agronomy

MDPI

Article

Detection of Male and Female Litchi Flowers Using YOLO-HPFD Multi-Teacher Feature Distillation and FPGA-Embedded Platform

Shilei Lyu [1,2,3], Yawen Zhao [1,2], Xueya Liu [1,2], Zhen Li [1,2,3,*], Chao Wang [4] and Jiyuan Shen [4]

[1] College of Electronic Engineering (College of Artificial Intelligence), South China Agricultural University, Guangzhou 510642, China; lvshilei@scau.edu.cn (S.L.); zhaoyawen@stu.scau.edu.cn (Y.Z.); liuxueya@stu.scau.edu.cn (X.L.)
[2] Pazhou Lab, Guangzhou 510330, China
[3] Division of Citrus Machinery, China Agriculture Research System of MOF and MARA, Guangzhou 510642, China
[4] College of Horticulture, South China Agricultural University, Guangzhou 510642, China; ccwang@stu.scau.edu.cn (C.W.); jyshen@scau.edu.cn (J.S.)
* Correspondence: lizhen@scau.edu.cn

Abstract: Litchi florescence has large flower spikes and volume; reasonable control of the ratio of male to female litchi flowers is the key operational aspect of litchi orchards for preserving quality and increasing production. To achieve the rapid detection of male and female litchi flowers, reduce manual statistical errors, and meet the demand for accurate fertilizer regulation, an intelligent detection method for male and female litchi flowers suitable for deployment to low-power embedded platforms is proposed. The method uses multi-teacher pre-activation feature distillation (MPFD) and chooses the relatively complex YOLOv4 and YOLOv5-l as the teacher models and the relatively simple YOLOv4-Tiny as the student model. By dynamically learning the intermediate feature knowledge of the different teacher models, the student model can improve its detection performance by meeting the embedded platform application requirements such as low power consumption and real-time performance. The main objectives of this study are as follows: optimize the distillation position before the activation function (pre-activation) to reduce the feature distillation loss; use the LogCosh-Squared function as the distillation distance loss function to improve distillation performance; adopt the margin-activation method to improve the features of the teacher model passed to the student model; and propose to adopt the Convolution and Group Normalization (Conv-GN) structure for the feature transformation of the student model to prevent effective information loss. Moreover, the distilled student model is quantified and ported for deployment to a field-programmable gate array (FPGA)-embedded platform to design and implement a fast, intelligent detection system for male and female litchi flowers. The experimental results show that compared with an undistilled student model, the m*AP* of the student model obtained after MPFD feature distillation is improved by 4.42 to 94.21%; the size of the detection model ported and deployed to the FPGA-embedded platform is 5.91 MB, and the power consumption is only 10 W, which is 73.85% and 94.54% lower than that of the detection models on the server and PC platforms, respectively, and it can better meet the application requirements of rapid detection and accurate statistics of male and female litchi flowers.

Keywords: florescence information monitoring; litchi flowers; sep feature distillation; YOLO; FPGA

Citation: Lyu, S.; Zhao, Y.; Liu, X.; Li, Z.; Wang, C.; Shen, J. Detection of Male and Female Litchi Flowers Using YOLO-HPFD Multi-Teacher Feature Distillation and FPGA-Embedded Platform. *Agronomy* **2023**, *13*, 987. https://doi.org/10.3390/agronomy13040987

Academic Editor: Peng Fu

Received: 20 February 2023
Revised: 18 March 2023
Accepted: 24 March 2023
Published: 27 March 2023

1. Introduction

Litchi (*Litchi chinensis Sonn.*) is a specialty fruit tree extensively cultivated in subtropical regions and has a high commercial value due to its nutritious properties and good taste [1,2]. Litchi has a short florescence, large flower spikes, and a large flower volume, among other characteristics. In the same inflorescence, male and female flowers are heterozygous, and the proportion of female flowers is relatively low. Therefore, litchi orchard

management monitors flowering information and regulates the number of flower spikes through the rational application of drugs, water, and fertilizer to improve the proportion of female flowers, create favorable conditions for pollination and fertilization, and enhance the fruiting rate of litchi [3,4]. The scientific management of litchi orchards must solve the necessary technical problems, one of which is how to quickly and accurately collect flowering information.

In recent years, researchers at home and abroad have proposed various solutions based on machine-vision technology to the problems of litchi orchard flower information collection and flower detection. To address the need to detect densely clustered litchi flowers in complex natural environments, Juntao Xiong et al. [5] proposed a deep semantic segmentation network for detecting litchi flowers and leaf pixels to achieve precision segmentation of litchi flowers and leaves in natural environments. Lin et al. [6] developed a model for estimating the number of flower clusters on a single litchi tree using unmanned aerial vehicle images and computer vision technology, combining the YOLOv4 model and equation-fitting. An estimation model of the number of flower clusters and bushes on a single litchi tree was constructed by combining the YOLOv4 model and equation-fitting to achieve automatic statistics of the number of litchi flower clusters and bushes in the natural environment. To better measure the flowering intensity of litchi flowers for yield estimation, Lin et al. [7] combined the number of litchi flowers and their density maps to generate a density map and the number of litchi flowers by using images of male litchi flowers as input to a multicolumn convolutional neural network (CNN) to achieve a method for counting male litchi flowers that is superior to object detection. Ye et al. [8] proposed an aggregation loss, and an aggregation-loss function combining the polyphyletic loss and segmentation loss was proposed to improve the robustness of detecting dense flowers and to detect litchi flowers in dense environments. For other orchard-flowering information collection requirements, Ambrosio et al. [9] and Sun et al. [10] proposed end-to-end residual CNNs and fine-tuned DeepLab–ResNet networks to detect flowers of deciduous fruit trees in natural environments by applying semantic segmentation methods, respectively, to achieve the detection of multi-scene flowers such as those of apples, pears, and peaches. Wu et al. [11] proposed an improved YOLOv4 model that combines the channel pruning method to detect apple flowers in natural environments with different fruit tree species and light directions. Dorj et al. [12] and Lyu et al. [13] applied color detection algorithms and cascade fusion object detection methods, respectively, to design embedded systems capable of automatically detecting citrus flowers in natural environments.

Current research has verified the feasibility of using machine-vision technology to collect information about litchi flowers, fruits, pests, and diseases in natural environments [14–16]. However, there are specific challenges in the statistical analysis of litchi flowering information. On the one hand, the number of litchi flowers and the number of flowers of a single flower spike can reach thousands, and the flowers' precision is small and mutually obscuring [17,18]; on the other hand, affected by variety, tree age, climate, and other conditions, the number of flowers of different litchi orchards differs significantly, and litchi flowering has multiple batches of blooms [19,20]. Therefore, capturing a single moment in the litchi flowering period of the flower-spike flower-volume detection lacks statistical significance. Generally, horticultural experts select several typical flower spikes, use high-density nylon nets for bagging, and manually count the dried litchi flowers after they have all fallen off naturally. Furthermore, to improve statistical efficiency, the collected litchi flowers are manually classified into female or male collections and then weighed to estimate the flower volume and male-to-female ratio, inevitably leading to statistical data errors. Therefore, the study of how to achieve rapid detection and accurate statistics of litchi flowers is consistent with the practical application requirements of litchi orchard flowering information collection. Furthermore, most related researchers are focused on improving the object-detection effect of different types of florescence. However, the computational environment is mainly limited to a server, which is difficult to apply directly to a mobile, embedded platform. In addition, extracting large features is

computationally intensive, and the data communication rate of embedded edge devices is limited, making it challenging to ensure real-time data processing.

Therefore, in this study, a multi-teacher pre-activation feature distillation (MPFD) method is proposed to improve the detection performance of a lightweight object detection model by guiding it to learn complex model features of dried litchi flowers, and the model is suitable for edge device applications [21]. In addition, this study proposes porting object detection models to field-programmable gate array (FPGA) platforms with high parallelism and accelerated CNN models based on embedded technology: namely, computationally capable FPGA platforms that can accelerate the computation of CNN models [22–24]. The main contributions of this study are as follows:

(1) A "dual-teacher–single-student"-based feature distillation MPFD-YOLO training method is proposed to select YOLOv4 and YOLOv5-l, which have relatively complex network structures, as dual-teacher models and YOLOv4-Tiny, which is relatively simple, as student models. The detection performance of the student model is improved by dynamically learning the intermediate feature knowledge of the different teacher models.

(2) The distillation position is optimized before the activation function (pre-activation) to reduce the feature distillation loss; we propose using the LogCosh-Squared function as the distillation-distance loss function to improve the distillation performance and using the margin-activation method to modify the activation function, allowing more positive features of the teacher model to be passed to the student model; the Conv-GN structure is used to transform the features of the student model to prevent the loss of effective information.

(3) Using a hardware and software design approach, the distilled model is format-converted, quantized, compiled, and processed for deployment to a low-power FPGA-embedded platform. A rapid intelligent detection system is designed and implemented for male and female litchi flowers. This study is outlined as follows: Section 2 shows the collection and processing of experimental data; Section 3 introduces the YOLO-MPFD multi-teacher feature-distillation method; Section 4 introduces the porting and deployment methods of the detection model on the FPGA-embedded platform after feature distillation; Section 5 analyzes the experimental results in detail; Section 6 concludes the study.

