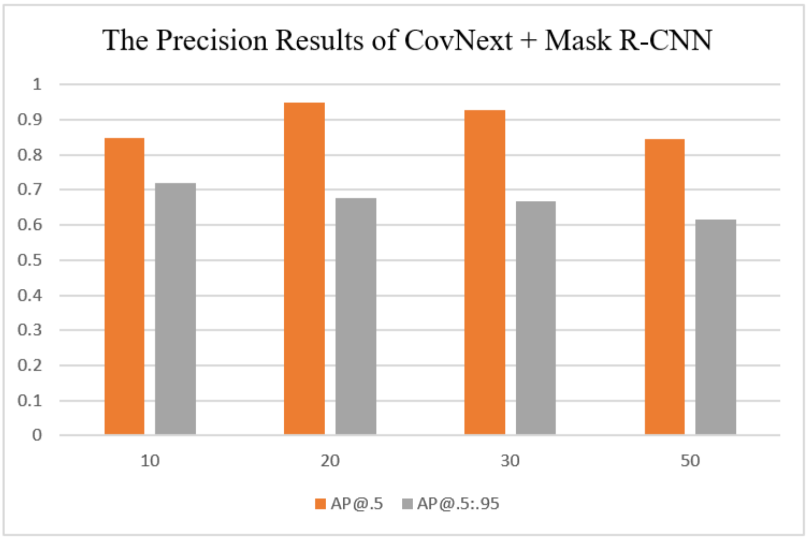


Figure 5.18: The results of ConvNeXt + Mask R-CNN model

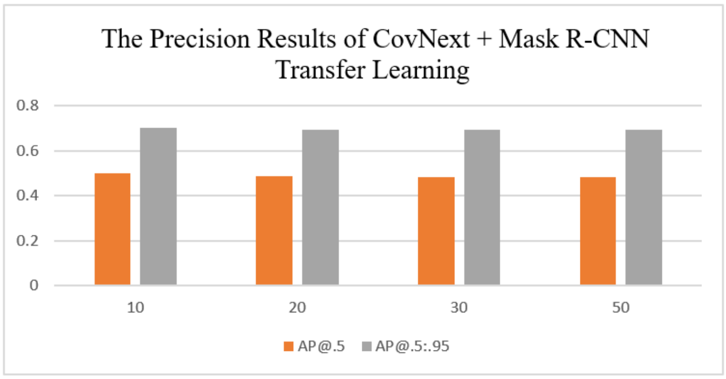


Figure 5.19: The results of CovNeXt + Mask R-CNN transfer learning

As a traditional CNN model, ConvNeXt exhibits superior training results. Under identical training parameters, the transfer learning model does not confer significant advantages. However, as illustrated in Figure 5.18 and Figure 5.19, transfer learning offers time-saving benefits when certain pre-training parameters are frozen. The ConvNeXt model fails to fully capture fruit features. In Table 5.19 and Table 5.20, the ConvNeXt transfer learning model demonstrates a marginal advantage in terms of detection speed. Nevertheless, in regard to accuracy, the YOLO model remains superior.

The CenterNet model and the YOLOv8 model are both fundamentally anchor-free in nature. The CenterNet model employs ResNet-50 as its backbone, while the YOLOv8 model utilizes three distinct weight configurations, namely YOLO8n, YOLOv8m, and YOLOv8x, which vary in size from small to large.

Table 5.21: CenterNet results

Model	Weights	Freeze Epoch	Epoch	AP0.5	AP@0.5:0.95	Average inference time (millisecond)
CenterNet	CenterNet_ResNet-50	20	30	0.138	0.122	10
		30	50	0.138	0.125	10
		50	100	0.135	0.126	10
		100	200	0.913	0.854	10
		200	300	0.933	0.889	10
		300	400	0.960	0.909	10
		400	500	0.928	0.881	10

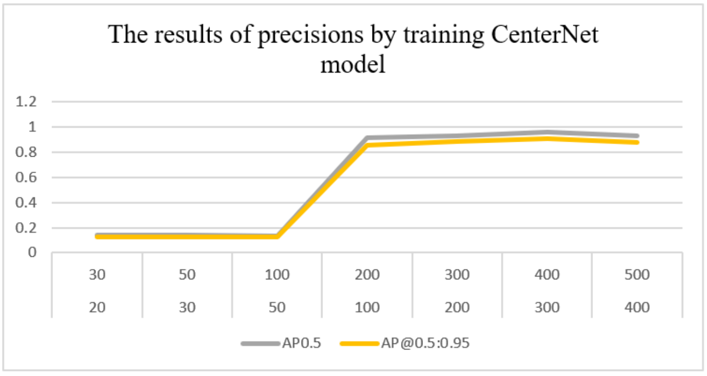
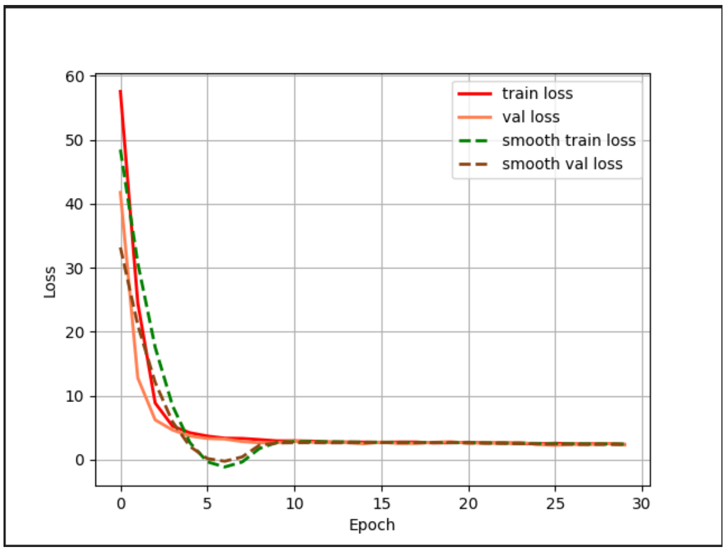
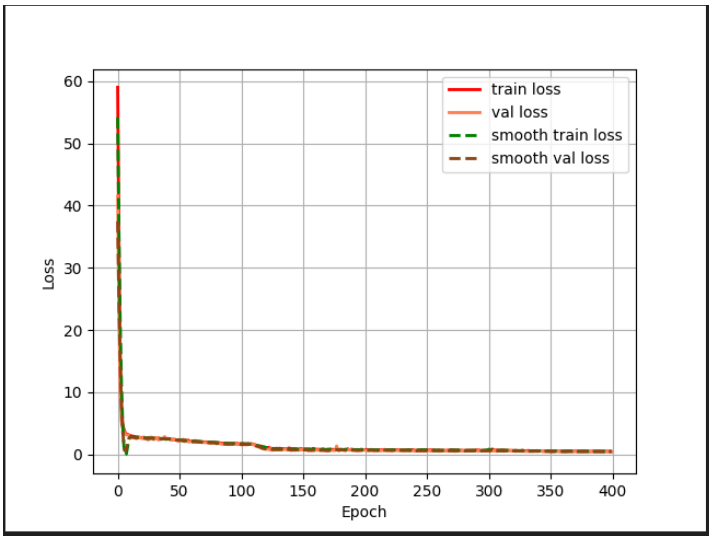


Figure 5.20: The results of CenterNet model

During the inference process, CenterNet suppresses the eight positions surrounding the key point of the local maximum value and selects the top k bounding boxes as the output. Simultaneously, it incorporates an output threshold for control. The prediction mechanism of CenterNet, which is centered around the key point, distinguishes it from anchor-based models. In contrast, anchor-based detection models exhibit a dense distribution of center points. Therefore, as depicted in Figure 5.20, the CenterNet model necessitates a certain number of iterations to fully grasp the underlying features.



(a) 30 epochs Loss map.



(b) 400 epochs Loss map.

Figure 5.21: Loss maps

Throughout the model training phase, our samples were categorized into two groups: Training and validation, maintaining a ratio 9:1. The loss value of the training model is subsequently divided into the overall loss for the training set and the validation loss for the test set. A decline in both the training and validation losses, as illustrated in Figure 5.21 (b), signifies successful training and the proximity of this model to its optimal state. Conversely, if both the training and validation losses remain stagnant, as depicted in Figure 5.21 (a), it indicates potential bottlenecks in the learning process, erroneous training parameter selections, and subpar model performance. The feature extraction network remains unaltered, and the backbone remains frozen during the CenterNet training procedure. Consequently, extending the training period can help circumvent the identification of local optimal solutions.

The anchor-free structure (AFS) of the YOLOv8 model integrates dynamic task alignment learning and a combination of distribution focal loss (DFL) and CIoU loss for the regression branch, thereby fostering a high consistency between the classification and regression tasks. By concerning data enhancement during the training phase, the deactivation of mosaic enhancement in the final 10 epochs enhances stability in model convergence. As presented in Table 5.22, an enhancement in the number of training epochs from 50 to 100 results in a more comprehensive model training. Finally, YOLOv8 attains a state of lightweight and efficient detection capabilities.

Table 5.22: YOLOv8 results

Model	Weights	Epoch	mAP0.5	mAP@0.5:0.95	Average inference time (millisecond)
YOLOv8	YOLOv8n	10	0.995	0.793	2.8
		20	0.994	0.958	3.4
		30	0.995	0.982	3.1
		50	0.995	0.990	3.6
		100	0.995	0.993	2.9
	YOLOv8m	10	0.982	0.831	6.6
		20	0.995	0.956	6.6
		30	0.995	0.985	6.7
		50	0.995	0.991	6.6
		100	0.995	0.993	6.6
		200	0.995	0.994	6.5
	YOLOv8x	10	0.947	0.815	16.7
		20	0.995	0.962	17.3
		30	0.995	0.986	16.9
		50	0.995	0.992	17.2
		100	0.995	0.993	17.2
		200	0.995	0.993	17.3

