

Article

An Edge-Oriented RT-DETR Integrated with Efficient Feature Extraction and Fusion Architecture and Lightweight Processing for Blueberry Maturity Detection

Lei Shi ^{1,2}, Zhuo Bai ¹, Yinyi Zhang ¹, Shuai Wang ¹, Qiyuan Fu ¹, Ziyue Li ¹, Yuhang Cui ², Yiman Dong ², Zhiyin Yang ^{1,2,*} and Yuxin Ye ^{3,*}

¹ Institute of Smart Agriculture, Jilin Agricultural University, Changchun 130118, China

² College of Information Technology, Jilin Agricultural University, Changchun 130118, China

³ College of Computer Science and Technology, Jilin University, Changchun 130012, China

* Correspondence: zyyang@jlau.edu.cn (Z.Y.); yeyx@jlu.edu.cn (Y.Y.)

Abstract

To address challenges such as severe occlusion caused by the dense growth of blueberry fruits in natural environments, complex backgrounds, and the limited computational resources of agricultural edge devices, this study proposes BR-DETR-Prune, a lightweight object detection model oriented towards edge computing environments. Based on the RT-DETR architecture, the model introduces a PConv-based FasterNet as the backbone network, which effectively reduces memory access latency and floating-point operation costs. Furthermore, it utilizes a “Gather-and-Distribute” (GD) mechanism to reconstruct the feature fusion neck. Through the unified aggregation and multi-branch distribution of global information, it significantly enhances the model’s feature extraction capability for dense and overlapping targets. An AIFI-RepBN encoder is designed, integrating re-parameterization technology into the attention module to further reduce computational redundancy. For lightweight processing, a random channel pruning strategy based on the “Lottery Ticket Hypothesis” is adopted to perform structural compression and fine-tuning on the model, achieving a significant reduction in the number of parameters while inversely improving accuracy. The experimental results demonstrate that BR-DETR-Prune achieves an mAP@0.5 of 97.1% on a self-built blueberry dataset, with only 15.52 M parameters and a computational load reduced to 34.0 GFLOPs. Its comprehensive performance is superior to mainstream models such as YOLOv8, YOLO11, and the original RT-DETR. Particularly, deployment testing on the NVIDIA Jetson Orin Nano Super embedded edge computing platform reveals that the model achieves a real-time inference speed of 20.5 FPS under FP16 precision, exhibiting smooth detection frames and strong robustness against occlusion. This study provides an effective optimization solution for the deployment of high-precision Transformer architectures on low-computational-power devices, offering an efficient and reliable visual perception approach for automated blueberry harvesting and yield estimation.

Keywords: blueberry maturity detection; RT-DETR; edge computing; model lightweighting; random channel pruning



Academic Editors: Luís Pádua and Anabela A. Fernandes-Silva

Received: 27 April 2026

Revised: 18 May 2026

Accepted: 22 May 2026

Published: 25 May 2026

Correction Statement: This article has been republished with a minor change. The change does not affect the scientific content of the article and further details are available within the backmatter of the website version of this article.

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

As a globally significant berry with high economic value, blueberries—particularly the Northern Highbush varieties widely introduced in Northern China—are hailed as the

“King of Berries” due to their abundance of anthocyanins, vitamins, and various antioxidant components. Against the backdrop of modern agriculture’s transition toward precision and intelligence, the large-scale cultivation of blueberries has imposed higher demands on field management. Among these, the timely and accurate determination of fruit maturity is not only a prerequisite for identifying the optimal harvest window and ensuring flavor and nutritional value, but also a critical link in optimizing post-harvest grading, storage logistics, and supply chain management. Blueberries at different maturity stages exhibit significant variations in firmness, sugar–acid ratio, and shelf-life: fully ripe fruits reach peak sugar content but are prone to softening and decay, necessitating immediate harvesting, whereas immature fruits require further development on the bush. Consequently, establishing an efficient and objective maturity monitoring system is of great significance for enhancing the economic benefits of the blueberry industry [1].