2. Experimental Data and Processing Methods

Litchi flower data were collected from February to March 2022 at the Litchi Experimental Park, South China Agricultural University (113°36′ N, 23°16′ E; Guangzhou, China), with the "Feizixiao" litchi. Five litchi trees greater than ten years of age were selected. Each tree has more than 200 flower spikes, and every flower spike has thousands of litchi flowers. The early flowers of litchi were bagged using high-density nylon mesh on several typical flower spikes; these were collected at the end of the flowering period in the bag and dried to obtain a sample of dried litchi flowers, as depicted in Figure 1. Considering the small size and light weight of dried litchi flowers, black corrugated paper was used as the sampling background to avoid disorderly disturbance of objectives. Images of dried litchi flowers were taken using a high-definition mobile phone, and LabelImg software (version 1.8.3) was used for manual labeling. In Figure 2, the red box represents a female flower with a larger pollen sac, shorter filaments, and a bifid stigma with a ramshorn shape; the yellow box represents a male flower with a developed receptacle and longer filaments. The final dataset contains 1237 images of dried litchi flowers; the number of male and female flowers is 5627 and 2679, respectively, divided into training and test sets according to the ratio of 8:2, and the data are presented in Table 1.

Table 1. Datasets of male and female litchi flowers.

Tag Name	Training Set	Test Set	Total Number
Male	4502	1125	5627
Female	2143	536	2679

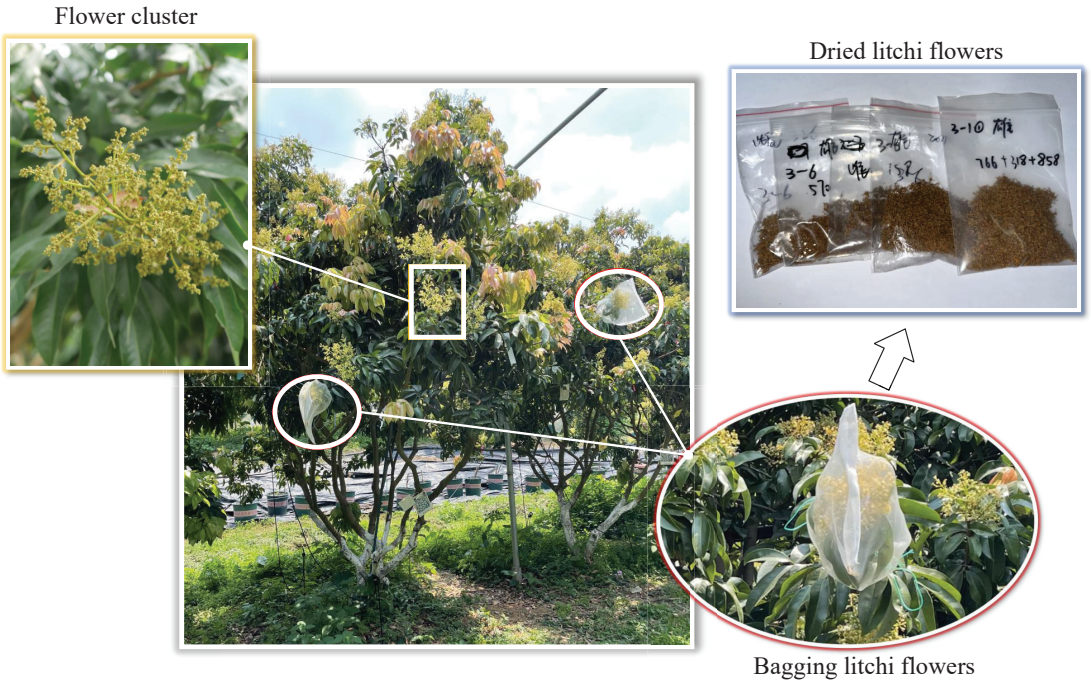

Figure 1. Litchi flower spike sleeve with dried flower sample.

Figure 2. Sample of male and female dried litchi flowers. (**a**) Litchi flowers (Sparse). (**b**) Litchi flowers (Intensive).

3. YOLO-MPFD Multi-Teacher Feature-Distillation Method Design

Traditional knowledge distillation can transfer knowledge from a complex teacher model to a simple student model at the cost of a slight loss of performance. However, only lightweight student models can be trained on the soft objects of the teacher-model output [25,26]. Because the YOLO model is complex, learning only the teacher's output feature knowledge is insufficient. There is a significant capacity difference between the complex teacher model and the simple student model in the middle implicit layer,

which makes them have different feature representation capabilities [27]. Therefore, the YOLO-MPFD feature-distillation method is designed to improve the student model's performance by using the multi-teacher model to select better intermediate feature knowledge suitable for detecting smaller precisions such as litchi flowers. This method addresses the "Gap" between the teacher and student models in terms of capacity and thus enables the implicit layer of the student model to more effectively predict outputs similar to the outputs of the implicit layer of the teacher. Moreover, the approach enables models with high performance to meet requirements such as low power consumption and real-time performance of low-resource devices while not degrading their performance as much as possible but facilitating model portability for edge platforms.

As shown in Figure 3, the YOLO-MPFD multi-teacher feature-distillation method primarily consists of three parts: teacher model, student model, and MPFD feature-distillation loss. Among these, the teacher model consists of YOLOv4 [28] and YOLOv5-l (edition 5.0) [29] with relatively complex networks, and the student model is YOLOv4-Tiny [30] with a relatively simple network.

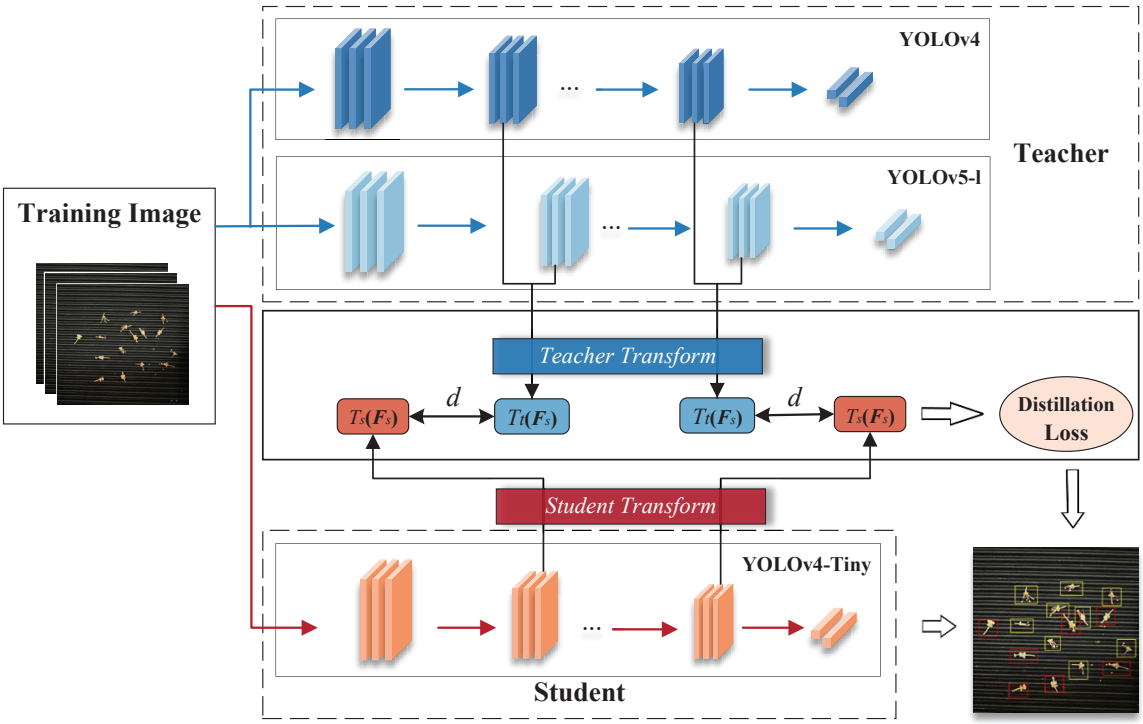

Figure 3. The framework of the YOLO-MPFD feature distillation network model.

The feature distillation approach aims to obtain the optimal student network model performance with minimized task loss and feature-variance penalty. If the features after Teacher Transform are denoted as $\boldsymbol{F_t}$, the features after Student Transform are denoted as $\boldsymbol{F_s}$, and their feature dimensions are denoted as T_t and T_s, respectively; the distance d between $\boldsymbol{F_t}$ and $\boldsymbol{F_s}$ features, i.e., the distillation loss function $L_{distill}$, is shown in Equation (1).

$$L_{distill} = d(T_t(\boldsymbol{F_t}), T_s(\boldsymbol{F_s})) \tag{1}$$

In terms of designing the feature extraction loss of the YOLO-MPFD model and thus improving the distillation performance, the main tasks in this study include the following: (1) For two different teacher models, the distillation loss position is set before the activation function (pre-activation) in the end layer of the backbone network to better analyze and utilize the value of the activation function and thus deliver more positive information. (2) To solve the problem of the distillation position before the activation function, the LogCosh-Squared function is designed as the distillation distance loss function

to screen out the negative information in the activation function and then improve the distillation performance. (3) To solve the problem that teacher models have different activation functions, a margin-activation method applicable to different models is proposed, which not only effectively transmits the positive information of intermediate features of different teacher models but also can activate negative information neurons, thereby minimizing the neuron activation difference between the teacher and student models. (4) To solve the problem of matching the feature dimensions of the teacher and student models, we propose using the Conv-GN structure for the feature transformation of the student model, which can effectively prevent the loss of positive information without reducing the dimensionality of the teacher features.