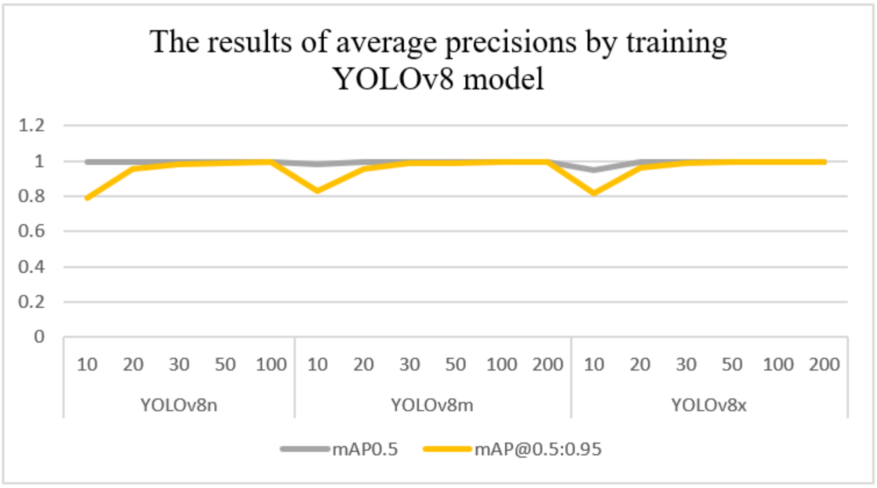
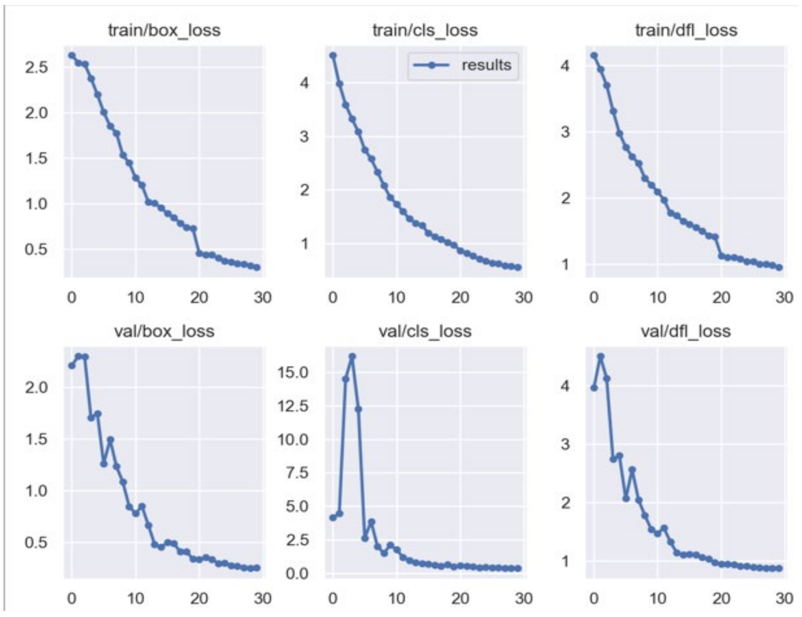
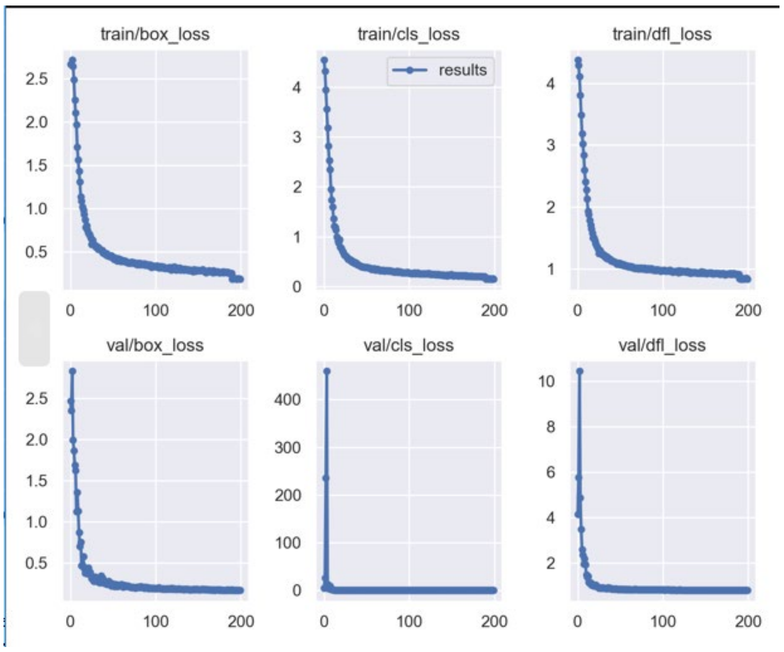


Figure 5.22: The results of YOLOv8 model



(a) YOLOv8n 30 epochs Loss map.



(b) YOLOv8n 200 epochs Loss map.

Figure 5.23: YOLOv8n loss map

Table 5.23: YOLOv8n results

Model	Weights	Epoch	Class	AP0.5	AP@0.5:0.95
YOLOv8	YOLOv8n	10	Ripe apple	0.994	0.800
			Overripe apple	0.995	0.864
			Ripe pear	0.985	0.808
			Overripe pear	0.848	0.679
		20	Ripe apple	0.994	0.973
			Overripe apple	0.995	0.942
			Ripe pear	0.995	0.972
			Overripe pear	0.992	0.944
		30	Ripe apple	0.995	0.981
			Overripe apple	0.995	0.973
			Ripe pear	0995	0.993
			Overripe pear	0.995	0.981
		50	Ripe apple	0.994	0.990
			Overripe apple	0.995	0.984
			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.991
		100	Ripe apple	0.995	0.991
			Overripe apple	0.995	0.990

			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.994

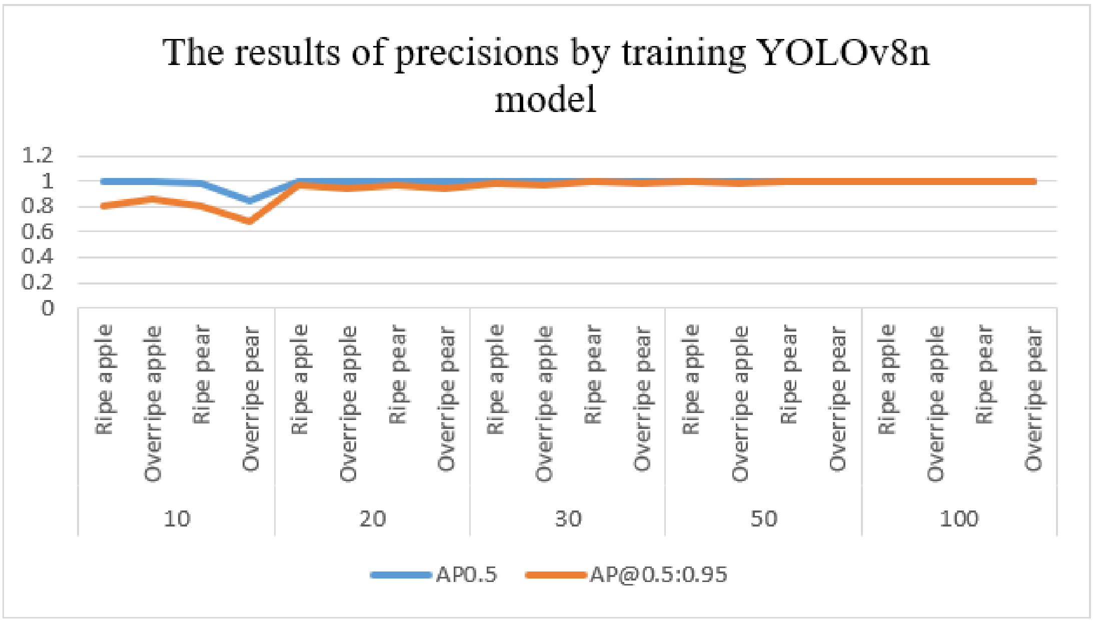


Figure 5.24: The results of YOLOv8n model

The utilization of larger pre-training weights leads to lengthier average inference times, as evidenced by the data presented in Table 5.22. The employment of weight magnification leads to an increase in the average processing times for each image, albeit without any improvement in average precision, as indicated in Table 5.23, Table 5.24, and Table 5.25. In fact, the adoption of weight amplification can potentially result in overfitting during the training process, thereby squandering valuable resources. Throughout the training procedure, a reduced learning rate was employed to facilitate the model's ability to effectively converge upon the optimal point. However, it is necessary to note that an excessive number of iterations can still pose challenges in terms of training efficacy for the model.

Table 5.24: YOLOv8m results

Model	Weights	Epoch	Class	AP0.5	AP@0.5:0.95
YOLOv8	YOLOv8m	10	Ripe apple	0.990	0.841
			Overripe apple	0.995	0.880
			Ripe pear	0.995	0.852
			Overripe pear	0.949	0.753
		20	Ripe apple	0.995	0.961
			Overripe apple	0.995	0.937
			Ripe pear	0.995	0.982
			Overripe pear	0.995	0.945
		30	Ripe apple	0.995	0.948
			Overripe apple	0.995	0.975
			Ripe pear	0.995	0.992
			Overripe pear	0.995	0.988
		50	Ripe apple	0.995	0.992
			Overripe apple	0.995	0.984
			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.992
		100	Ripe apple	0.994	0.992
			Overripe apple	0.995	0.992
			Ripe pear	0.995	0.994
			Overripe pear	0.995	0.994
		200	Ripe apple	0.995	0.993
			Overripe apple	0.995	0.992
			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.995

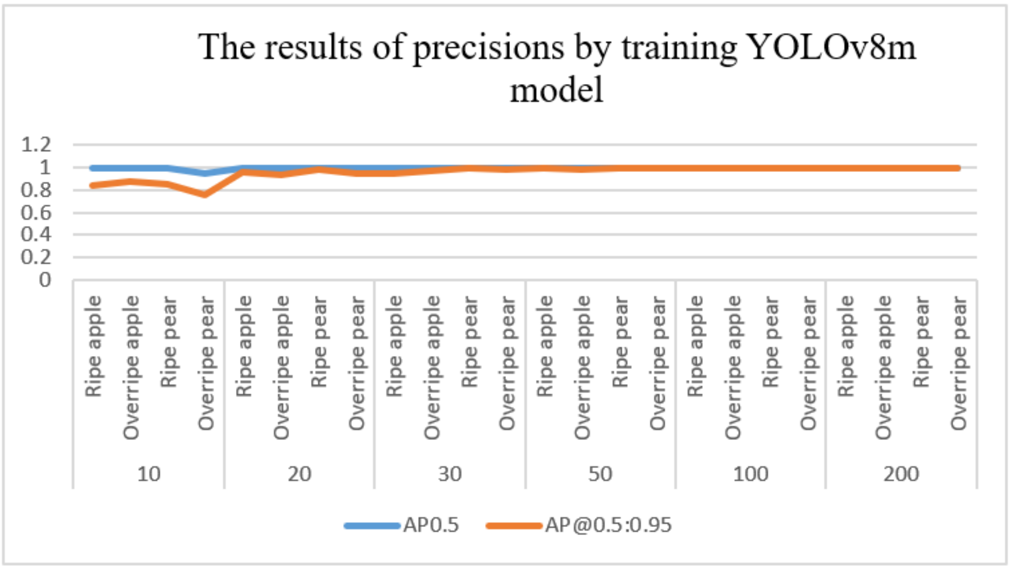


Figure 5.25: The results of YOLOv8m model

Table 5.25: YOLOv8x results

Model	Weights	Epoch	Class	AP0.5	AP@0.5:0.95
YOLOv8	YOLOv8x	10	Ripe apple	0.995	0.840
			Overripe apple	0.995	0.852
			Ripe pear	0.990	0.865
			Overripe pear	0.808	0.703
		20	Ripe apple	0.995	0.978
			Overripe apple	0.995	0.943
			Ripe pear	0.995	0.963
			Overripe pear	0.995	0.965
		30	Ripe apple	0.994	0.989
			Overripe apple	0.995	0.978
			Ripe pear	0.995	0.990
			Overripe pear	0.995	0.989
		50	Ripe apple	0.994	0.991
			Overripe apple	0.995	0.987
			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.994
		100	Ripe apple	0.995	0.993
			Overripe apple	0.995	0.991
			Ripe pear	0.995	0.994
			Overripe pear	0.995	0.993
		200	Ripe apple	0.995	0.993
			Overripe apple	0.995	0.990
			Ripe pear	0.995	0.995
			Overripe pear	0.995	0.994