However, in actual orchard production environments, the precise monitoring of blueberry maturity faces several formidable challenges. Firstly, blueberry fruits are small in size and extremely densely distributed; a single bush often bears hundreds of berries growing in clusters, leading to severe mutual occlusion between fruits. The coverage of fruits by dense foliage is equally problematic, while complex lighting conditions (e.g., shadows, backlighting) further increase the difficulty of visual recognition. Traditional monitoring relies primarily on manual visual inspection, which is labor-intensive, time-consuming, and fails to meet the inspection needs of large-scale plantations; moreover, it is highly susceptible to subjective experience and fatigue, resulting in inconsistent standards and high misjudgment rates. Although the agronomy field has established the comprehensive BBCH (Biologische Bundesanstalt, Bundessortentamt und Chemische Industrie) scale system to describe various plant growth stages, this system relies primarily on manual visual inspection [2], making it unsuitable for direct application in non-contact, real-time monitoring tasks based on machine vision. Therefore, transforming agronomic maturity standards into computer-recognizable visual phenotypic features and achieving high-precision automated detection in complex, unstructured field environments has become an urgent challenge to be resolved in the field of smart agriculture.

With the global advancement of smart agriculture, deep learning techniques have been widely explored internationally for specialized fruit detection tasks, including berry yield estimation using smart drones [3] and quality trait assessment via lightweight network architectures [4]. Within this diversified research landscape, the You Only Look Once (YOLO) series of algorithms have become some of the most widely applied models in agricultural vision tasks due to their efficient one-stage detection architecture and excellent real-time performance [5–9]. In the field of blueberry detection, numerous studies have introduced targeted improvements centered on the YOLO framework. Several scholars have conducted research specifically on blueberry maturity detection. Sun et al. designed a lightweight deep learning framework, BMDNet-YOLO, which integrates improved feature extraction, attention-based fusion, and progressive transfer learning to enhance robustness and adaptability, achieving an mAP50 of 95.6% [10]. To detect blueberry maturity efficiently and accurately, Yuan et al. proposed CES-YOLO, a lightweight yet high-performance object detection model that achieved 91.22% mAP50, 69.18% mAP95, 89.21% precision, and 85.23% recall [11]. The GLL-YOLO method proposed by Xu et al., based on the YOLOv8 network, addresses fruit occlusion and complex backgrounds, improving the mAP for immature, semi-ripe, and ripe blueberries by 4.29%, 1.67%, and 1.39%, respectively, reaching 94.51%, 91.72%, and 93.32% [12]. This method offers the advantages of small model size and high precision, providing reliable support for intelligent blueberry harvesting. Wu et al. introduced an integrated framework combining the STF-YOLO model with the ByteTrack algorithm for maturity detection and counting, achieving 79.7% mAP50 and

stable counting in video streams [13]. Deng et al. developed “BlueberryCounter,” an iOS-based Swift application that allows growers to evaluate fruit maturity and count berries in real-time [14]. The user-friendly interface supports detection based on two YOLOv8-based fruit detectors—YOLOv8m (fast) and YOLOv8l (accurate)—allowing users to choose between speed and accuracy. Zhao et al. constructed a UAV remote sensing dataset for blueberry canopy fruits and proposed the PF-YOLO model; their experimental results showed mAP improvements ranging from 1.5% to 6.8% compared to YOLOv5, YOLOv8, and YOLOv9c, with an mAP50 of 54.4% [15]. Finally, Schumann et al. evaluated six versions of YOLO models for blueberry maturity detection, finding that the YOLOv4-Small network performed best with an average absolute error of 24.1%, successfully identifying three maturity stages of blueberries [16].