3.1. Distillation Feature Position Design

Because the two teacher models have more network layers and inconsistent model structures compared to the student models, the most-representative distillation points in the models should be selected as the locations of the feature distillation loss. The two-dimensional image $\boldsymbol{X}$ (matrix) is input into the teacher model $\boldsymbol{F_t}$ and the student model $\boldsymbol{F_s}$, respectively, and the intermediate feature set generated by its location at the precision distillation location is shown in Equation (2).

$$\boldsymbol{T} = \boldsymbol{F}_t(\boldsymbol{X}) \in \mathbb{R}^{C_T \times N}, \boldsymbol{S} = \boldsymbol{F}_S(\boldsymbol{X}) \in \mathbb{R}^{C_T \times N} \tag{2}$$

Generally, the feature maps have the same spatial size N = HW (Height and Width) but can contain different numbers of channels. As depicted in Figure 4, the input size of the image is 640 × 640 × 3, and the backbone of all three models consists of convolutional blocks (CBx) and residual blocks (Resblock). The output channel size of the backbone of the teacher model YOLOv5-l and YOLOv4 (C_T) is 1024, and that of the student model YOLOv4-Tiny (C_S) is 512, which is half that of the teacher model.

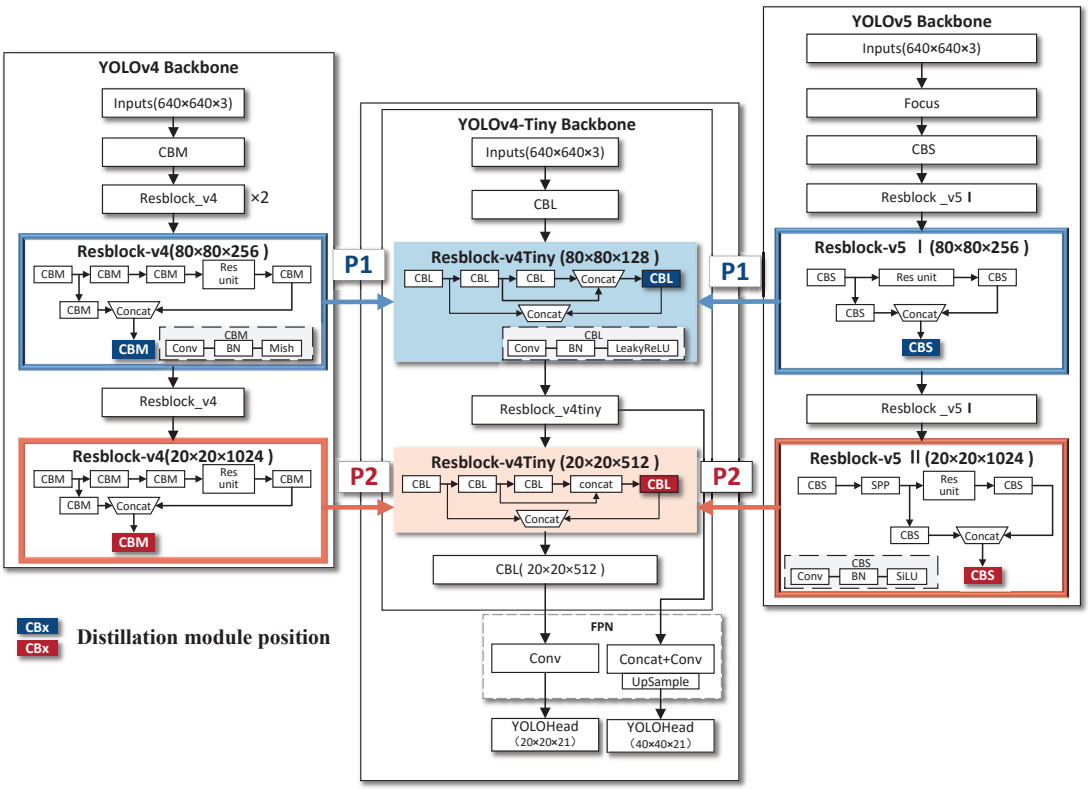

Figure 4. Design framework for feature-distillation position optimization.

Because the student model has two output layers with different scales, two feature distillation locations are connected. In Figure 4, P1 and P2 represent the feature-distillation

locations of the two teacher models corresponding to the student model, both of which are the last model of the residual block. The size of the feature-map space and the number of channels (HWC_T) corresponding to the teacher model are 80 × 80 × 256 and 20 × 20 × 1024, respectively, and the (HWC_S) of the student model are 80 × 80 × 128 and 20 × 20 × 512, respectively. The teacher model channel C_T and the student model channel C_S are multiplicatively related.

The activation function is responsible for processing the output of the upper-layer neurons and transmitting the result to the lower-layer neurons, which plays a crucial role in the neural network. In existing studies, the end layer of the middle-module layer is chosen as the location for feature distillation [31–33]; however, the activation function for the teacher model lacks the necessary consideration. The two teacher models used the Mish and SiLU activation functions, respectively. Both allow positive feature information to pass through but do not consider the neural network's activation boundaries for negative feature information values. Therefore, it is difficult to keep the separation boundary between the teacher and student models consistent. To maximize the retention of teacher-model information during feature distillation, this study sets the distillation position in front of the activation function of the residual module's last convolution block (CBx). The pre-activation position is depicted in Figure 5. This distillation position enables the student model to learn the features passed by the teacher model before the activation function, and the positive and negative values of feature information are therefore retained before the activation function without any deformation, avoiding the problem of valid feature information being discarded after the activation function.

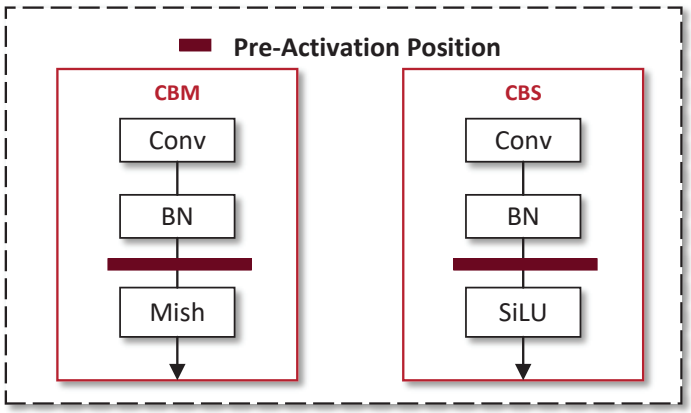

Figure 5. Distillation position before activation function.

3.2. LogCosh-Squared Distance-Loss Function

To obtain the optimal student model $\boldsymbol{F_s}$ that minimizes the training loss value and incorporates a penalty for feature differences, a teacher model $\boldsymbol{F_t}$ with fixed parameters is given, and its training loss is expressed in Equations (3) and (4).

$$L_{task} = L_{location} + L_{confidence} + L_{class} \tag{3}$$

$$Loss = L_{task} + L_{distill} \tag{4}$$

In Equation (4), the main components are task loss L_{task} and distillation loss $L_{distill}$. In Equation (3), the YOLOv4 standard Loss function is used as the task loss function, $L_{location}$ is the location loss, $L_{confidence}$ is the confidence loss, and L_{class} is the categorization loss. Among these, $L_{confidence}$ uses a combination of Ciou Loss and Focal Loss to address the number of difficult and easy samples and the litchi flower data set imbalance [34]. The key to feature distillation is to design the distillation loss $L_{distill}$, as expressed in Equation (5), which ensures the similarity between the intermediate features $\boldsymbol{T}$ and $\boldsymbol{S}$.

$$L_{distill} = d_p(\sigma_T(p(\boldsymbol{T}, \boldsymbol{S})), \sigma_S(\boldsymbol{S})) \tag{5}$$

where σ_T, σ_S, and d_p denote the teacher feature transform, student feature transform, and distance function, respectively. During feature distillation, positive feature information is passed to the student model, but negative feature information after the activation function does not consider the specific activation boundary value of the neural network. Therefore, if the feature response of the student model is higher than that of the teacher model, it is a positive teacher feature, and the distillation loss value should be reduced; otherwise, it is not necessary to increase the value. For the positive teacher characteristics, the LogCosh-Squared of hyperbolic cosine (LogCosh-Squared) is used as the distance loss function, which can reduce the influence of outliers and make the results smoother. Because the distillation position is set before the activation function (pre-activation), to better measure the difference between the teacher and student models and ensure that the feature information can be transferred to the student model after the activation function, we propose using the LogCosh-Squared function applicable to the pre-activation distillation position as the distance loss function, as shown in Equation (6).

$$d_p(\boldsymbol{T}, \boldsymbol{S}) = \sum_{i}^{C} \sum_{j}^{N} \begin{cases} 0 & s_{ij} \leq \mathrm{t_{ij}} \leq 0 \\ \log\left(\cosh\left(\left(t_{ij} - s_{ij}\right)^2\right)\right) & \text{otherwise} \end{cases} \tag{6}$$

Here, for any pair of matrices of the same dimension $\boldsymbol{T}, \boldsymbol{S} \in \mathbb{R}^{\mathrm{C} \times \mathrm{N}}$.