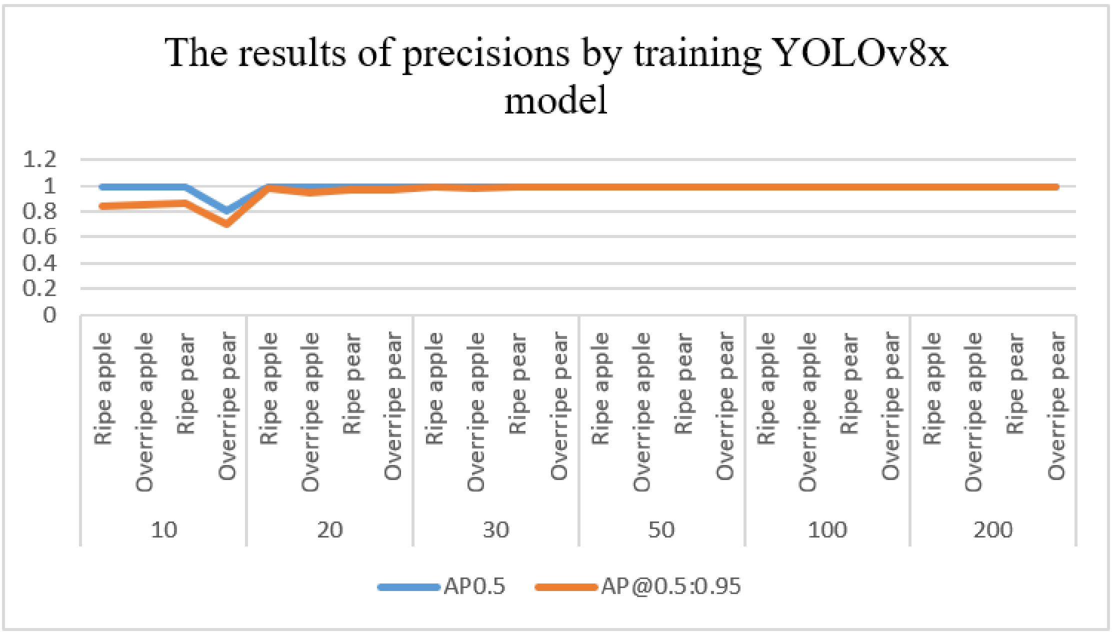


Figure 5.26: The results of YOLOv8x model

Table 5.26: Comparison results

Model	Weights	Epoch	AP0.5	AP@0.5:0.95	Average inference time (millisecond)
CenterNet	centernet _ResNet-50	100	0.135	0.136	10
		200	0.913	0.854	10
YOLOv8	YOLOv8n	100	0.995	0.993	2.9
	YOLOv8 m	200	0.995	0.994	6.5
	YOLOv8x	200	0.995	0.993	17.3

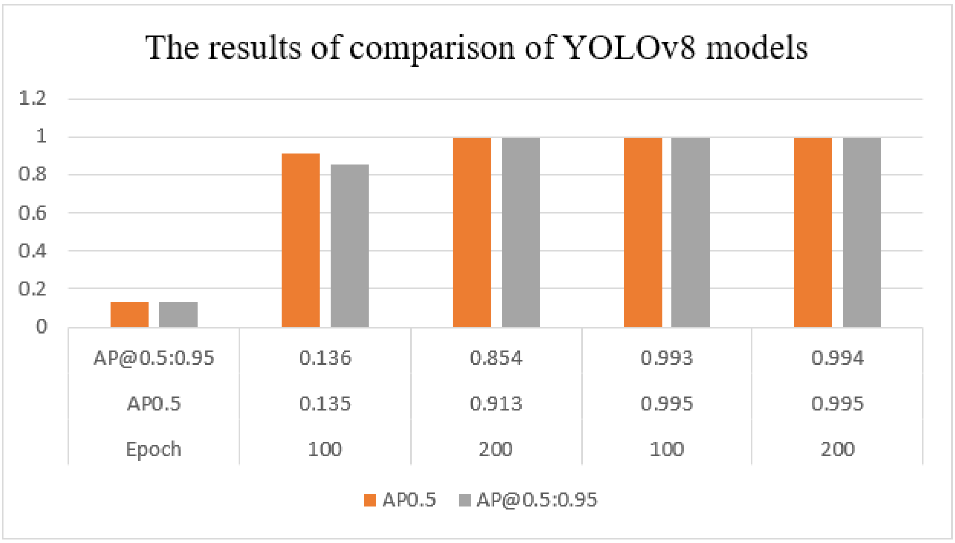


Figure 5.27: Comparison of YOLOv8 model

Ablation experiments were conducted in Table 5.26 to assess the training outcomes of both the YOLOv8 and the CenterNet model. The anchor-free models, YOLOv8 and CenterNet, exhibit commendable performance in fruit ripeness detection. While CenterNet effectively accomplishes transfer learning by freezing the backbone during training, it does not significantly surpass the YOLOv8 model in terms of training precision. Moreover, CenterNet necessitates more time for inference during a two hundred iteration training session compared to YOLOv8m. Despite the resource conservation during training offered by the CenterNet model and the reduction in overall weight facilitated by using C2f module, the lightweight YOLOv8n model can complete detection with swifter response times in just 100 iterations, assuming constant training parameters. It is critical to note that CenterNet encounters difficulties in accurately extracting fruit features beyond 100 iterations.

5.2 Discussions

The precision was achieved by using the green apple detection method (Sun et al. 2023) amounted to 34.2%. Employing the transformer model, Wang et al. (2023) successfully discriminated between various sizes and varieties of tomatoes, yielding an astonishing precision rate of 89.4%. Alzahrani and Alsaade (2023) utilized DenseNet-169 to attain an exceptional precision score of 99.88% in the identification of fruit lesions. In a separate investigation (Kim et al., 2023), the detection of small objects from UAV photos was explored, revealing that the YOLOv8 model exhibited a data processing rate of 45.7 Frames per Second (FPS) in the P2 layer, despite the presence of environmental noise.

Our fruit detection experiment adheres to the methodology employed in previous studies, employing a relatively minimal number of parameters. By maintaining a precision level of 99.3%, our YOLOv8 model is able to achieve a detection speed as low as 2.9 milliseconds, resulting in significant time savings in real-world applications. Our analysis demonstrates that the Swin Transformer model achieved an average precision of 87.43%. Within a mere 0.13 seconds, our algorithm swiftly and accurately identifies apples or pears in input photos, subsequently classifying them as either “Ripe” or

“Overripe”.

The inadequacy of the experiment in relation to previous studies necessitates a more thorough consideration of the influence of environmental noise on the dataset collection process, specifically pertaining to the diverse fruit environments. To accurately simulate the growth conditions experienced by the fruits during the harvesting process, it is necessary to consider modifying a wider range of pixel types in the dataset.

In our trials on fruit ripeness classification, the Swin Transformer model has exhibited its advantageous qualities and precision in handling minute targets and limited datasets. The practical implementation of fruit classification based on ripeness has been successfully executed, thereby enabling the utilization of this model in real-world scenarios such as automated agricultural harvesting and warehouse management.

Empirical evidence has revealed the inadequacy of the Vision Transformer in terms of CNN response. The CNN solely permits the calculation of correlations between adjacent pixels, rendering spatial information irrelevant due to the inherent limitations of sliding window convolution, which prevents the simultaneous estimation of non-local pixels. Conversely, the Swin Transformer facilitates the provision of hierarchical feature representation, thereby enabling successful feature extraction through the utilization of self-attention windows.

Our comparative analysis of the ConvNeXt and YOLOv7 architectures, which exemplify the CNN and YOLO frameworks respectively, has showcased remarkable performance in the domain of fruit detection. The ConvNeXt model builds upon the residual structure of ResNet, thereby significantly enhancing detection speed. By considering the primary objective of our research, which revolves around the realization of automated fruit harvesting, we conclude that the lightweight YOLOv7 model presents a more favorable equilibrium between detection accuracy and computational efficiency.

From a precision standpoint, it is observed that the YOLO series models exhibit superior performance. The YOLOv8 model demonstrates the ability to accomplish recognition tasks within a mere 2.9 seconds.

In the field of target detection, YOLOv5 and YOLOv7 represent two deep learning models. Their primary distinction lies in their network architecture and performance characteristics. YOLOv5, a lightweight target detection model, employs an FPN-based backbone network structure and an anchor-free detection methodology, thereby reducing both model computation and parameter count. Therefore, YOLOv5 has made significant strides in terms of speed and accuracy, thereby achieving enhanced performance. On the other hand, YOLOv7 represents a novel target detection model. In comparison to YOLOv5, it adopts a deeper network structure and introduces novel technical mechanisms, such as the Bottleneck Attention Module (BAM), which further enhances accuracy. Additionally, YOLOv7 incorporates supplementary structures, including the Spatial Pyramid Pooling (SPP) module and the Spatial Attention Module (SAM) module, thereby endowing the model with increased potency.

It is evident that the detection speed of YOLOv7 is marginally slower than that of YOLOv8, yet both models are capable of achieving real-time detection. However, in contrast to YOLOv5, YOLOv7 does not exhibit a significant improvement in detection accuracy.

While considering the YOLOv5, YOLOv7, and YOLOv8 models, it is apparent that the accuracy and recall do not increase with the deepening of the model. The presence of numerous redundant parameters in the model leads to a decline in various performance indicators. Therefore, for a small dataset, opting for a model with a reduced parameter count not only enhances accuracy and reduces training time but also confers a notable advantage in terms of prediction speed.

While comparing various models, it is evident that YOLOv8 exhibits a certain degree of improvement in terms of accuracy compared to YOLOv5 and YOLOv7, albeit the gap is not substantial.