In general, most current research on blueberry maturity detection focuses on the YOLO series of algorithms based on Convolutional Neural Networks (CNNs), which primarily identify targets through local feature aggregation and lack the ability to capture global contextual information. This often leads to insufficient semantic understanding when the model distinguishes immature green fruits from green leaves with highly similar backgrounds or infers fruit integrity under severe occlusion. Moreover, many studies suffer from inadequate precision due to dataset or model limitations, and frequently experience accuracy loss during lightweighting processes. To address these technical bottlenecks and practical application requirements, this study proposes BR-DETR-Prune, a lightweight blueberry maturity detection model based on RT-DETR tailored for edge computing environments. The specific contributions are as follows:

- **An efficient feature extraction and fusion architecture for the edge is proposed:** To overcome the high latency of traditional CNN backbones, this study introduces FasterNet based on Partial Convolution (PConv) as the backbone, effectively reducing Memory Access Costs (MACs) and enhancing Floating Point Operations (FLOPs) efficiency. Simultaneously, to address the loss of features for small-target fruits in deep networks, a “Gather-and-Distribute” (GD) mechanism was innovatively employed to reconstruct the feature fusion neck. This significantly enhances the model’s perception of densely occluded targets through unified global information aggregation and multi-branch distribution. Additionally, an AIFI-RepBN module was designed, integrating re-parameterization techniques into the hybrid encoder to maintain the global receptive field advantages of Transformers while further reducing computational redundancy [17,18].
- **The superiority of the random channel pruning strategy in the Transformer architecture is verified:** This study explores a new path for model compression, confirming that the Random pruning strategy based on the “Lottery Ticket Hypothesis” outperforms traditional importance-based methods such as L1-Norm or Lamp in blueberry detection tasks. By generating random masks to physically eliminate redundant channels and combining this with a fine-tuning strategy, this method not only drastically compresses the model size but also introduces a structured regularization effect. This successfully resolves the overfitting issues caused by over-parameterization, achieving simultaneous model “slimming” and accuracy enhancement [19].
- **An SOTA-level balance between detection accuracy and inference speed is achieved:** Through the synergistic optimization of architecture reconstruction and deep pruning, BR-DETR-Prune demonstrated exceptional comprehensive performance on a high-quality self-built blueberry dataset. The experimental results show that while maintaining only 15.52 M parameters and 34.0 GFLOPs, the model achieved an mAP@0.5 of 97.1%. This result comprehensively surpasses mainstream detection models like YOLOv8, YOLO11, and the original RT-DETR, proving that a targeted lightweight

Transformer architecture can perfectly adapt to resource-constrained agricultural embedded devices, providing robust technical support for real-time precise monitoring in smart orchards [20,21].

2. Materials and Methods

2.1. Dataset Construction and Data Preprocessing

This study constructed a multi-source dataset combining field sampling and web mining. Benchmark samples were collected from Baishan City, Jilin Province (elevation 471 m), an area characterized by a North Temperate Continental Monsoon Climate. With an annual average temperature of 4.6 °C (summer maximum 36.5 °C, winter minimum −42.2 °C), a spring and autumn diurnal temperature difference exceeding 10 °C, an effective accumulated temperature of 2600–2700 °C, an annual average precipitation of 883.4 mm, 2259 h of sunshine, and a frost-free period of 110–125 days, this region is highly suitable for the growth of low-temperature plants such as blueberries. Focusing on locally introduced Northern Highbush blueberries, 847 high-resolution raw images were captured using the 48-megapixel multi-focal lenses of an iPhone 13. Images were taken at distances ranging from 15 to 30 cm from the fruit during typical lighting periods (9:00, 12:30, 15:00). To strengthen the model's generalization ability, an additional 1483 images of the same blueberry variety were acquired through web mining, ensuring that all exogenous data remained highly consistent with the field samples in terms of resolution, format, and fruit morphology. The overall data collection process was rigorous and objective, strictly adhering to data security and ethical compliance standards, thereby laying a high-quality foundation for subsequent model development.