3.3. Teacher Transform

The teacher feature-transformation process converts the teacher's hidden features into an easily transferable form and is an important component of feature extraction. To retain more useful feature-distillation information, this study proposes a teacher feature-transformation method (margin-activation) applicable to different activation functions. Existing studies that use pre-activation as the distillation location only consider the case where the teacher model is a ReLU activation function [35,36]. The expression of the ReLU activation function is $f(x)$ = max(0,1), which can apply only a linear mapping to positive values, and negative values are eliminated and fixed to zero. However, the two teacher models used in this study contain two activation functions, Mish and SiLU, respectively, with expressions:

$$\text{Mish} = x * \tanh(\ln(1 + \mathrm{e}^x)) \tag{7}$$

$$\text{SiLU} = x * \frac{1}{1 + e^{-x}} \tag{8}$$

The activation boundary of a neuron is a separated hyperplane that determines the activation and deactivation of a neuron, whereas the hidden layer of a neural network contains a large amount of information suitable for knowledge transfer; however, it is not easy to achieve perfect transfer due to the high dimensionality and nonlinearity of hidden-layer neurons [35]. Both Mish and SiLU activation functions allow positive feature information greater than zero to pass through, but feature information less than zero does not take into account the specific activation boundary values of the neural network and, therefore, cannot make the separation boundary between the teacher and student networks as consistent as possible. Because the feature values of the teacher $\boldsymbol{F}_t$ are the values before activation, if the value of the teacher network is positive, the student must produce the same value as the teacher. Conversely, suppose that some value on the teacher network is negative. In that case, the student needs to produce a value less than zero to keep the neuron in the same activation state, and this value affects the calculation of the loss-function value. Therefore, this study combines different activation functions and batch normalization (BN) layers [37] and proposes a novel margin-activation method with adaptive changes in activation functions for teacher feature conversion, which preserves positive teacher features and combines activation functions and boundary values that are less than zero after BN layers.

The activation functions Margin-Mish and Margin-SiLU are shown in Equations (9) and (10), respectively. For the stability of training, the margin value m is introduced, where the margin value m is a negative value calculated from the BN layer, and replacing it with the eigenvalues of the teacher model output can retain more distillation features.

$$\text{Margin-Mish} = \begin{cases} x * \tanh(\ln(1 + e^x)) & x \geq 0 \\ m & \text{otherwise} \end{cases} \tag{9}$$

$$\text{Margin-SiLU} = \begin{cases} x * \frac{1}{1+e^{-x}} & x \geq 0 \\ m & \text{otherwise} \end{cases} \tag{10}$$

For channel C and the teacher feature F_t^i of i element, set the boundary value m_c of that channel to the expected value of all training images.

$$m_C = E\left[F_t^i \mid F_t^i < 0, i \in C\right] \tag{11}$$

Because the BN layer is located when the activation function is preceded, it determines the feature F_t^i distribution in a batch, and the feature normalization of each channel of the BN layer can be approximated as a Gaussian distribution with mean μ and variance σ, $F_t^i \sim N(\mu, \sigma)$. Using the F_t^i distribution, we can simply calculate the margin value m_c for that channel, as shown in Equation (12). Therefore, using Equation (12), we can obtain the boundary value m in the channel direction without sampling and training process averaging, which is combined with Equations (9) and (10) to calculate the margin values of different activation functions to form a new activation function and complete the feature transformation of the teacher model.

$$m_C = \frac{1}{Z}\int_{-\infty}^{0} \frac{x}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \mu - \frac{\sigma e^{-\frac{\mu^2}{2\sigma^2}}}{\sqrt{2\pi} \times \Phi\left(-\frac{\mu}{\sigma}\right)} \tag{12}$$

where $\Phi(.)$ denotes the normal distribution cumulative distribution function.

3.4. Student Transform

Typically, feature distillation would set the student conversion T_s to the same function as the teacher conversion T_t [32,38]. However, because the YOLO-MPFD teacher model and the student model do not have the same feature-distillation-position channel size, we propose using a 1×1 convolution for the student feature conversion to increase the feature dimension of the student model to make it consistent with the teacher model rather than reducing the feature dimension, which would result in information loss [35,38,39].

Because the BN operation is performed after each convolutional layer of the student model, the features from the teacher model also need to undergo the normalization operation. When using a multi-teacher model for feature distillation training, the batch size can only be set to a small value because of the numerous calculations required and the limited performance of the server. Nevertheless, BN requires a large batch size to achieve good results. Setting the batch size to a very small value leads to an inaccurate estimation of the batch statistics, thereby significantly increasing the model error. In contrast, group normalization (GN) [40] is calculated independent of the batch size, which divides the channels into groups and calculates the mean and variance within each group for normalization. Therefore, this study appends a GN layer after the 1×1 convolutional layer, using the Conv-GN structure as the student-feature conversion method. This method solves the problem of significant performance degradation of BN in small-batch-size optimization. It allows the student model to better learn the information about the middle-layer features of the teacher model. Experiments have proven that it can achieve better results than traditional BN.

4. Design of Male and Female Litchi Flower Detection System Based on FPGA-Embedded Platform

The FPGA-embedded platform used in this study was a Xilinx Zynq UltraScale+^{TM} MPSoC (AMD, Santa Clara, CA, USA) heterogeneous embedded platform based on the Linaro 96 Boards Consumer Edition specification, model number Ultra96-V2 [41]. The platform consists of FPGA and ARM processing systems, 154 K system logic units, 71 K lookup table units, quad-core ARM Cortex A53 application processors, and dual-core Cortex-R5 real-time processors, as well as a deep-learning processor unit (DPU) that can support different neural network architectures through a reconfigurable hardware architecture.

4.1. FPGA Port Deployment of Feature-Distillation Post Model

The model-porting deployment process after feature distillation is depicted in Figure 6, and the main steps include three operations: model format conversion, quantization compilation, and programming processing, as outlined below.

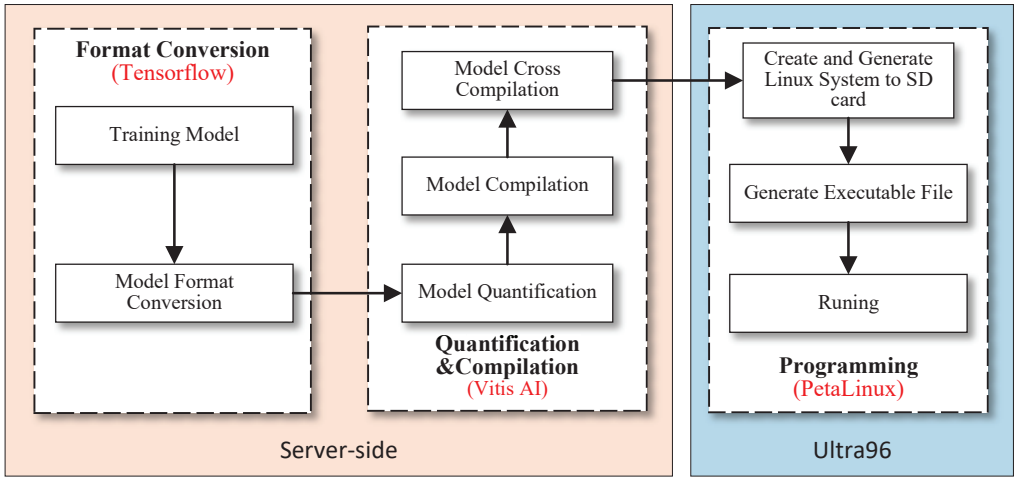

Figure 6. FPGA porting and deployment flow of the detection model of male and female litchi flowers.

The model format conversion operation must be performed in the server platform Tensorflow environment. For the detection model to be quantized in Xilinx Vitis AI (version 1.2), the training-generated Keras weight files and network architecture files need to be converted to binary protobuf files. During the conversion process, the nodes are frozen using the keras_to_tensorflow tool [42] (converting all TF variables into TF constants), and the inference maps and weights are saved to binary protobuf (.pb) files. During freezing, the model is frozen, nodes are pruned, and network nodes that do not affect the output tensor are removed.

The quantization compilation operation needs to be compiled and quantized using the Xilinx Vitis AI tool on the server platform Linux system. Because the use of floating-point operations requires large memory bandwidth and wastes too many resources during FPGA development, the model needs to be quantized by converting the model's 32-bit floating-point numbers into 8-bit integer fixed-point numbers. First, the litchi training set is used as the calibration dataset, combined with the converted format protobuf file, and after starting the Xilinx Vitis AI Docker, the Tensorflow environment is activated, and the model is fixed-point quantized using the quantizer tool. Next, according to the FPGA-embedded platform model, the convolutional structure of model B2304 is selected for deploying and accelerating the algorithm model. The quantized model is compiled using a compiler to output the instruction file (elf file) and DPU Kernel information required for the DPU. Finally, the Xilinx Vitis AI tool output elf file is processed using a cross-compilation tool to generate a Linux dynamic link library so file, which is ported to the Ultra96-V2 embedded platform using an SD card.