The YOLOv8 model, renowned for its exceptional scalability, represents a state-of-the-art approach. This framework has been designed to accommodate all preceding iterations of YOLO, thereby facilitating transitions between different versions. In addition to its scalability, YOLOv8 includes a plethora of enhancements that enhance its adaptability for object identification and image segmentation applications. Notable advancements of the YOLOv8 model include a unique backbone network, an anchor-free network detecting head, and an upgraded loss function capability.

In terms of visual object detection speed, the Detection Transformer (DETR) model reigns supreme. However, it is important to note that the precision of the DETR model does not match that of the Swin Transformer. The Transformer architecture effectively transforms RNN into a superposition of multiple self-attention structures, thereby parallelizing the correlation between any element in the sequence and all other elements. This efficient extraction of contextual correlations is further complemented by introducing a multi-head attention mechanism, which enables feature extraction from multiple perspectives. Moreover, the utilization of position coding facilitates the depiction of sequence information before and after, effectively replacing the sequential calculation process of the RNN.

The advantage of the Transformer model over CNN lies in its ability to handle long-range dependencies. By employing the self-attention mechanism, the Transformer model establishes connections between the first and last words in a sentence, enabling the capture of global relationships. Therefore, the Vision Transformer can effectively utilize the attention mechanism to extract global features from images, surpassing the mere capture of dependencies between adjacent elements.

In contrast to CNN, which relies on the receptive field to calculate the correlation between each pixel of the feature map and all other pixels, DETR achieves this through the Transformer architecture. Notably, the Transformer model exhibits a broader receptive range than CNN, resulting in superior performance on the test set. In the Encoder stage of DETR, the characteristics of the input are encoded, while the Decoder stage converts 100 queries into 100 targets. Typically, 100 queries suffice, as few images contain more than 100 targets (unless when addressing highly complex tasks). In contrast, CNN-based methods require the prediction of tens of thousands of anchors, incurring substantial computational costs. The reduction in computational burden afforded by the Transformer model enables efficient object detection.

DETR's approach bears resemblance to Mask R-CNN, as it predicts the segmentation of the instance's corresponding bounding box based on the provided box prediction. To achieve this, DETR upsamples the attention map output by the mask head and incorporates it into specific branches of the backbone, effectively emulating the functionality of an FPN. Subsequently, a bitwise argmax operation is performed on the mask maps associated with all boxes to obtain the final segmentation map. Notably, the detection results obtained by Transformer-based models surpass those achieved by Mask RCNN in terms of accuracy and performance.

In the object recognition tasks of computer vision, an image typically undergoes a series of convolution operations, subsequently yielding a feature vector that encapsulates the comprehensive characteristics through the utilization of Global Average Pooling (GAP). This resultant feature vector is then fed into the classification layer for the purpose of classification. Conversely, in the field of NLP, tasks often employ a Class Token, referred to as CLS in the aforementioned context. A vector representing the class token, along with another vector representing the distillation token, are inputted into the Encoder of the Transformer to process the sequence of image patches.

During the training phase, a supervised loss is employed for the output of the class token, while a distillation loss is employed for the output of the distillation token. Finally, image recognition is performed by taking the mean value of the output from the class token and the output from the distillation token. The Transformer architecture necessitates a sequence (Sequence) input signal, whereas our image is a two-dimensional input signal. Therefore, we segment the image into blocks and subsequently apply the Flatten operation. However, this intuitive approach fails to perfectly model images due to the absence of internal information pertaining to each patch in ViT. Hence, ViT is unsuitable for detecting smaller objects.

The axial displacement strategy of MLP spatially shifts features in both horizontal and vertical directions, thereby arranging features from distinct spatial positions into the same position. Subsequently, the MLP is employed to combine these features. The axial displacement strategy of the MLP enables the model to acquire more localized dependencies, thereby facilitating rapid and accurate identification.

In terms of results and speed, the YOLOX decoupled head, which replaces the YOLO head, not only enhances the rate of convergence but also exhibits superior performance. By incorporating the Swin Transformer module, YOLOX+Swin Transformer outperforms alternative models in terms of both detection speed and accuracy.

5.3 Summary of This Chapter

Within the scope of this key chapter, a comprehensive analysis of all experimental findings is presented. To accomplish this objective, ablation experiments were conducted, enabling a meticulous dissection of the merits and demerits associated with various research methods.

Chapter 6 Conclusion and Future Work

In this chapter, an exhaustive synthesis of all research methodologies and experimental data is provided. Future research directions are visioned based on the empirical results. A clear and well-organized explanation of the numerous techniques and strategies employed in our research is proffered. This chapter comprises minute details pertaining to our data collection methodologies, experimental designs, and the analytical instruments employed to derive valuable insights.

6.1 Conclusion

Our findings substantiate that the combination of YOLOX with the Swin Transformer model yields superior detection results. YOLOX incorporates CSPNet, SiLU activation function, and PANet, which are based on YOLOv3 and YOLOv5. In comparison to the conventional YOLO model, YOLOX maintains a certain model size while achieving higher detection accuracy.

Compared to the earlier iterations of YOLO, the decoupling head in YOLOX exhibits notable differences. Specifically, a 1×1 convolution layer is employed in the decoupling head of YOLOX to modify the channel count prior to the integration of two parallel branches. Each branch consists of two convolution layers, with one branch dedicated to classification tasks and the other branch including regression tasks, which also incorporates a branch for determining the degree of overlap.

The architecture of the Focus network is also designed to optimize the extraction of YOLOX features. By assigning values to alternate pixels in an image, the Focus network generates four distinct feature layers. Subsequently, these four separate feature layers are stacked, thereby merging the width and height data into the channel data and multiplying the input channel by four.

Additionally, YOLOX Neck component employs a bottom-up feature pyramid to combine two parameters across different detection levels.

Functioning as a hierarchical network structure, the Swin Transformer effectively enhances computational speed through the utilization of shifted windows.

The layered architecture of the Swin Transformer affords modeling flexibility at various scales, thereby ensuring that the model can accomplish the classification task to its fullest potential.

The YOLOX+Swin Transformer model combines the characteristics of both models to enhance the accuracy of classification. Moreover, the network structure of these models guarantees the efficient and rapid transmission of characteristic details pertaining to fruits in the model.

The ConvNeXt network maintains the general framework and design principles of its predecessor, with the incorporation of advanced Transformer network concepts and specific modifications to the traditional ResNet-50 network. The objective of ConvNeXt is to combine the strengths of both networks and enhance the performance of CNNs by integrating recent Transformer network concepts and technology into the existing CNN network modules.

ConvNeXt employs a more radical approach by utilizing depth-wise convolution to generate a convolution block, as opposed to the bottleneck structure employed by ResNet. This results in a significant reduction in the network's parameter count, albeit at the cost of some accuracy. It is evident that ConvNeXt may not be the optimal choice for precision-oriented fruit detection tasks.

Both CornerNet and CenterNet establish the spatial relationship between key points and the detected targets. CornerNet establishes the positional relationship between the corner point and the target. On the other hand, CenterNet establishes the positional relationship between the center point and the target.

CornerNet adopts the hourglass network as the backbone feature extraction network. This hourglass network is typically employed in pose estimation tasks and comprises a combination of downsampling and upsampling operations in an hourglass-shaped architecture. However, when the CornerNet heatmap is downsampled to its original 1/n size and subsequently upsampled, it leads to a loss of accuracy. Therefore, the positional accuracy of small object frames is significantly compromised, resulting in unsatisfactory detection results for CornerNet.

CenterNet suppresses the eight positions surrounding the key point of the local maximum value during its processing, and selects the top k bounding boxes as the output. The prediction of CenterNet, which is based on the center point, results in less dense bounding boxes compared to anchor-based detection models. When fruits are being detected, it becomes susceptible to generating significant losses in center point offset.

The MLP model, which operates in both horizontal and vertical directions, can arrange features at various spatial locations in the same position. This characteristic enhances the model's local dependency, thereby improving its performance. The MLP classifier exhibits a commendable recognition rate and faster classification speed. However, its training process is not as swift as that of SVM classification, particularly when dealing with extensive training sets. If there is a high requirement for classification efficiency, the MLP method serves as an excellent choice.

The Vision Transformer utilizes a standard Transformer Encoder to carry out detection tasks. By pre-training on substantial amounts of data and transferring the knowledge to multiple small and medium-sized image recognition benchmarks, the Vision Transformer achieves superior results while demanding fewer training resources. Nevertheless, the fruit detection dataset only represents a small sample of the model and fails to meet the requirements of ViT and MLP for comprehensive model datasets.

6.2 Future Work

For our initial experiments, a selection of images depicting apples on a tree with green leaves was chosen. However, the presence of occlusion in the data adversely affects the model's ability to extract fruit features. Therefore, a group of images with low pixel counts and excessive occlusion were deliberately excluded during the one-stage fruit detection process.

Nevertheless, by considering the practical application of fruit detection in agriculture, it is crucial to account for environmental noise. The impact of weather conditions on identification is substantial. Considering the potential influence of hurricanes, heavy rains, and other weather phenomena on the fruit images captured by cameras, expanding our dataset remains necessary.

The impact of fruit stacking on the detection of unpicked fruit trees (specifically apples and pears) must be considered. In the context of agricultural applications, it is necessary to consider the detection of apple stems. The detection of apple stems, being a relatively small target, presents inherent difficulties. In addition, the stacking of apple stems in an orchard setting introduces additional challenges that need to be addressed in future experiments.

6.3 Summary of This Chapter

In our experiments, we successfully accomplish the detection of agricultural targets from digital images, specifically fruits. By leveraging the Transformer and YOLO models, we are able to combine the strengths and fundamental principles of the one-stage model, thereby achieving rapid and real-time detection. This breakthrough lays the groundwork for the automation of agricultural harvesting processes.

References

An, N., Yan, W.: Multitarget tracking using Siamese neural networks. ACM Transactions on Multimedia Computing, Communications and Applications, 17(2s),1-16 (2021).

Amara, J., Bouaziz, B., & Algergawy, A.: A deep learning-based approach for banana Leaf diseases classification. In BTW (Workshops), pp. 79-88 (2017)

Arkin, E., Yadikar, N., Muhtar, Y., & Ubul, K.: A survey of object detection based on CNN and Transformer. In IEEE International Conference on Pattern Recognition and Machine Learning (PRML) (pp. 99-108) (2021)

Basri, H., Syarif, I., & Sukaridhoto, S.: Faster R-CNN implementation method for multi-fruit detection using TensorFlow platform. In IEEE International Electronics Symposium on Knowledge Creation and Intelligent Computing, pp. 337-340 (2018)

Bazame, H. C., Molin, J. P., Althoff, D., & Martello, M.: Detection, classification, and mapping of coffee fruits during harvest with computer vision. Computers and Electronics in Agriculture, 183, 106066 (2021)

Behera, S. K., Rath, A. K., & Sethy, P. K.: Fruits yield estimation using Faster R-CNN with MIoU. Multimedia Tools and Applications, 80(12), 19043-19056 (2021)

Benz, P., Ham, S., Zhang, C., Karjauv, A., & Kweon, I. S.: Adversarial robustness comparison of vision transformer and MLP-mixer to CNNs. arXiv preprint arXiv:2110.02797 (2021).