Regarding the construction of the label system, the BBCH scale is a widely recognized visual phenological scale that categorizes plant developmental stages based on observable external morphological characteristics, rather than relying on internal physicochemical parameters. The blueberry detection system proposed in this study is strictly constructed following the scientific logic of the BBCH scale. To adapt to computer vision tasks, the complex developmental stages were deconstructed into three highly identifiable visual categories: the fully ripe stage, identified primarily by deep blue coverage and fruit fullness, corresponding to the optimal harvest period; the semi-ripe stage, characterized by uneven pink coloration reflecting the intermediate state of anthocyanin accumulation [22,23]; and the immature stage, marked by distinct green skin (see Figure 1). Based on these criteria, the dataset contains a total of 6296 blueberry fruit targets, with instances distributed as follows: 2292 fully ripe fruits, 2174 semi-ripe fruits, and 1830 immature fruits. This definition method based on visual phenotypes effectively resolves the issue of subjectivity in data annotation, ensuring the rigor and reproducibility of the training labels.

To ensure the accuracy and consistency of the dataset, all annotation tasks were executed by one author and strictly supervised by another. Subsequently, to meet the requirements of deep learning models for large-scale samples, systematic processing was performed on the initial data. To strictly prevent data leakage, dataset splitting (training, validation, and test sets at a ratio of 6:2:2) was conducted before applying any data augmentation algorithms, ensuring that no duplicate features from the same original image existed across different subsets. Following the split, geometric transformation algorithms, including spatial flipping and angular rotation, were applied only to the training set to simulate complex field environment variations (see Figure 2). During this process, the algorithm simultaneously corrected the annotation files associated with the image spatial coordinates to maintain the accuracy of the supervisory signals. Through these data augmentation techniques, the sample size was expanded to 6000 images. This processing aims to enhance the model's robustness under various complex environments rather than increasing the

actual diversity of biological samples, thereby providing sufficient prior information to improve the accuracy of the classification model.



Figure 1. Visual representation of blueberry maturity stages based on the proposed visual classification criteria.

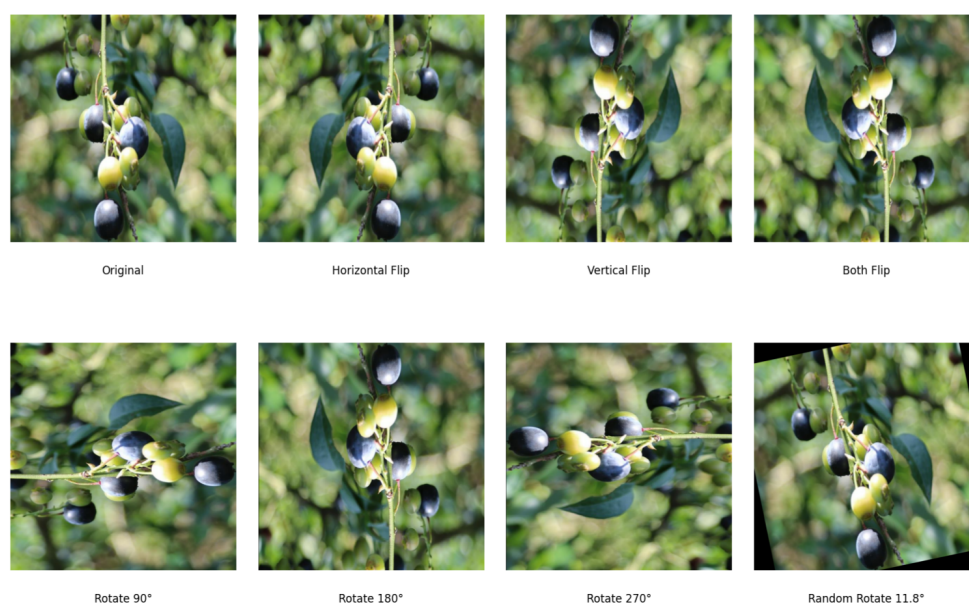


Figure 2. Demonstration of data augmentation methods, including spatial flipping and angular rotation.

2.2. RT-DETR and BR-DETR-Prune

The emergence of RT-DETR marks a significant paradigm shift in computer vision, enabling Transformer architectures to achieve industrial-grade real-time detection performance while maintaining high accuracy. It successfully bridges the “performance–speed” gap that has long hindered the widespread application of end-to-end models. Prior to RT-DETR, real-time object detection relied primarily on CNN architectures represented by YOLO. While their inference is rapid, they are often constrained by Non-Maximum Suppression (NMS), a non-end-to-end post-processing step that complicates the inference chain and makes global optimization difficult. Traditional DETR series removed NMS dependency via object query mechanisms but suffered from redundant computational loads when processing multi-scale high-resolution features, making them inadequate for low-latency scenarios such as autonomous driving and industrial inspection [24,25].