In the programming and processing part of the embedded platform, this study uses the PetaLinux tool to build an embedded Linux system. A Python program was written to pre-process the litchi flower images using the OpenCV library, DPU library functions, and APIs on the ARM side and to call DPU to realize the real-time detection and statistics of male and female litchi flowers.

4.2. Male and Female Litchi Flower Detection System Design

The male and female litchi flower detection system is configured with USB, Micro SD, DDR, Mini MDPI, and other interfaces, as well as a Uart interface for debugging, and a slow vibration platform is used to separate sticky dried litchi flowers. The workflow of the detection system is depicted in Figure 7. The main control platform consists of two parts–namely, the ARM processing subsystem (PS, Processing System) and FPGA programmable logic (PL, Programmable Logic)–with the two sides using the AXI bus to achieve high-speed communication. The PS side acquires the image of a dried litchi flower through the camera and completes the real-time processing of image data on the embedded Linux system, including image acquisition, storage, reading, scaling format conversion, and display of processing results. The PL side acquires the processed images, detection model weights and biases, and related instructions through the AXI bus and calls the DPU to achieve accelerated inference of the detection model.

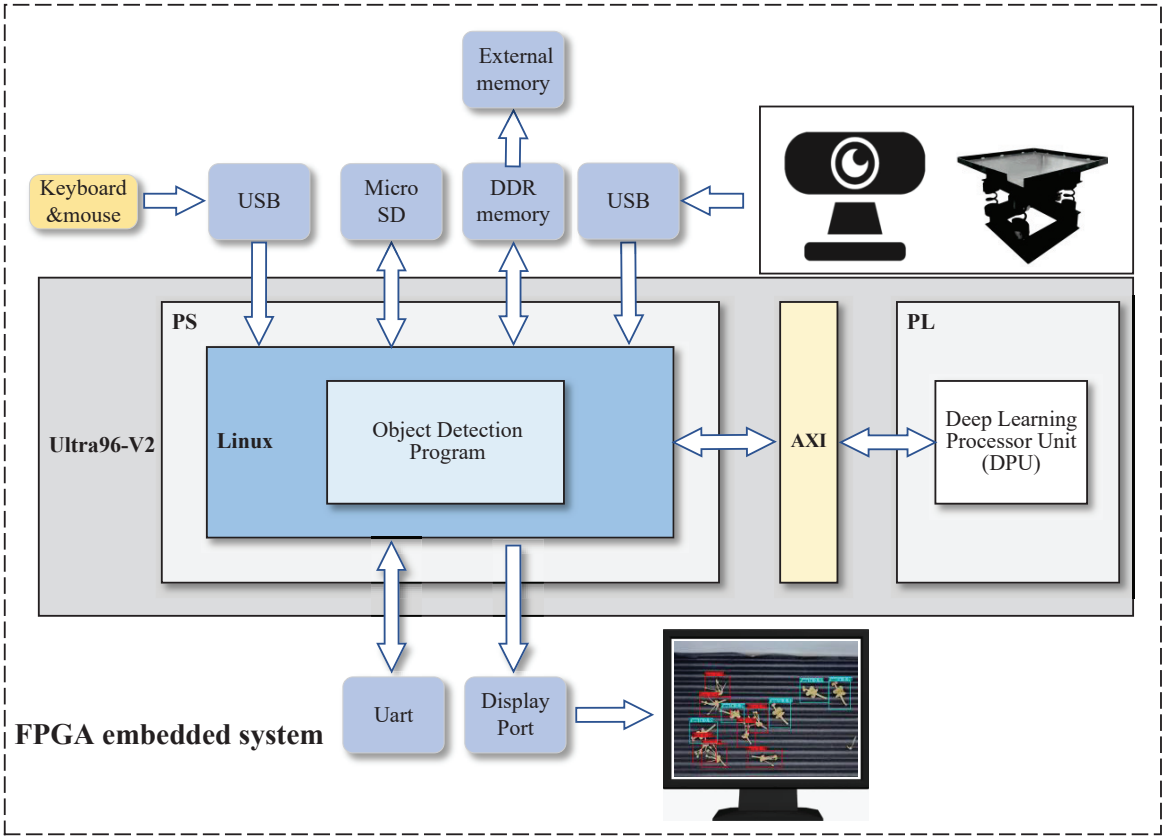

Figure 7. Workflow of male and female litchi flower detection system.

5. Experimental Results and Analysis

To further analyze the system performance of the male and female litchi flower detection model after feature distillation and its transplantation to the FPGA platform, this study combined YOLO-MPFD multi-teacher feature distillation with various improvement strategies to conduct experiments and analyze data results based on different test platforms such as server, PC, and FPGA, and the configuration environment of each test platform is presented in Table 2. *F*1 measure, mean average precision (m*AP*), and Recall are used

as the evaluation metrics of model detection accuracy, as shown in Equations (13)–(16). Frames per second (FPS) is used as the evaluation metric of the model detection speed.

$$P = \frac{T_P}{T_P + F_P} \times 100\% \tag{13}$$

$$R = \frac{T_P}{T_P + F_N} \times 100\% \tag{14}$$

$$F1 = \frac{2PR}{P + R} \times 100\% \tag{15}$$

$$\mathrm{m}AP = \int_0^1 P(R)dR \times 100\% \tag{16}$$

where P denotes the precision, R the recall, T_P the number of true-positive samples, F_P the number of false-positive samples, and F_N the number of false-negative samples.

Table 2. Configuration environment for different test platforms.

No.	Platform	System	Configuration	Operating Environment
1	Server	Windows 10	Intel Core i9-10900K @ 3.70 GHz Ten-core CPU, 32 GB RAM, Nvidia GeForce RTX 3080 (10 GB) GPU	The test framework is Tensorflow and Keras, using CUDA parallel computing framework with CUDNN deep neural network acceleration library
2	PC	Windows 10	Intel Core i7-8500U @ 1.80 GHz Four-core CPU, 16 GB RAM, Nvidia GeForce MX 450 (4 GB) GPU	
3	FPGA	Linux	Xilinx Zynq UltraScale+ MPSoC EV (Ultra96 SoC)	Compilation environment uses OpenCV and Xilinx AI runtimes

5.1. Analysis of Model Test Results

To analyze the performance of the MPFD feature-distillation model, first, training of 150 epochs is performed for YOLOv4 and YOLOv5-l in turn, and the trained model is used as the teacher model in the feature distillation process. Based on the same training set and testing environment, the performances of the undistilled YOLOv4-Tiny and the model after MPFD feature distillation are tested with the following parameters: an input image size of 640 × 640, a training process with reverse gradient, and a validation process without reverse gradient; each model was trained for 150 epochs. In order to compare performance more fairly with respect to algorithm improvement, the test set in this paper is changeless. In the training process, the repeat holdout cross-validation method is used. For each training, the data in the training set are randomly divided into an 80% training set and a 20% validation set. This process repeats five times, the precision, Recall and F1 values of each training are recorded, and the average of the five times is collated as the evaluation metric. In this case, the trained YOLOv4-Tiny model is used as a pre-trained model in the feature-distillation training process. The YOLOv4 standard loss function (Equation (3)) is used as the training set and test performance metric for the YOLOv4-Tiny model and the validation metric for the distillation model. The proposed distillation loss (Equation (4)) is used as a training metric for the distillation model MPFD YOLOv4-Tiny, and the training process is performed on the server. The variation curves of the model loss values after YOLOv4-Tiny and distillation are shown in Figure 8, and the model results are compared in Table 3.

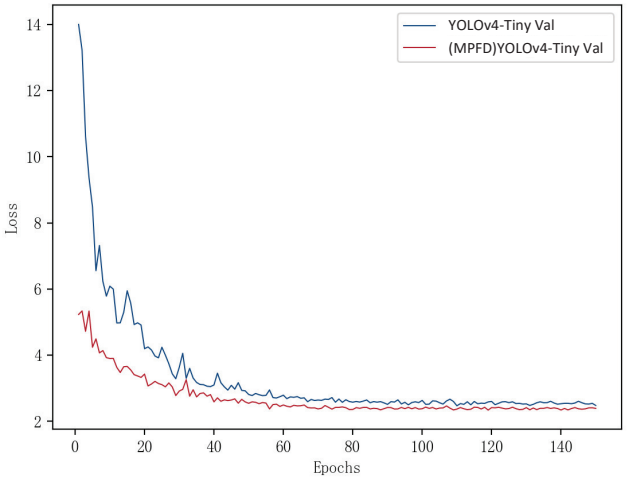

Figure 8. Validation set loss values.

Table 3. Feature distillation training results.