Bhargava, A., & Bansal, A.: Fruits and vegetables quality evaluation using computer vision: A review. Journal of King Saud University-Computer and Information Sciences, 33(3), 243-257 (2021).

Bodla, N., Singh, B., Chellappa, R., & Davis, L. S.: Soft-NMS—Improving object detection with one line of code. In IEEE International Conference on Computer Vision, pp. 5561-5569, (2017) Buzzelli, M., Belotti, F., & Schettini, R.: Recognition of edible vegetables and fruits for smart home appliances. In International Conference on Consumer Electronics-Berlin (ICCE-Berlin), pp. 1-4 (2018).

Byeon, Y. H., & Kwak, K. C.: A performance comparison of pedestrian detection using Faster RCNN and ACF. In International Congress on Advanced Applied Informatics (IIAI-AAI), pp. 858-863, (2017)

Cai, Z., & Vasconcelos, N.: Cascade R-CNN: Delving into high quality object detection. In IEEE Conference on Computer Vision and Pattern Recognition, pp. 6154-6162, (2018)

Cao, C., Wang, B., Zhang, W., Zeng, X., Yan, X., Feng, Z., … & Wu, Z.: An improved Faster R-CNN for small object detection. IEEE Access, 7, 106838-106846 (2019).

Cardellicchio, A., Solimani, F., Dimauro, G., Petrozza, A., Summerer, S., Cellini, F., & Renò, V.: Detection of tomato plant phenotyping traits using YOLOv5-based single stage detectors. Computers and Electronics in Agriculture, 207, 107757 (2023).

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S.: End-to-end object detection with transformers. In European Conference on Computer Vision (pp. 213-229). Springer, Cham. (2020)

Chen, D., & Wang, H.: Application on intersection classification algorithm based on clustering analysis. In Annual Computer Software and Applications Conference (COMPSAC) ,Vol. 2, pp. 290-297 (2018).

Chen, M., Bai, F., & Gerile, Z.: Special object detection based on Mask R-CNN. In International Conference on Computational Intelligence and Security (CIS) (pp. 128-132). IEEE (2021)

Chen, X., Hsieh, C. J., & Gong, B.: When vision transformers outperform ResNets without pre-training or strong data augmentations. arXiv:2106.01548. (2021)

Dai, Z., Cai, B., Lin, Y., & Chen, J.: Up-DETR: Unsupervised pre-training for object detection with transformers. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1601-1610) (2021)

De Rita, N., Aimar, A., & Delbruck, T.: CNN-based object detection on low precision hardware: Racing car case study. In IEEE Intelligent Vehicles Symposium (IV), pp. 647-652 (2019)

Dias, P. A., Tabb, A., & Medeiros, H.: Multispecies fruit flower detection using a refined semantic segmentation network. IEEE Robotics and Automation Letters, 3(4), 3003-3010 (2018)

Ding, M., Xiao, B., Codella, N., Luo, P., Wang, J., & Yuan, L.: DaViT: Dual attention Vision Transformers. arXiv:2204.03645 (2022).

Dong, E., Lu, Y., & Du, S.: An improved SSD algorithm and its mobile terminal implementation. In IEEE International Conference on Mechatronics and Automation (ICMA), pp. 2319-2324, (2019)

Eaton, A. T.: Fruit injury types recognized in annual new 158 ampshire apple harvest evaluations. Entomology, 13 (2017)

Egi, Y., Hajyzadeh, M., & Eyceyurt, E.: Drone-computer communication based tomato generative organ counting model using YOLOv5 and deep-sort. Agriculture, 12(9), 1290 (2022).

Fachrurrozi, M., Fiqih, A., Saputra, B. R., Algani, R., & Primanita, A.: Content based image retrieval for multi-objects fruits recognition using K-means and K-nearest neighbour. In International Conference on Data and Software Engineering (ICoDSE), pp. 1-6, (2017)

Fachantidis, A., Partalas, I., Taylor, M. E., & Vlahavas, I.: Transfer learning with probabilistic mapping selection. Adaptive Behaviour, 23(1), 3-19 (2015)

Feng, J., Zeng, L., & He, L.: Apple fruit recognition algorithm based on multi-spectral dynamic image analysis. Sensors, 19(4): 949 (2019)

Femling, F., Olsson, A., & Alonso-Fernandez, F.: Fruit and vegetable identification using machine learning for retail applications. In International Conference on Signal-Image Technology & Internet-Based Systems (SITIS) (pp. 9-15). IEEE (2018).

Feng, J., Tan, H., Li, W., & Xie, M.: Conv2NeXt: Reconsidering ConvNeXt network design for image recognition. In International Conference on Computers and Artificial Intelligence Technologies (CAIT) (pp. 53-60) (2022).

Fu, L., Feng, Y., Majeed, Y., Zhang, X., Zhang, J., Karkee, M., & Zhang, Q.: Kiwifruit detection in field images using Faster R-CNN with ZFNet. IFAC-Papers OnLine, 51(17), 45-50 (2018)

Fu, L., Yang, Z., Wu, F., Zou, X., Lin, J., Cao, Y., & Duan, J.: YOLO-Banana: A lightweight neural network for rapid detection of banana bunches and stalks in the natural environment. Agronomy, 12(2), 391 (2022).

Fu, Y., Nguyen, M., Yan, W.: Grading methods for fruit freshness based on deep learning. Springer Nature Computer Science, 3, 264 (2022).

Fu, Y.: Fruit Freshness Grading Using Deep Learning. Master's Thesis. Auckland University of Technology, New Zealand (2020).

Gao, F., Fu, L., Zhang, X., Majeed, Y., Li, R., Karkee, M., & Zhang, Q.: Multi-class fruit-on-plant detection for apple in SNAP system using Faster R-CNN. Computers and Electronics in Agriculture, 176, 105634 (2020)

Gao, F., Fang, W., Sun, X., Wu, Z., Zhao, G., Li, G., … & Zhang, Q.: A novel apple fruit detection and counting methodology based on deep learning and trunk tracking in modern orchard. Computers and Electronics in Agriculture, 197, 107000 (2022).

Gao, J., Dai, S., Huang, J., Xiao, X., Liu, L., Wang, L., ... & Li, M.: Kiwifruit detection method in orchard via an improved light-weight YOLOv4. Agronomy, 12(9),2081 (2022).

Ganesh, P., Volle, K., Burks, T. F., & Mehta, S.: Deep orange: Mask R-CNN based orange detection and segmentation. IFAC-PapersOnLine, 52(30), 70-75 (2019)

Gokhale, A., Chavan, A., & Sonawane, S.: Leveraging ML techniques for image-based freshness index prediction of fruits and vegetables. In IEEE International Conference on Emerging Smart Computing and Informatics (ESCI) (pp. 1-6) (2023).

Gongal, A., Karkee, M., & Amatya, S.: Apple fruit size estimation using a 3D machine vision system. Information Processing in Agriculture, 5(4), 498-503 (2018)

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

Gowdra, N., Sinha, R., MacDonell, S., Yan, W.: Maximum Categorical Cross Entropy (MCCE): A noise-robust alternative loss function to mitigate racial bias in Convolutional Neural Networks (CNNs) by reducing overfitting. Pattern Recognition, 119: 108057 (2021)

Gowdra, N.: Entropy-Based Optimization Strategies for Convolutional Neural Networks. PhD Thesis, Auckland University of Technology, New Zealand (2021).

Guo, L., Lei, Y., Xing, S., Yan, T., & Li, N.: Deep convolutional transfer learning network: A new method for intelligent fault diagnosis of machines with unlabelled data. IEEE Transactions on Industrial Electronics, 66(9), 7316-7325 (2018)

Häni, N., Roy, P., & Isler, V.: A comparative study of fruit detection and counting methods for yield mapping in apple orchards. Journal of Field Robotics, 37(2), 263-282 (2020).

Hameed, K., Chai, D., & Rassau, A.: Score-based mask edge improvement of Mask R-CNN for segmentation of fruit and vegetables. Expert Systems with Applications, 190, 116205 (2022).

Han, Y., Batra, R., Boyd, N., Zhao, T., She, Y., Hutchinson, S., & Zhao, Y.: Learning generalizable vision-tactile robotic grasping strategy for deformable objects via Transformer. arXiv:2112.06374. (2021)

Hameed, K., Chai, D., & Rassau, A.: Score-based mask edge improvement of Mask R-CNN for segmentation of fruit and vegetables. Expert Systems with Applications, 190, 116205 (2022).

Hassanien, M. A., Singh, V. K., Puig, D., & Abdel-Nasser, M.: Predicting breast tumor malignancy using deep ConvNeXt radiomics and quality-based score pooling in ultrasound sequences. Diagnostics, 12(5), 1053 (2022).

He, K., Gkioxari, G., Dollár, P., & Girshick, R.: Mask R-CNN. In IEEE International Conference on Computer Vision (pp. 2961-2969). (2017)

Hou, L., Wu, Q., Sun, Q., Yang, H., & Li, P.: Fruit recognition based on convolution neural network. In International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), pp.18-22 (2016)

Howlader, M. R., Habiba, U., Faisal, R. H., & Rahman, M. M.: Automatic recognition of Guava leaf diseases using deep convolution neural network. In International Conference on Electrical, Computer and Communication Engineering (ECCE), pp. 1-5 (2019)

Hua, X., Wang, X., Rui, T., Zhang, H., & Wang, D.: A fast self-attention cascaded network for object detection in large scene remote sensing images. Applied Soft Computing, 94, 106495 (2020)

Huang, H., Huang, T., Li, Z., Lyu, S., & Hong, T.: Design of citrus fruit detection system based on mobile platform and edge computer device. Sensors, 22(1), 59 (2022).

Huang, H., & Luo, X.: A holistic approach to IGBT board surface fractal object detection based on the multi-head model. Machines, 10(8), 713 (2022).

Huang, Z., Cao, Y., & Wang, T.: Transfer learning with efficient convolutional neural networks for fruit recognition. In IEEE Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 358-362 (2019)

Hussain, M., Al-Aqrabi, H., Munawar, M., Hill, R., & Alsboui, T.: Domain feature mapping with YOLOv7 for automated edge-based pallet racking inspections. Sensors, 22(18), 6927 (2022).