To address these challenges, RT-DETR implements a dual innovation in architectural design. First, the Efficient Hybrid Encoder of RT-DETR processes multi-scale features by decoupling intra-scale interaction from cross-scale fusion, a design that significantly reduces computational costs and facilitates real-time object detection. Through IoU-aware query selection, RT-DETR can focus on the most relevant targets within a scene, thereby enhancing detection accuracy. Furthermore, RT-DETR supports flexible adjustments of inference speed by employing different decoder layers without requiring retraining, which endows

the model with superior adaptability across various real-time object detection scenarios. Second, addressing the accuracy issues in query initialization, RT-DETR introduces IoU-aware Query Selection. By explicitly learning IoU scores at the encoder side, the model is guided to select more discriminative features as initial values for object queries, accelerating training convergence and significantly optimizing localization accuracy [26].

To meet the demand for real-time blueberry maturity detection on computationally constrained edge devices, this study proposes BR-DETR-Prune, a lightweight improved model based on the RT-DETR architecture. This model systematically reconstructs and prunes the backbone, feature fusion modules, and encoder to drastically reduce parameter count and latency while maintaining superior detection accuracy (the overall architecture is shown in Figure 3).

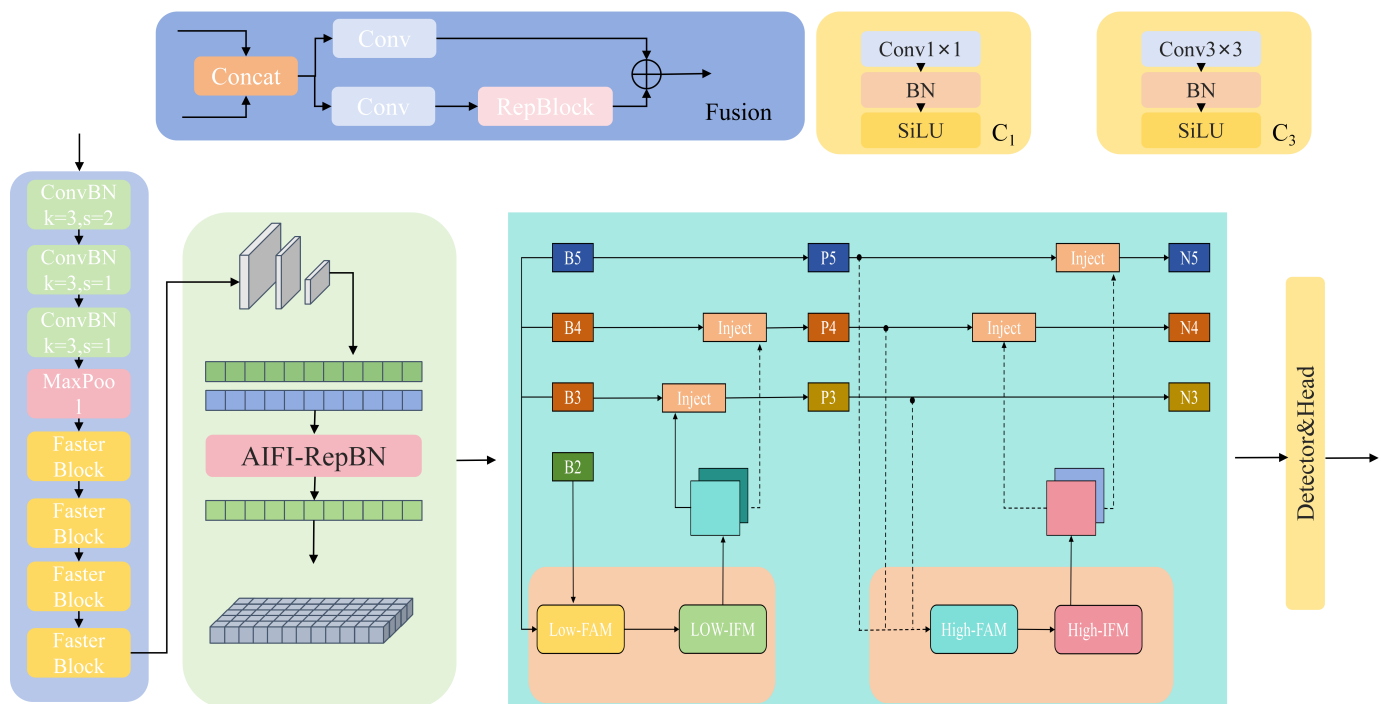


Figure 3. The overall network architecture of the proposed BR-DETR-Prune model, featuring the FasterNet backbone, the Gather-and-Distribute (GD) neck, and the AIFI-RepBN encoder.

Specifically, to address the memory access latency of traditional CNNs in hardware deployment, BR-DETR-Prune replaces the original backbone with FasterNet. FasterNet builds inverted residual blocks based on Partial Convolution (PConv) and Pointwise Convolution (PWConv), performing spatial downsampling and channel expansion through four hierarchical stages. This effectively reduces FLOPs while maintaining feature diversity via shortcut connections. Furthermore, to solve the information loss issue in the original CCFM module caused by indirect cross-layer transmission, this study introduces a “Gather-and-Distribute” (GD) mechanism to reconstruct the feature fusion neck. This mechanism utilizes High-GD and Low-GD branches, employing Feature Alignment Modules (FAM) and Information Fusion Modules (IFM) to collect multi-level features for global fusion, which are then distributed back to different levels via Injectors. This design effectively avoids semantic loss in multi-scale interactions and strengthens the positioning and recognition capabilities for blueberries of different sizes. At the encoder level, the AIFI-RepBN module is proposed, integrating Re-parameterized Batch Normalization (RepBN) into the original AIFI module. AIFI-RepBN eliminates redundant parameter overhead while preserving the advantage of attention mechanisms in capturing long-distance dependen-