Model	Size/MB	m*AP*/%	*Recall*/%	*F*1/%
(Teacher1) YOLOv4	244.97	94.24	80.74	86.54
(Teacher2) YOLOv5-l	178.94	94.49	82.50	87.75
(Student)YOLOv4-Tiny	22.60	89.79	73.85	84.01
(MPFD) YOLOv4-Tiny	22.60	94.21	80.90	87.36

The change in the loss function of the neural network model during the training process can reflect the performance of the completed training model to some extent. Because the distilled model uses the YOLOv4-Tiny model as the pre-trained student model, the validation set loss value in the pre-training period of the undistilled YOLOv4-Tiny model is higher than that of the distilled model, and the overall fluctuation is more remarkable. However, the convergence speed is faster, and the validation loss value of the distilled model is lower than that of the undistilled YOLOv4-Tiny model. The overall loss value is lower, and the fit is better, which proves that MPFD feature distillation can improve the training effect of the model. Finally, the validation loss value of the undistilled student model is stable at approximately 2.8, and the validation loss value of the distilled model is stable at approximately 2.5 with only slight oscillation, which indicates the end of model fitting and proves that MPFD feature distillation reduces the model training loss value.

As presented in Table 3, the student model improved in all evaluation metrics of model precision after MPFD feature distillation. Among them, the YOLOv4 and YOLOv5-l teacher models have similar detection precision with m*AP* values of 94.24% and 94.49%, respectively, with only a 0.25% difference between them. In contrast, the m*AP* value of the undistilled YOLOv4-Tiny model is only 89.79%. However, the sizes of the two teacher models, YOLOv4 and YOLOv5-l, are 244.97 and 178.94 MB, respectively, which are 10.84 and 7.92 times larger than that of the student model, YOLOv4-Tiny (22.60 MB), respectively, and neither of them is suitable for deployment to low-power embedded platforms. The m*AP* value of the MPFD YOLOv4-Tiny model after feature distillation reaches 94.21%, which is 4.41% higher than that of the undistilled YOLOv4-Tiny model and is very close to the m*AP* value of the teacher model YOLOv4, and the Recall and *F*1 values are 80.90% and 87.36%, respectively, which are 7.05% and 3.35% higher than those of the undistilled YOLOv4-Tiny. Therefore, MPFD feature distillation can improve the detection index of the student model without changing the model size, making it close to or better than the teacher model performance.

The Precision–Recall (PR) curve reveals the relation between precision and recall, and the larger the area it covers, the better the detection result is. In order to evaluate the

classification effect of the distilled model, the *PR* curve and *AP* are used in this paper. The *PR* curves of the distilled model are shown in Figure 9, with *P* representing Precision and *R* representing Recall, and the confidence interval is 0.5. The *AP* values of litchi male and female flowers can be obtained by calculating the area of the shaded part of the *PR* curve. Since five cross-validations were performed, the *AP* values of litchi male and female flowers after each training are shown in the figure and averaged separately. The *AP* values of litchi male and female flowers were 92.22% and 96.20%; the change trend of the two *PR* curves is relatively flat, indicating that the two types of object detection are balanced. Therefore, the detection results meet the requirements of litchi flower male and female detection.

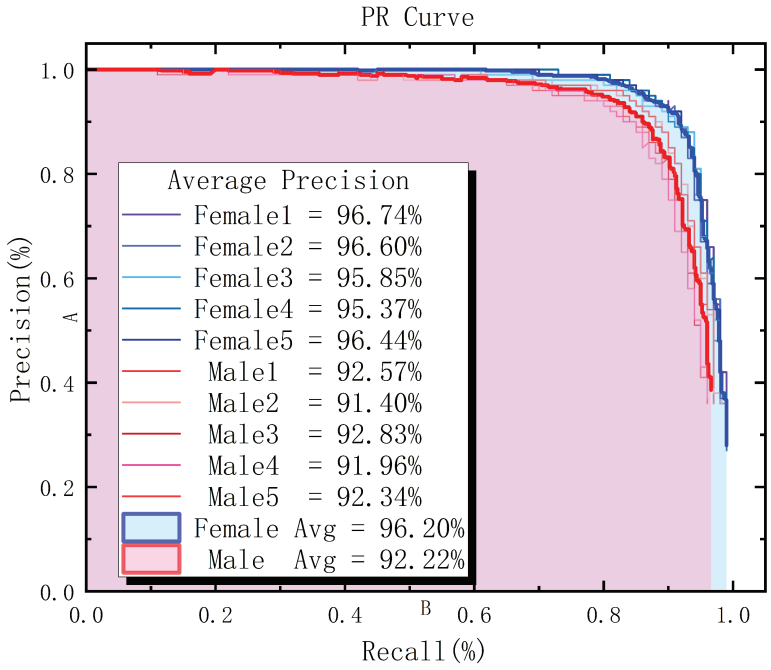

Figure 9. *PR* curves of litchi female and male flowers for the distilled model.

5.2. Ablation Study

To further analyze the performance of different improvement strategies, this study sequentially tests the performance of different combinations of improvement strategies for YOLO-MPFD feature distillation using the same training set, training method, and testing environment. Among them, the base feature distillation method (Baseline) uses YOLOv4 or YOLOv5-l as the single-teacher model and YOLOv4-Tiny as the student model. The L2 function in FitNets [39] is used as the distance loss function, the convolution + BN is used as the student transformation, and the feature distillation loss position is set before the activation function. The test results are presented in Table 4.

Table 4. Results of ablation experiments.

Method	m*AP*/%	*Recall*/%	*F*1/%
YOLOv4-Tiny	89.79	73.85	84.01
Baseline (Teacher1)	91.89	76.02	84.92
Baseline (Teacher2)	92.21	76.05	85.05
+Multiple Teacher	92.67	77.18	85.58
+Loss	93.13	78.47	86.36
+Margin-Activation	93.68	79.31	86.67
+Conv-GN	94.21	80.90	87.36

As shown in Table 4, by using YOLOv4 (Teacher1) or YOLOv5-l (Teacher2) as a single-teacher model and training the student model YOLOv4-Tiny using the base feature distillation method (Baseline), respectively, the m*AP* values of the trained student models improved by 2.10% and 2.42%. Moreover, because YOLOv4 and YOLOv4-Tiny have a more-similar model structure than YOLOv5 and YOLOv4-Tiny, which is more conducive to the student model learning the feature map knowledge in the implicit layer of the teacher model, the training result of the teacher model YOLOv4 is slightly better than that of YOLOv5-l. The category "+Multiple Teacher" indicates that a two-teacher model was used instead of a single-teacher model. The m*AP*, Recall, and *F*1 of the student model trained by the multi-teacher model are 92.67%, 77.18%, and 85.58%, respectively. All metrics are improved compared with the training results of the single-teacher model. This is because the multi-teacher model could provide more explanatory information for the detection task of the student model. The student model can take advantage of the "Views" of the teacher model on the precision task and combine the respective strengths of the different teacher models to improve its model performance. The category "+Loss" indicates that the LogCosh-Squared function is used as the loss function, which aims to reduce the influence of outliers and smooth the training results, as well as prevent the transmission of invalid negative information; "+Margin-Activation" indicates that the margin-activation method is used to modify the activation functions of different teacher models, thereby retaining more useful feature distillation information; "+Conv-GN" indicates the use of the Conv-GN structure for the feature distillation of student models, which aims to address the performance degradation of BN in small batch optimization. The data in Table 4 show that various improvement strategies have a positive effect on the training results of the detection model. After applying various improvement strategies together, the m*AP* of the MPFD YOLOv4-Tiny model is 94.21%, which is an improvement of 2.32% (Teacher1) and 2.00% (Teacher2) over the training results of the two single-teacher models, respectively.

To further prove the performance of YOLO-HPFD multi-teacher feature distillation, the distilled model is compared to other lightweight versions of YOLO in the same test environment. The experiment uses the same training and test sets for model training and accuracy testing in TensorFlow (GPU). Speed tests are performed on the same picture with input sizes of 640 × 480. The experimental results are shown in Table 5. From the empirical observations, it can be noted that the distilled model has many distinct benefits over other models. The m*AP*, Recall, and test speeds of YOLOv4-Tiny-MPFD are 94.21%, 80.90% and 17.93ms–all better than the other models. In addition, although the model size of YOLOX-Tiny is the smallest, its Recall is the worst among all models, and it is 12.66 ms slower than YOLOv4-Tiny-MPFD. Thus, the distilled model has the best overall performance and is more compatible with the requirements of the embedded platform.

Table 5. Performance of lightweight models.

Model	Size/M	m*AP*/%	*Recall*/%	Test Time/ms
YOLOv4-Tiny [30]	22.60	89.79	73.85	38.22
YOLOv5-s [31]	27.28	91.06	70.87	58.50
YOLOv5-s-6.1 [43]	27.11	92.60	76.98	52.39
YOLOX-Tiny [44]	19.34	90.60	69.81	50.88
YOLOv7-Tiny [45]	23.43	92.09	76.22	42.44
YOLOv4-Tiny (MPFD)	22.60	94.21	80.90	38.22

5.3. Visual Analysis of Male and Female Litchi Flower Test Results

The detection results of male and female litchi flowers by applying the YOLOv4-Tiny model after MPFD feature distillation in the server platform are depicted in Figure 10. Among them, male litchi flowers (blue detection box) have well-developed receptacles and longer filaments, whereas female litchi flowers (red detection box) have larger pollen sacs and shorter filaments, and the stigma is bifid and ramshorn-shaped. Note that there is a problem with filament shedding in the process of separating litchi flowers using the

slow-vibration platform (Figure 10 in the white round box for the shedding of filaments), and even the female flowers of the pollen sacs fall off, which not only results in more diverse male and female flowers but also has a significant negative impact on the detection of litchi flowers. The test results show that both male and female litchi flowers can be successfully detected and classified, and only a tiny portion of the filaments fall off due to missed detection (yellow box in Figure 10), which can meet the needs of rapid detection of litchi flowers.