Hsu, S. C., Huang, C. L., & Chuang, C. H.: Vehicle detection using simplified Fast R-CNN. In International Workshop on Advanced Image Technology (IWAIT), pp. 1-3 (2018)

Hsu, S. C., Wang, Y. W., & Huang, C. L.: Human object identification for human-robot Interaction by using Fast R-CNN. In IEEE International Conference on Robotic Computing (IRC), pp. 201-204 (2018)

Islam, M., Dinh, A., Wahid, K., & Bhowmik, K.: Detection of potato diseases using image segmentation and multiclass support vector machine. In Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1-4 (2017)

Jana, S., Basak, S., & Parekh, R.: Automatic fruit recognition from natural images using colour and texture features. In the Conference on Devices for Integrated Circuit (DevIC) (pp. 620-624) (2017).

Jaju, S., & Chandak, M.: A transfer learning model based on ResNet-50 for flower detection. In IEEE International Conference on Applied Artificial Intelligence and Computing (ICAAIC), pp. 307-311 (2022)

Jia, W., Tian, Y., Luo, R., Zhang, Z., Lian, J., & Zheng, Y.: Detection and segmentation of overlapped fruits based on optimized mask R-CNN application in apple harvesting robot. Computers and Electronics in Agriculture, 172, 105380 (2020)

Ji, W., Pan, Y., Xu, B., & Wang, J.: A real-time Apple targets detection method for picking robot based on ShufflenetV2-YOLOX. Agriculture, 12(6), 856 (2022)

Ji, W., Zhao, D., Cheng, F., Xu, B., Zhang, Y., & Wang, J.: Automatic recognition vision system guided for apple harvesting robot. Computers & Electrical Engineering. 38 (5), pp. 1186-1195 (2012)

Ji, Y., Zhang, H., Zhang, Z., & Liu, M.: CNN-based encoder-decoder networks for salient object detection: A comprehensive review and recent advances. Information Science, 546, 835-857 (2021)

Jimenez, A. R., Ceres, R., & Pons, J. L.: A survey of computer vision methods for locating fruit on trees. Transactions of the ASAE, 43(6), 1911 (2000).

Juhnevica-Radenkova, K., Radenkovs, V., & Seglina, D.: Microbiological changes and severity of decay in apples stored for a long-term under different storage conditions. Zemdirbyste Agriculture, 103(4), pp. 391-396 (2016)

Junos, M. H., Mohd Khairuddin, A. S., Thannirmalai, S., & Dahari, M.: Automatic detection of oil palm fruits from UAV images using an improved YOLO model. The Visual Computer, 38(7), 2341-2355 (2022).

Kang, H., & Chen, C.: Fast implementation of real-time fruit detection in apple orchards using deep learning. Computers and Electronics in Agriculture, 168, 105108 (2020).

Kendall, A. G.: Geometry and Uncertainty in Deep Learning for Computer Vision (PhD thesis), University of Cambridge, UK (2019)

Khan, R., & Debnath, R.: Multiclass fruit classification using efficient object detection and recognition techniques. International Journal of Image, Graphics and Signal Processing, 11(8), 1. (2019).

Kim, J. Y., Vogl, M., & Kim, S. D.: A code-based fruit recognition method via image conversion using multiple features. In International Conference on IT Convergence and Security (ICITCS), pp. 1-4 (2014)

KIVRAK, O., & GÜRBÜZ, M. Z.: Performance comparison of YOLOv3, YOLOv4 and YOLOv5 algorithms: A case study for poultry recognition. Avrupa Bilim ve Teknoloji Dergisi, (38), 392-397 (2022).

Koirala, A., Walsh, K. B., Wang, Z., & McCarthy, C.: Deep learning for real-time fruit detection and orchard fruit load estimation: Benchmarking of 'MangoYOLO'. Precision Agriculture, 20(6), 1107-1135 (2019).

Kumar, D., & Kukreja, V.: Image-based wheat mosaic virus detection with Mask R-CNN model. In International Conference on Decision Aid Sciences and Applications (DASA) (pp. 178-182). IEEE (2022)

Kumar, T., & Shivani, R.: Vision Transformer based system for fruit quality evaluation (2022). https://doi.org/10.21203/rs.3.rs-1526586/v1

Kuznetsova, A., Maleva, T., & Soloviev, V.: Using YOLOv3 algorithm with pre-and post-processing for apple detection in fruit-harvesting robot. Agronomy, 10(7), 1016 (2020).

Kuznetsova, A., Maleva, T., & Soloviev, V.: YOLOv5 versus YOLOv3 for apple detection. Cyber-Physical Systems: Modelling and Intelligent Control (pp. 349-358). Springer, Cham (2021)..

Kwon, O. J., Kim, B., & Choi, Y.: Raw produce quality detection with shifted window self-attention. arXiv:2112.13845 (2021).

Latha, R. S., Sreekanth, G. R., Rajadevi, R., Nivetha, S. K., Kumar, K. A., Akash, V., ... & Anbarasu, P.: Fruits and vegetables recognition using YOLO. In IEEE International Conference on Computer Communication and Informatics (ICCCI) pp. 1-6 (2022)

Lal, S., Behera, S. K., Sethy, P. K., & Rath, A. K.: Identification and counting of mature apple fruit based on BP feed forward neural network. In International Conference on Sensing, Signal Processing and Security (ICSSS), pp. 361-368 (2017)

Lee, J., & Hwang, K. I.: YOLO with adaptive frame control for real-time object detection applications. Multimedia Tools and Applications, 81(25), 36375-36396 (2022).

Lee, D., Kim, J., & Jung, K.: Improving object detection quality by incorporating global contexts via self-attention. Electronics, 10(1), 90 (2021)

Lee, Y. H., & Kim, Y.: Comparison of CNN and YOLO for object detection. Journal of Semiconductor & Display Technology, 19(1), 85-92 (2020).

Lian, D., Yu, Z., Sun, X., & Gao, S.: AS-MLP: An axial shifted MLP architecture for vision. arXiv preprint arXiv:2107.08391 (2021).

Li, G., Fu, L., Gao, C., Fang, W., Zhao, G., Shi, F., ... & Cui, Y.: Multiclass detection of kiwifruit flower and its distribution identification in orchard based on YOLOv5l and Euclidean distance. Computers and Electronics in Agriculture, 201, 107342 (2022).

Li, G., Ma, Z., & Wang, H.: Image recognition of grape downy mildew and grape powdery mildew based on support vector machine. In International Conference on Computer and Computing Technologies in Agriculture, pp. 151-162 (2011)

Li, K., Zhai, L., Pan, H., Shi, Y., Ding, X., & Cui, Y.: Identification of the operating position and orientation of a robotic kiwifruit pollinator. Biosystems Engineering, 222, 29-44 (2022)

Li, T., Feng, Q., Qiu, Q., Xie, F., & Zhao, C.: Occluded apple fruit detection and localization with a frustum-based point-cloud-processing approach for robotic harvesting. Remote Sensing, 14(3), 482 (2022)

Liu, B., Zhang, Y., He, D., & Li, Y.: Identification of apple leaf diseases based on deep convolutional neural networks. Symmetry, 10(1), 11 (2017)

Liu, B., Zhao, W., & Sun, Q.: Study of object detection based on Faster R-CNN. In Chinese Automation Congress (CAC), pp. 6233-6236 (2017)

Liu, C., Tao, Y., Liang, J., Li, K., & Chen, Y.: Object detection based on YOLO network. In IEEE Information Technology and Mechatronics Engineering Conference (ITOEC) (2018).

Liu, G., Hou, Z., Liu, H., Liu, J., Zhao, W., & Li, K.: TomatoDet: Anchor-free detector for tomato detection. Frontiers in Plant Science, 13, 942875 (2022).

Liu, G., Nouaze, J. C., Touko Mbouembe, P. L., & Kim, J. H.: YOLO-Tomato: A robust algorithm for tomato detection based on YOLOv3. Sensors, 20(7), 2145 (2020).

Liu, H., Sun, F., Gu, J., & Deng, L.: SF-YOLOv5: A lightweight small object detection algorithm based on improved feature fusion mode. Sensors, 22(15), 5817 (2022).

Liu, J., Pan, C., Yan, W.: Litter detection from digital images using deep learning. Springer Nature Computer Science, 4(2), 134 (2023).

Liu, T. H., Nie, X. N., Wu, J. M., Zhang, D., Liu, W., Cheng, Y. F., ... & Qi, L.: Pineapple (Ananas comosus) fruit detection and localization in natural environment based on binocular stereo vision and improved YOLOv3 model. Precision Agriculture, 1-22 (2022)

Liu, X., Li, G., Chen, W., Liu, B., Chen, M., & Lu, S.: Detection of dense citrus fruits by combining coordinated attention and cross-scale connection with weighted feature fusion. Applied Sciences, 12(13), 6600 (2022).

Liu, X., & Yan, W. Q.: Vehicle-related distance estimation using customized YOLOv7. In IVCNZ (pp. 91-103). Springer Nature Switzerland (2023).

Liu, X.: Vehicle-Related Scene Understanding Using Deep Learning. PhD Thesis, Auckland University of Technology, New Zealand (2024).

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B.: Swin Transformer: Hierarchical vision transformer using shifted windows. In IEEE/CVF International Conference on Computer Vision (pp. 10012-10022) (2021).

Liu, Z., Mao, H., Wu, C. Y., Feichtenhofer, C., Darrell, T., & Xie, S.: A ConvNet for the 2020s. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 11976-11986) (2022).

Liu, Z., Deng, Y., Ma, F., Du, J., Xiong, C., Hu, M., ... & Ji, X.: Target detection and tracking algorithm based on improved Mask R-CNN and LMB. In International Conference on Control, Automation and Information Sciences (ICCAIS) (pp. 1037-1041).

IEEE (2021).

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B.: Swin Transformer: Hierarchical vision transformer using shifted windows. In IEEE/CVF International Conference on Computer Vision (pp. 10012-10022) (2021).

Lu, S., Chen, W., Zhang, X., & Karkee, M.: Canopy-attention-YOLOv4-based immature/mature apple fruit detection on dense-foliage tree architectures for early crop load estimation. Computers and Electronics in Agriculture, 193, 106696 (2022)..

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

Luo, Z., Nguyen, M., Yan, W.: Sailboat detection based on automated search attention mechanism and deep learning models. In International Conference on Image and Vision Computing New Zealand (2021)

Luo, Z., Nguyen, M., Yan, W.: Kayak and sailboat detection based on the improved YOLO with Transformer. In ACM ICCCV, 36-41 (2022)

Luo, Z.: Sailboat and Kayak Detection Using Deep Learning Methods. Masters Thesis, Auckland University of Technology, New Zealand (2022)

Ma, W., Zhang, T., & Wang, G.: Miti-DETR: Object detection based on Transformers with mitigatory self-attention convergence. arXiv preprint arXiv:2112.13310 (2021).