cies [27]. Finally, based on the “Lottery Ticket Hypothesis,” a Random channel pruning strategy is adopted to compress the model. Unlike traditional methods relying on complex importance scores, Random pruning physically eliminates redundant channels and connections by generating random masks following a Bernoulli distribution, essentially performing a structured Dropout. Combined with subsequent fine-tuning [28], this strategy achieved an mAP@0.5 of 97.1%, outperforming YOLO series models of the same scale and proving the effectiveness and robustness of BR-DETR-Prune in real-time agricultural detection tasks.

2.3. FasterNet

Based on the novel Partial Convolution (PConv) and standard Pointwise Convolution (PWConv) as its primary building blocks, FasterNet has been proposed as a backbone that is both fast and efficient for various vision tasks [29]. The core objective is to maintain an architecture that is as simple as possible, avoiding redundant design elements to ensure the model is hardware-friendly. FasterNet consists of four hierarchical stages, each preceded by an embedding or merging layer for spatial downsampling and channel count expansion. Each stage contains multiple FasterNet blocks. Since the blocks in the last two stages involve lower memory access costs and typically higher FLOPS, more blocks are allocated to these stages to concentrate the majority of the computational effort.

Each FasterNet block comprises one PConv layer and two PWConv layers. These are integrated into an inverted residual block structure, where the intermediate layer has an expanded number of channels and shortcut connections are implemented to reuse input features. This design effectively reduces overall computational latency. Furthermore, Batch Normalization (BN) is utilized. A key advantage of BN is that it can be fused into adjacent convolutional layers during the inference phase, leading to faster execution while remaining as effective as other normalization layers. Regarding activation functions, GELU is selected for smaller FasterNet variants, while ReLU is used for larger ones, balancing runtime and efficiency. The final three layers of the network—Global Average Pooling, a 1×1 Convolutional layer, and a Fully Connected layer—are employed for feature transformation and classification. The overall architecture is shown in Figure 4.