(a) **(b)**

Figure 10. Test results of the model for the detection of male and female litchi flowers after distillation. (**a**) Sparse object. (**b**) Intensive object.

5.4. Analysis Results of the Distilled Model

The performance tests are conducted by applying the distilled detection model to the same dataset on the server and FPGA platforms, respectively. The data results are presented in Table 6. The detection precision of female litchi flowers in different testing platforms is better than that of male litchi flowers. This is because compared with female litchi flowers, male litchi flowers are smaller in size, and filaments are more likely to fall off or even appear to have only the receptacle left, all of which make the detection of male flowers more difficult. In addition, the distilled model quantization is ported to the FPGA platform. The m*AP* value was 93.74%, which is 0.47% lower than that on the server platform. Because the FPGA platform uses a DPU for neural network acceleration, it needs to convert 32-bit floating-point operations into 8-bit fixed-point operations. There is a certain precision loss in this conversion process. However, the size of the detection model ported to the FPGA platform is only 5.91 MB, which is 73.85% lower than that of the detection model of the server platform. Therefore, the porting operation provides significant compression of the model size with no significant reduction in the detection precision of litchi flowers, which improves the real-time detection rate.

Table 6. Comparison of test results of different test platforms.

Platform	*AP*/% Male	*AP*/% Female	m*AP*/%	Model Size/MB
Servers	92.22	96.20	94.21	22.60
FPGA	91.71	95.76	93.74	5.91

To verify the real-time detection of the system, the detection effects of the server, PC, and FPGA platforms on video streams are also compared and tested. The video stream testing scheme, based on the FPGA platform, is depicted in Figure 11. The camera is connected to the Ultra96-V2 through the USB interface and captures the video stream of the litchi flower in real time. The monitor is connected to Ultra96-V2 through an HDMI interface and displays the real-time detection results processed by Ultra96-V2. The input

size of the model is 640 × 640, the video stream resolution is 640 × 480, and a power-metering socket is used to monitor the power consumption of different test platforms. The data results are shown in Table 7. Among them, the server platform has the fastest video stream detection rate, based on the GPU/RTX 3080 detection rate of 26 FPS. However, its power consumption reached 183 W, far exceeding the PC and FPGA platform power consumption. The FPGA platform has a video stream detection rate of 6 FPS, which is better than that of the PC platform. In addition, the FPGA platform's power consumption is only 10 W, 94.54% and 82.14% lower than that of the server (GPU) and PC (GPU) platforms, respectively.

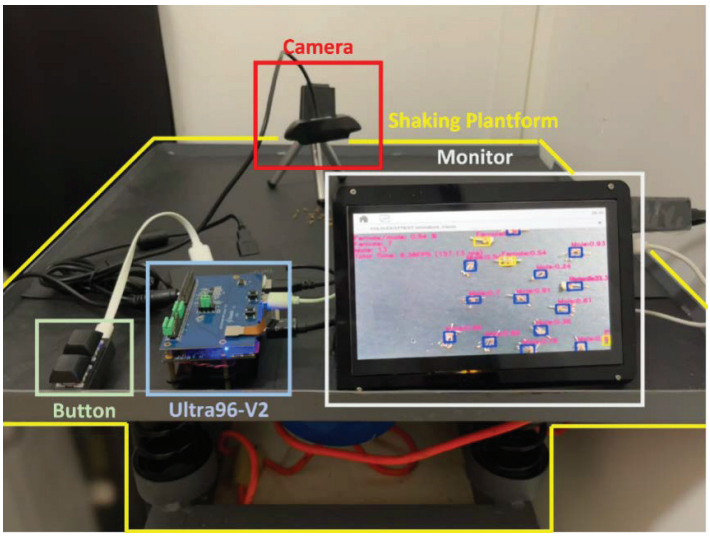

Figure 11. Video streaming test based on FPGA platform.

Table 7. Comparison of real-time detection efficiency of different test platforms.

Platform	Configuration	Detection Rate/FPS	Power/W
Server	CPU/core i9-10900K (3.70 GHz deca-core)	11	164
	GPU/RTX 3080 (10 GB)	26	183
PC	CPU/core i7-8550U (1.8 GHz quad-core)	5	38
	GPU/MX150 (4 GB)	4	56
FPGA	Ultra96-V2	6	10

To verify the practicality of the male and female litchi flower detection system, this study also conducts a comparison experiment between system detection and manual weighing. Considering that the manual weighing method needs to obtain the exact weight of litchi flowers, this study distinguishes between female and male litchi flowers manually after randomly selecting five groups of samples. The number of female and male litchi flowers in each group is 50, and then the weight of a single male and female flower is estimated using high-precision electronic scales. The experimental comparison samples consist of five groups. Considering the actual proportion of male and female flowers in a single spike, the number of male flowers in each group ranges from 200 to 400, whereas that of female flowers ranges from 30 to 60. The data results (Table 8) show that the average accuracy of the system detection is 93.25%, which is significantly better than the manual weighing method in terms of detection efficiency and precision.

Table 8. System testing vs. manual weighing results.

No.	Number of Male Flowers	Male Flower Weight/g	Number of Female Flowers	Female Flower Weight/g	Manual Weighing Precision	System Detection Precision
1	268	0.47	37	0.16	87.09%	92.78%
2	324	0.56	58	0.26	88.04%	93.37%
3	287	0.50	43	0.18	85.42%	93.72%
4	219	0.39	32	0.14	88.27%	93.45%
5	353	0.60	41	0.18	86.40%	92.94%
Total	1451	2.52	211	0.92	87.04%	93.25%

6. Conclusions

The YOLO-MPFD feature-based distillation method for detecting male and female litchi flowers is proposed. Moreover, the distillation position before the activation function (pre-activation) is optimized in the distillation loss design. The LogCosh-Squared distillation distance function, margin-activation for teacher transformation to extract features method, and Conv-GN structure based on student transformation to extract features method are also integrated to improve the feature distillation learning effect of the student model. The experimental results show that the m*AP* and *F*1 of the distilled student model YOLOv4-Tiny reach 94.21% and 87.36%, respectively, which is an improvement of 4.42% and 3.29%, respectively, compared with the original detection model. This study designed a fast detection system for male and female litchi flowers based on an FPGA-embedded platform through detection-model porting and deployment. The detection precision of male and female litchi flowers reaches 91.71% and 95.76%, respectively. The size of the quantized and compiled detection model is only 5.91 MB, and the power consumption is 10 W, which has the characteristics of low power consumption and easy portability. This model can meet the demand for rapid detection and accurate statistics of male and female litchi flowers and provides reliable data support for orchard flowering information monitoring.

In addition, this research still needs improvement; for example, FPGA computing resources are not fully utilized in this study. The system's real-time performance and platform manipulation should be improved. Different varieties of litchi flowers have basic forms, such as corolla, calyx and stamens, which are morphologically similar. In the next step, the authors will expand the dataset and conduct experiments on the detection and statistics of different species of male and female litchi flowers to enhance the robustness of the model. Furthermore, a data acquisition scheme with a MIPI camera connected to the PL side of the FPGA platform will be considered to improve the stability and real-time performance of the system and give full play to the computational advantages of FPGAs.

Author Contributions: Conceptualization, S.L. and Z.L.; methodology, S.L. and Y.Z.; software, Y.Z.; validation, S.L., Y.Z. and X.L.; formal analysis, X.L.; data curation, S.L. and Y.Z.; writing—original draft preparation, S.L. and Y.Z.; writing—review and editing, Z.L.; visualization, J.S. and C.W.; supervision, Z.L.; funding acquisition, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (32271997, 31971797); General Program of Guangdong Natural Science Foundation (2021A1515010923); Special Projects for Key Fields of Colleges and Universities in Guangdong Province (2020ZDZX3061); China Agriculture Research System of MOF and MARA (CARS-26); and Basic and Applied Basic Research Project of Guangzhou Basic Research Plan in 2022 (202201010077).