Mai, X., Zhang, H., & Meng, M. Q. H.: Faster R-CNN with classifier fusion for small fruit detection. In IEEE International Conference on Robotics and Automation (ICRA), pp. 7166-7172 (2018)

Mai, X., Zhang, H., Jia, X., & Meng, M. Q. H.: Faster R-CNN with classifier fusion for automatic detection of small fruits. IEEE Transactions on Automation Science and Engineering, 17(3), 1555-1569 (2020)

Manana, M., Tu, C., & Owolawi, P. A.: Preprocessed Faster R-CNN for vehicle detection.

In International Conference on Intelligent and Innovative Computing Applications (ICONIC), pp. 1-4 (2018)

Marathe, A., Anirudh, R., Jain, N., Bhatele, A., Thiagarajan, J., Kailkhura, B., ... & Gamblin, T.: Performance modelling under resource constraints using deep transfer learning. In International Conference on High Performance Computing, Networking, Storage and Analysis, p. 31 (2017).

Mohamud, A. H., & Gopalakrishnan, A. K.: Fruit feature recognition based on unsupervised competitive learning and backpropagation algorithms. In International Conference on Engineering, Applied Sciences, and Technology (ICEAST), pp. 29-32 (2018)

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

Murugan, V., Vijaykumar, V. R., & Nidhila, A.: A deep learning R-CNN approach for vehicle recognition in traffic surveillance system. In International Conference on Communication and Signal Processing (ICCSP), pp. 0157-0160 (2019)

Nourmohammadi-Khiarak, J., Mazaheri, S., Moosavi-Tayebi, R., & Noorbakhsh-Devlagh, H.: Object detection utilizing modified auto encoder and convolutional neural networks. In Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), pp. 43-49 (2018).

Nyarko, E. K., Vidovic, I., Radocia, K., & Cupec, R.: A nearest neighbor approach for fruit recognition in RGB-D images based on detection of convex surfaces. Expert Systems with Applications, 114, pp. 454-466 (2018)

Nguyen, D. T., Nguyen, T. N., Kim, H., & Lee, H. J.: A high-throughput and power-efficient FPGA implementation of YOLO CNN for object detection. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems (2019)

Pan, C., Yan, W.: A learning-based positive feedback in salient object detection. In

International Conference on Image and Vision Computing New Zealand (2018)

Pan, C., Yan, W.: Object detection based on saturation of visual perception. Multimedia Tools and Applications, 79 (27-28), 19925-19944 (2020)

Pan, C., Liu, J., Yan, W., Zhou, Y.: Salient object detection based on visual perceptual saturation and two-stream hybrid networks. IEEE Transactions on Image Processing, 30, 4773-4787 (2021)

Pannerselvam, K.: Adaptive parking slot occupancy detection using Vision Transformer and LLIE. In IEEE International Smart Cities Conference (ISC2) (pp. 1-7). IEEE (2021).

Qi, J., Nguyen, M., & Yan, W.: Waste classification from digital images using ConvNeXt. In Pacific-Rim Symposium on Image and Video Technology (PSIVT) 1-13 (2022)

Qi, J., Nguyen, M., Yan, W.: Small visual object detection in smart waste classification using Transformers with deep learning. In International Conference on Image and Vision Computing New Zealand, 301-314 (2022).

Qi, J., Nguyen, M., Yan, W.: CISO: Co-iteration semi-supervised learning for visual object detection. Multim. Tools Appl. 83(11), 33941-33957 (2024)

Qi, J., Nguyen, M., Yan, W.: NUNI-Waste: Novel semi-supervised semantic segmentation for waste classification with non-uniform data augmentation. Multimedia Tools and Applications (2024)

Rachmawati, E., Supriana, I., & Khodra, M. L.: Toward a new approach in fruit recognition using hybrid RGBD features and fruit hierarchy property. In International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), pp. 1-6 (2017)

Ranftl, R., Bochkovskiy, A., & Koltun, V.: Vision transformers for dense prediction. In IEEE/CVF International Conference on Computer Vision (pp. 12179-12188) (2021)

Rochac, J. F., Zhang, N., Xiong, J., Zhong, J., & Oladunni, T.: Data augmentation for mixed spectral signatures coupled with convolutional neural networks. In IEEE International Conference on Information Science and Technology (ICIST) (2019).

Ruiz, N., Bargal, S., Xie, C., Saenko, K., & Sclaroff, S.: Finding differences between Transformers and ConvNets using counterfactual simulation testing. In Advances in Neural Information Processing Systems, 35, 14403-14418 (2022).

Rzanny, M., Seeland, M., Waldchen, J., & Mader, P.: Acquiring and preprocessing leaf images for automated plant identification: Understanding the tradeoff between effort and information gain. In BMC, 7674 (2017)

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

Saedi, S. I., & Khosravi, H.: A deep neural network approach towards real-time on-branch fruit recognition for precision horticulture. Expert Systems with Applications, 159, 113594 (2020).

Scheffler, O., Coetzee, C., & Opara, U.: A discrete element model (DEM) for predicting apple damage during handling. Biosystems Engineering, 172, pp.29-48 (2018)

Shalini, K., Srivastava, A. K., Allam, S., & Lilaramani, D.: Comparative analysis on deep convolutional neural network models using PyTorch and OpenCV DNN frameworks for identifying optimum fruit detection solution on RISC-V architecture. In IEEE Mysore Sub Section International Conference (MysuruCon) (pp. 738-743). IEEE (2021).

Shen, D., Xin, C., Nguyen, M., Yan, W.: Flame detection using deep learning. In International Conference on Control, Automation and Robotics (2018)

Shen, H., Kankanhalli, M., Srinivasan, S., Yan, W.: Mosaic-based view enlargement for moving objects in motion pictures. In IEEE ICME (2004).

Shi, R., Li, T., & Yamaguchi, Y.: An attribution-based pruning method for real-time mango detection with YOLO network. Computers and Electronics in Agriculture, 169, 105214 (2020).

Shukla, D., & Desai, A.: Recognition of fruits using hybrid features and machine learning. In International Conference on Computing, Analytics and Security Trends (CAST), pp. 572-577 (2016).

Song, H., Sun, D., Chun, S., Jampani, V., Han, D., Heo, B., ... & Yang, M. H.: ViDT: An efficient and effective fully Transformer-based object detector. arXiv:2110.03921 (2021)

Song, W., Wang, H., Maguire, P., & Nibouche, O.: Differentiation of organic and non-organic apples using near infrared reflectance spectroscopy—A pattern recognition approach. Sensors, pp. 1-3 (2016)

Sozzi, M., Cantalamessa, S., Cogato, A., Kayad, A., & Marinello, F.: Automatic bunch detection in white grape varieties using YOLOv3, YOLOv4, and YOLOv5 deep learning algorithms. Agronomy, 12(2), 319 (2022).

Sun, S., Wu, Q., Jiao, L., Long, Y., He, D., & Song, H.: Recognition of green apples based on fuzzy set theory and manifold ranking algorithm. Optik, 165, pp. 395-407 (2018)

Sun, W., & He, Y.: Spatial-chromatic clustering for colour image compression. In IEEE World Congress on Computation Intelligence, pp. 1601-1604 (1998)

Szegedy, S., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A.: Going deeper with convolutions. In IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-9 (2015)

Tang, Y., Zhou, H., Wang, H., & Zhang, Y.: Fruit detection and positioning technology for a *Camellia oleifera C. Abel* orchard based on improved YOLOv4-tiny model and binocular stereo vision. Expert Systems with Applications, 211, 118573 (2023).

Thilagavathi, M., & Abirami, S.: Application of image processing in diagnosing guava leaf diseases. International Journal of Scientific Research and Management, 5(7), pp. 5927-5933 (2017)

Tian, G., Wang, Z., Wang, C., Chen, J., Liu, G., Xu, H., … & Peng, L.: A deep ensemble learning-based automated detection of COVID-19 using lung CT images and Vision Transformer and ConvNeXt. Frontiers in Microbiology, 13 (2022).

Tian, Y., Yang, G., Wang, Z., Wang, H., Li, E., & Liang, Z.: Apple detection during different growth stages in orchards using the improved YOLO-V3 model. Computers and Electronics in Agriculture, 157, 417-426 (2019).

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., ... & Dosovitskiy, A.: MLP-mixer: An all-MLP architecture for vision. In Advances in Neural Information Processing Systems, 34, 24261-24272 (2021).

Tu, S., Pang, J., Liu, H., Zhuang, N., Chen, Y., Zheng, C., ... & Xue, Y.: Passion fruit detection and counting based on multiple scale Faster R-CNN using RGB-D images. Precision Agriculture, 21(5), 1072-1091 (2020).

Tu, S., Xue, Y., Zheng, C., Qi, Y., Wan, H., & Mao, L.: Detection of passion fruits and maturity classification using red-green-blue depth images. Biosystems Engineering, 175, pp. 156-167 (2018)

Ünal, H. B., Vural, E., Savaş, B. K., & Becerikli, Y.: Fruit recognition and classification with deep learning support on embedded system (FruitNet). Innovations in Intelligent Systems and Applications Conference (ASYU), pp. 1-5 (2020).

Ukwuoma, C. C., Zhiguang, Q., Bin Heyat, M. B., Ali, L., Almaspoor, Z., & Monday, H. N.: Recent advancements in fruit detection and classification using deep learning techniques. Mathematical Problems in Engineering, 2022, 1-29 (2022).

Vasconez, J. P., Delpiano, J., Vougioukas, S., & Cheein, F. A.: Comparison of convolutional neural networks in fruit detection and counting: A comprehensive evaluation. Computers and Electronics in Agriculture, 173, 105348 (2020).

Villacrés, J., Viscaino, M., Delpiano, J., Vougioukas, S., & Cheein, F. A.: Apple orchard production estimation using deep learning strategies: A comparison of tracking-by-detection algorithms. Computers and Electronics in Agriculture, 204, 107513 (2023).

Wan, S., & Goudos, S.: Faster R-CNN for multiclass fruit detection using a robotic vision system. Computer Networks, 168, 107036 (2020)

Wang, D., & He, D.: Channel pruned YOLOv5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning. Biosystems Engineering, 210, 271-281 (2021).

Wang, H., Mou, Q., Yue, Y., & Zhao, H.: Research on detection technology of various fruit disease spots based on Mask R-CNN. In IEEE International Conference on Mechatronics and Automation (ICMA) (pp. 1083-1087). IEEE (2020).

Wang, L., Zhao, Y., Xiong, Z., Wang, S., Li, Y., & Lan, Y.: Fast and precise detection of litchi fruits for yield estimation based on the improved YOLOv5 model. Frontiers in Plant Science, 13 (2022).