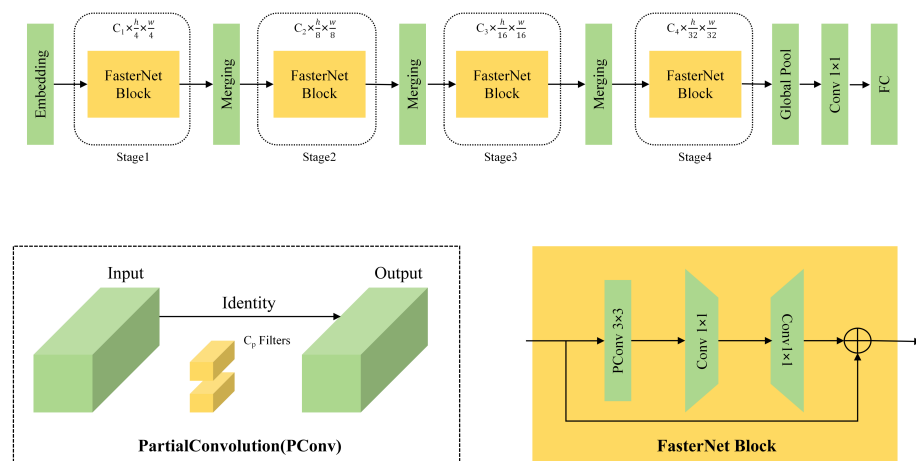


Figure 4. FasterNet diagram.

2.4. AIFI-RepBN

In RT-DETR, the AIFI (Attention-based Intra-scale Feature Interaction) module is integrated into the hybrid encoder to perform intra-scale feature interactions. It operates exclusively on high-level feature maps, utilizing the Multi-Head Self-Attention (MHSA) mechanism to capture rich global contextual information and establish long-range depen-

dencies. However, directly increasing the depth or width of the attention blocks to improve accuracy often introduces considerable memory access cost (MAC) and latency during edge device deployment. To enhance the feature representation capability without introducing additional inference burden, this study designs the AIFI-RepBN module by incorporating structure re-parameterization technology (RepBN). During the training phase, RepBN enriches feature representation through its multi-branch design; during the inference phase, these branches are equivalently fused into a single normalization layer. This design effectively improves the model's detection accuracy while maintaining hardware-friendly inference speeds [30]. The specific formulation of RepBN is defined in Equation (1).

$$\text{RepBN}(X) = \text{BN}(X) + \alpha X \quad (1)$$

In Equation (1), the variable α represents a learnable parameter. RepBN optimizes the model by adjusting the weights and offsets of the BN layer. When α is set to zero, the RepBN structure reverts to a standard BN configuration. Furthermore, RepBN can be parameterized to match the representation of BN, enabling efficient fusion with adjacent linear layers. This adaptability allows RepBN to enhance model performance while maintaining computational efficiency and reducing parameter redundancy [31] (see Figure 5).