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the anonymous reviewers for their criticism and suggestions. We would also like to thank Jianqiang Lu for research data support.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Li, H.; Huang, D.; Ma, Q.; Qi, W.; Li, H. Factors influencing the technology adoption behaviours of litchi farmers in China. *Sustainability* **2019**, *12*, 271. [CrossRef]
2. Zhao, L.; Wang, K.; Wang, K.; Zhu, J.; Hu, Z. Nutrient components, health benefits, and safety of litchi (*Litchi chinensis Sonn.*): A review. *Compr. Rev. Food Sci. Food Saf.* **2020**, *19*, 2139–2163. [CrossRef]
3. Ding, F.; Li, H.; Wang, J.; Peng, H.; Chen, H.; Hu, F.; Lai, B.; Wei, Y.; Ma, W.; Li, H.; et al. Development of molecular markers based on the promoter difference of LcFT1 to discriminate easy-and difficult-flowering litchi germplasm resources and its application in crossbreeding. *BMC Plant Biol.* **2021**, *21*, 539. [CrossRef] [PubMed]
4. Liu, S.C.; Lin, J.T.; Wang, C.K.; Chen, H.Y.; Yang, D.J. Antioxidant properties of various solvent extracts from lychee (*Litchi chinenesis Sonn.*) flowers. *Food Chem.* **2009**, *114*, 577–581. [CrossRef]
5. Xiong, J.; Liu, B.; Zhong, Z.; Chen, S.; Zheng, Z. Litchi flower and leaf segmentation and recognition based on deep semantic segmentation. *Trans. Chin. Soc. Agric. Mach.* **2021**, *52*, 252–258. [CrossRef]
6. Lin, P.; Li, D.; Jia, Y.; Chen, Y.; Huang, G.; Elkhouchlaa, H.; Yao, Z.; Zhou, Z.; Zhou, H.; Li, J.; et al. A novel approach for estimating the flowering rate of litchi based on deep learning and UAV images. *Front. Plant Sci.* **2022**, *13*, 3001. [CrossRef]
7. Lin, J.; Li, J.; Yang, Z.; Lu, H.; Ding, Y.; Cui, H. Estimating litchi flower number using a multicolumn convolutional neural network based on a density map. *Precis. Agric.* **2022**, *23*, 1226–1247. [CrossRef]
8. Ye, J.; Wu, M.; Qiu, W.; Yang, J.; Lan, W. Polyphyletic Loss: Litchi Flower Detection with Occlusion. *Proc. J. Phys. Conf. Ser.* **2022**, *2171*, 012041. [CrossRef]
9. Dias, P.A.; Tabb, A.; Medeiros, H. Multispecies fruit flower detection using a refined semantic segmentation network. *IEEE Robot. Autom. Lett.* **2018**, *3*, 3003–3010. [CrossRef]
10. Sun, K.; Wang, X.; Liu, S.; Liu, C. Apple, peach, and pear flower detection using semantic segmentation network and shape constraint level set. *Comput. Electron. Agric.* **2021**, *185*, 106150. [CrossRef]
11. Wu, D.; Lv, S.; Jiang, M.; Song, H. Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Comput. Electron. Agric.* **2020**, *178*, 105742. [CrossRef]
12. Dorj, U.O.; Lee, M.; Lee, K.K.; Jeong, G. A novel technique for tangerine yield prediction using flower detection algorithm. *Int. J. Pattern Recognit. Artif. Intell.* **2013**, *27*, 1354007. [CrossRef]
13. Lyu, S.; Zhao, Y.; Li, R.; Li, Z.; Fan, R.; Li, Q. Embedded Sensing System for Recognizing Citrus Flowers Using Cascaded Fusion YOLOv4-CF+ FPGA. *Sensors* **2022**, *22*, 1255. [CrossRef] [PubMed]
14. Zhong, Z.; Xiong, J.; Zheng, Z.; Liu, B.; Liao, S.; Huo, Z.; Yang, Z. A method for litchi picking points calculation in natural environment based on main fruit bearing branch detection. *Comput. Electron. Agric.* **2021**, *189*, 106398. [CrossRef]
15. Guo, Q.; Chen, Y.; Tang, Y.; Zhuang, J.; He, Y.; Hou, C.; Chu, X.; Zhong, Z.; Luo, S. Lychee fruit detection based on monocular machine vision in orchard environment. *Sensors* **2019**, *19*, 4091. [CrossRef]
16. Xiong, C.; Xiaoman, C.; Zou, X. A Method for Identification and Matching of the Picking Point for Mature Litchi under Structural Environment. *J. Appl. Biotechnol. Bioeng.* **2017**, *3*, 4. [CrossRef]
17. Wang, H.; Qian, Z.; Ma, S.; Zhou, Y.; Patrick, J.W.; Duan, X.; Jiang, Y.; Qu, H. Energy status of ripening and postharvest senescent fruit of litchi (*Litchi chinensis Sonn.*). *BMC Plant Biol.* **2013**, *13*, 55. [CrossRef]
18. Ibrahim, S.R.; Mohamed, G.A. Litchi chinensis: Medicinal uses, phytochemistry, and pharmacology. *J. Ethnopharmacol.* **2015**, *174*, 492–513. [CrossRef]
19. Wu, Y.H.S.; Chiu, C.H.; Yang, D.J.; Lin, Y.L.; Tseng, J.K.; Chen, Y.C. Inhibitory effects of litchi (*Litchi chinensis Sonn.*) flower-water extracts on lipase activity and diet-induced obesity. *J. Funct. Foods* **2013**, *5*, 923–929. [CrossRef]
20. Wei, Y.; Dong, C.; Zhang, H.; Zheng, X.; Shu, B.; Shi, S.; Li, W. Transcriptional changes in litchi (*Litchi chinensis Sonn.*) inflorescences treated with uniconazole. *PLoS ONE* **2017**, *12*, e0176053. [CrossRef]
21. Liu, Y.; Zhang, W.; Wang, J. Adaptive multi-teacher multi-level knowledge distillation. *Neurocomputing* **2020**, *415*, 106–113. [CrossRef]
22. Zhang, M.; Li, L.; Wang, H.; Liu, Y.; Qin, H.; Zhao, W. Optimized compression for implementing convolutional neural networks on FPGA. *Electronics* **2019**, *8*, 295. [CrossRef]
23. Li, S.; Sun, K.; Luo, Y.; Yadav, N.; Choi, K. Novel CNN-based AP2D-net accelerator: An area and power efficient solution for real-time applications on mobile FPGA. *Electronics* **2020**, *9*, 832. [CrossRef]
24. 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 Trans. Very Large Scale Integr. (VLSI) Syst.* **2019**, *27*, 1861–1873. [CrossRef]
25. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. *arXiv* **2015**, arXiv:1503.02531. [CrossRef]
26. Aguilar, G.; Ling, Y.; Zhang, Y.; Yao, B.; Fan, X.; Guo, C. Knowledge distillation from internal representations. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 7350–7357. [CrossRef]
27. Gotmare, A.; Keskar, N.S.; Xiong, C.; Socher, R. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. *arXiv* **2018**, arXiv:1810.13243. [CrossRef]
28. Srinivas, S.; Fleuret, F. Knowledge transfer with jacobian matching. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4723–4731.
29. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934. [CrossRef]

30. Jiang, Z.; Zhao, L.; Li, S.; Jia, Y. Real-time object detection method based on improved YOLOv4-tiny. *arXiv* **2020**, arXiv:2011.04244. [CrossRef]
31. 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 Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2778–2788. [CrossRef]
32. Yim, J.; Joo, D.; Bae, J.; Kim, J. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4133–4141. [CrossRef]
33. Zaras, A.; Passalis, N.; Tefas, A. Improving knowledge distillation using unified ensembles of specialized teachers. *Pattern Recognit. Lett.* **2021**, *146*, 215–221. [CrossRef]
34. Guo, C.; Lv, X.-L.; Zhang, Y.; Zhang, M.-L. Improved YOLOv4-tiny network for real-time electronic component detection. *Sci. Rep.* **2021**, *11*, 22744. [CrossRef]
35. Heo, B.; Lee, M.; Yun, S.; Choi, J.Y. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 3779–3787. [CrossRef]
36. Yue, K.; Deng, J.; Zhou, F. Matching guided distillation. In Proceedings of the European Conference on Computer Vision, Online, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 312–328. [CrossRef]
37. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 6 July–11 July 2015; pp. 448–456. [CrossRef]
38. Kim, J.; Park, S.; Kwak, N. Paraphrasing complex network: Network compression via factor transfer. *Adv. Neural Inf. Process. Syst.* **2018**, *31*. [CrossRef]
39. Romero, A.; Ballas, N.; Kahou, S.E.; Chassang, A.; Gatta, C.; Bengio, Y. Fitnets: Hints for thin deep nets. *arXiv* **2014**, arXiv:1412.6550. [CrossRef]
40. Wu, Y.; He, K. Group normalization. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19. [CrossRef]
41. La, T.M.; Matas, K.; Grunchevski, N.; Pham, K.D.; Koch, D. Fpgadefender: Malicious self-oscillator scanning for xilinx ultrascale+ fpgas. *ACM Trans. Reconfigurable Technol. Syst. (TRETS)* **2020**, *13*, 1–31. [CrossRef]
42. Abdi, A. Keras_to_tensorflow Tool. Version 1.0. Github Repository. November 2021. Available online: https://github.com/amir-abdi/keras_to_tensorflow (accessed on 23 March 2023).
43. Jocher, G.; Stoken, A.; Chaurasia, A.; Borovec, J.; Kwon, Y.; Michael, K.; Liu, C.; Fang, J.; Abhiram, V.; Skalski, S.; et al. ultralytics/yolov5: v6. 0—YOLOv5n 'Nano' Models, Roboflow Integration, TensorFlow Export, OpenCV DNN Support. Zenodo. 2021. Available online: https://zenodo.org/record/5563715 (accessed on 23 March 2023). [CrossRef]
44. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. Yolox: Exceeding yolo series in 2021. *arXiv* **2021**, arXiv:2107.08430. [CrossRef]
45. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696. [CrossRef]