Wang, L., Yan, W.: Tree leaves detection based on deep learning. International Symposium on Geometry and Vision, 26-38 (2021)

Wang, Q., Chen, J., Deng, J., & Zhang, X.: 3D-CenterNet: 3D object detection network for point clouds with center estimation priority. Pattern Recognition, 115, 107884 (2021).

Wang, Q., & Qi, F.: Tomato diseases recognition based on faster RCNN. In International Conference on Information Technology in Medicine and Education (ITME) (pp. 772-776). IEEE (2019).

Wang, S., Chen, Z., & Ding, Z.: The unified object detection framework with arbitrary angle. In International Conference on Big Data and Information Analytics (BigDIA), pp. 103-107 (2019).

Wang, X., Ma, H., & Chen, X.: Salient object detection via Fast R-CNN and low-level cues. In IEEE International Conference on Image Processing (ICIP), pp. 1042-1046 (2016).

Wang, X., Wang, S., Cao, J., & Wang, Y.: Data-driven based tiny-YOLOv3 method for front vehicle detection inducing SPP-net. IEEE Access, 8, 110227-110236 (2020)

Wang, Z., Jin, L., Wang, S., & Xu, H.: Apple stem/calyx real-time recognition using YOLOv5 algorithm for fruit automatic loading system. Postharvest Biology and Technology, 185, 111808 (2022).

Wu, S., Sun, Y., & Huang, H.: Multi-granularity feature extraction based on Vision Transformer for tomato leaf disease recognition. In International Academic Exchange Conference on Science and Technology Innovation (IAECST) (pp. 387-390). IEEE (2021).

Xia, Y., Nguyen, M., Yan, W.: A real-time Kiwifruit detection based on improved YOLOv7. In IVCNZ, 48-61 (2022)

Xia, Y., Nguyen, M., Yan, W.: Kiwifruit counting using KiwiDetector and KiwiTracker. In Intelligent Systems and Applications, pp. 629–640 (2023)

Xia, Y., Nguyen, M., Yan, W.: Multiscale Kiwifruit detection from digital images. In PSIVT, 82-95 (2023)

Xiao, B.: Apple Ripeness Identification Using Deep Learning. Research Report. Auckland University of Technology, New Zealand (2019)

Xiao, B., Nguyen, M., Yan, W.: Apple ripeness identification using deep learning. In International Symposium on Geometry and Vision (ISGV), pp.53-67(2021).

Xiao, B., Nguyen, M., Yan, W.: Fruit ripeness identification using transformers. Applied Intelligence, 53(19): 22488-22499 (2023)

Xiao, B., Nguyen, M., Yan, W.: A mixture model for fruit ripeness identification in deep learning. Handbook of Research on AI and ML for Intelligent Machines and Systems, IGI Global, 1-21 (2023)

Xiao, B., Nguyen, M., Yan, W.: Apple ripeness identification from digital images using transformers. Multimedia Tools and Applications, Springer 83(3): 7811-7825 (2024)

Xiao, B., Nguyen, M., Yan, W.: Fruit ripeness identification using YOLOv8 model. Multimedia Tools and Applications, 83(9): 28039-28056 Springer (2024)

Xiang, A. J., Huddin, A. B., Ibrahim, M. F., & Hashim, F. H.: An oil palm loose fruits image detection system using Faster R-CNN and Jetson TX2. In International Conference on Electrical Engineering and Informatics (ICEEI) (pp. 1-6). IEEE (2021).

Xu, X., Feng, Z., Cao, C., Li, M., Wu, J., Wu, Z., ... & Ye, S.: An improved Swin Transformer-based model for remote sensing object detection and instance segmentation. Remote Sensing, 13(23), 4779 (2021).

Xue, Y. Yan, W.: YOLO models for fresh fruit classification from digital videos. Handbook of Research on AI and ML for Intelligent Machines and Systems, pp.421-435 IGI Global (2023)

Yan, B., Fan, P., Lei, X., Liu, Z., & Yang, F.: A real-time apple targets detection method for picking robot based on improved YOLOv5. Remote Sensing, 13(9), 1619 (2021).

Yan, J., Wang, H., Yan, M., Diao, W., Sun, X., & Li, H.: IOU-adaptive deformable R-CNN: Make full use of IOU for multiclass object detection in remote sensing imagery. Remote Sensing, 11(3), 286 (2019)

Yan, W.: Computational Methods for Deep Learning: Theory, Algorithms, and Implementations (2nd Edition). Springer (2023).

Yan, W.: Introduction to Intelligent Surveillance: Surveillance Data Capture, Transmission, and Analytics (3rd Edition). Springer (2019).

Yang, C., Hu, Y., Lin, H., Sa, L., Liu, Y.: Overlapped fruit recognition for citrus harvesting robot in natural scenes. In International Conference on Robotics and Automation Engineering (2017)

Yang, R., Hu, Y., Yao, Y., Gao, M., & Liu, R.: Fruit target detection based on BCo-YOLOv5 model. Mobile Information Systems, 1-8 (2022)

Yao, J., Qi, J., Zhang, J., Shao, H., Yang, J., & Li, X.: A real-time detection algorithm for Kiwifruit defects based on YOLOv5. Electronics, 10(14), 1711 (2021).

Yao, J., Wang, Y., Xiang, Y., Yang, J., Zhu, Y., Li, X., ... & Gong, G.: Two-stage detection algorithm for kiwifruit leaf diseases based on deep learning. Plants, 11(6), 768 (2022).

Yu, G., Cai, R., Luo, Y., Hou, M., & Deng, R.: A-pruning: A lightweight pineapple flowers counting network based on filter pruning. SSRN 4196753.

Yu, T., Li, X., Cai, Y., Sun, M., & Li, P.: Rethinking token-mixing MLP for MLP-based vision backbone. arXiv:2106.14882 (2021).

Yu, T., Li, X., Cai, Y., Sun, M., & Li, P.: S2-MLP: Spatial-shift MLP architecture for vision. In IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 297-306) (2022).

Yu, Y., Zhang, K., Yang, L., & Zhang, D.: Fruit detection for strawberry harvesting robot in non-structural environment based on Mask R-CNN. Computers and Electronics in Agriculture, 163, 104846 (2019).

Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., & Lee, B.: A survey of modern deep learning based object detection models. Digital Signal Processing, 103514 (2022).

Zeng, N., Wu, P., Wang, Z., Li, H., Liu, W., & Liu, X.: A small-sized object detection oriented multi-scale feature fusion approach with application to defect detection. IEEE Transactions on Instrumentation and Measurement, 71, 1-14 (2022).

Zhang, F., Gao, J., Zhou, H., Zhang, J., Zou, K., & Yuan, T.: Three-dimensional pose detection method based on keypoints detection network for tomato bunch. Computers and Electronics in Agriculture, 195, 106824 (2022)

Zhang, H., Wang, Y., Liu, Y., & Xiong, N.: IFD: An intelligent fast detection for real-time image information in industrial IoT. Applied Sciences, 12(15), 7847 (2022).

Zhang, L., Lin, H., & Wang, F.: Individual tree detection based on high-resolution RGB images for urban forestry applications. IEEE Access (2022).

Zhang, P., Dai, X., Yang, J., Xiao, B., Yuan, L., Zhang, L., & Gao, J.: Multiscale vision longformer: A new vision transformer for high-resolution image encoding. In IEEE/CVF International Conference on Computer Vision (pp. 2998-3008) (2021).

Zhang, R., Li, X., Zhu, L., Zhong, M., & Gao, Y.: Target detection of banana string and fruit stalk based on YOLOv3 deep learning network. In IEEE International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE) (pp. 346-349). IEEE (2021).

Zhang, W., Wang, J., Liu, Y., Chen, K., Li, H., Duan, Y., ... & Guo, W.: Deep-learning-based in-field citrus fruit detection and tracking. Horticulture Research, 9 (2022).

Zhang, X., Qiao, Y., Meng, F., Fan, C., & Zhang, M.: Identification of maize leaf diseases using improved deep convolutional neural networks. IEEE Access, 6 (2018)

Zhang, X., Wang, Z., Liu, D., & Ling, Q.: DADA: Deep adversarial data augmentation for extremely low data regime classification. In International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2807-2811 (2019)

Zhang, Y., Zhang, Y., & Zhang, Y.: Fruit and vegetable disease identification based on updating the activation function for the ConvNeXt model. In International Conference on Electronic Information Technology and Computer Engineering (pp. 1045-1049) (2022).

Zhang, Z., Gong, Z., Hong, Q., & Jiang, L.: Swin-transformer based classification for rice diseases recognition. In International Conference on Computer Information Science and Artificial Intelligence (CISAI) (pp. 153-156). IEEE (2021).

Zhang, Z., Lu, X., Cao, G., Yang, Y., Jiao, L., & Liu, F.: ViT-YOLO: Transformer-based

YOLO for object detection. In IEEE/CVF International Conference on Computer Vision (pp. 2799-2808) (2021).

Zhao, K., & Yan, W. Q.: Fruit detection from digital images using CenterNet. In International Symposium on Geometry and Visio, 313–326 (2021)

Zhao, K.: Fruit Detection Using CenterNet. Master's Thesis, Auckland University of Technology, New Zealand (2021)

Zhao, S., Chen, H., Wang, C., & Shi, S.: SNCf-Net: Scale-aware neighborhood correlation feature network for hotspot defect detection of photovoltaic farms. SSRN 4193393.

Zheng, H., Wang, G., & Li, X.: Swin-MLP: A strawberry appearance quality identification method by Swin transformer and multilayer perceptron. Journal of Food Measurement and Characterization, 1-12 (2022).

Zhijun, L. I., Shenghui, Y. A. N. G., Deshuai, S. H. I., Xingxing, L. I. U., & Yongjun, Z.: Yield estimation method of apple tree based on improved lightweight YOLOv5. Smart Agriculture, 3(2), 100 (2021).

Zhou, J., Hu, W., Zou, A., Zhai, S., Liu, T., Yang, W., & Jiang, P.: Lightweight detection algorithm of kiwifruit based on improved YOLOX-s. Agriculture, 12(7), 993 (2022).

Zhou, H.: Computational Analysis of Table Tennis Games from Real-Time Videos Using Deep Learning. Master's Thesis, Auckland University of Technology, New Zealand (2022).

Zhou, X., Li, G., Gong, C., Liu, Z., & Zhang, J.: Attention-guided RGBD saliency detection using appearance information. Image and Vision Computing, 95, 103888 (2020)

Zhu, X., Lyu, S., Wang, X., & Zhao, Q.: TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In IEEE/CVF International Conference on Computer Vision, pp. 2778-2788 (2021)

Zhu, Y., Yan, W.: Image-based storytelling using deep learning. ACM ICCCV, pp.179-186 (2022)