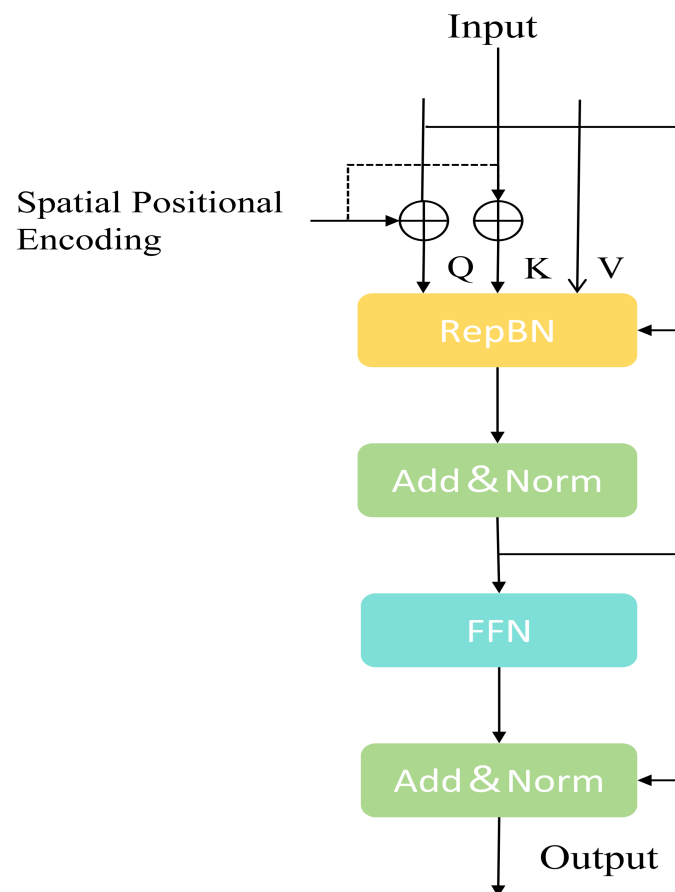


Figure 5. Schematic diagram of RepBN.

2.5. Gather-and-Distribute Mechanism(GD)

Multi-scale feature fusion is a widely used technique in object detection models aimed at enhancing the detection capability for objects of various sizes. By combining features from different layers of the network, this technique can capture information ranging from coarse to fine scales. Low-level features typically contain more details about small objects, while high-level features capture the semantic information of large objects. Multi-scale

feature fusion improves the model's ability to represent data by combining these multi-level features, enabling better detection and localization of objects of different sizes in an image.

Traditional object detection frameworks utilize the Feature Pyramid Network (FPN) structure as the neck, merging high- and low-level feature maps through upsampling. However, it can only fully merge features from adjacent layers; information from non-adjacent layers is transmitted indirectly, which may lead to data loss. The multi-scale information transmission in the CCFM module of the RT-DETR network suffers from the same issue. To address this problem, this study adopts the novel neck structure from Gold-YOLO [32]. This module introduces a Gather-and-Distribute (GD) mechanism, which collects and fuses information from all levels through a unified module and redistributes it to different levels, thereby avoiding information loss and enhancing feature fusion capabilities without significantly increasing latency.

The gather-and-distribute process involves three core modules: the Information Fusion Module (IFM), the Information Injector (Injector), and the Feature Alignment Module (FAM). During the gathering process, the FAM collects and aligns features from each layer, while the IFM merges these aligned features to create global information. The fused global information is then transmitted to each layer through the Injector module, while simple attention operations enhance the detection capability of the branches. To improve the model's ability to detect objects of different sizes, two branches were introduced: the High-Gather-and-Distribute (High-GD) and Low-Gather-and-Distribute (Low-GD) branches. These branches are responsible for extracting and combining feature maps of various sizes, enhancing the model's adaptability.

In the BR-DETR-Prune network, the CCFM module in RT-DETR is replaced by this "Gather-and-Distribute" structure. This enhancement optimizes the hierarchical feature representation of the model and broadens its receptive field. By integrating features of different scales, the model can capture a wider range of contextual information, improving the accuracy and reliability of object detection. This structure is particularly vital as it helps to preserve more semantic feature information across all levels, reducing the loss of semantic details [33]. The schematic diagram of the Gather-and-Distribute mechanism is shown in Figure 6.

2.6. Random Pruning Method

The introduction of lightweight models can effectively reduce computational resource requirements, enabling the detection system to maintain excellent real-time performance in environments with limited computing power. This study employs the Random channel pruning method to optimize the improved RT-DETR model, reducing computational complexity while preserving detection accuracy. In research on lightweight deep learning models, random channel pruning is an effective approach that optimizes the model by randomly removing a portion of the network channels. It aims to reduce redundant computation and enhance operational efficiency while retaining the core performance of the model as much as possible [34]. The process involves the following steps: based on a pre-trained full network, a certain proportion of channels are randomly selected for removal; these channels are then eliminated to form a simplified network structure; subsequently, fine-tuning training is conducted to recover the performance decline caused by channel removal; and the simplified model's predictive accuracy is evaluated.

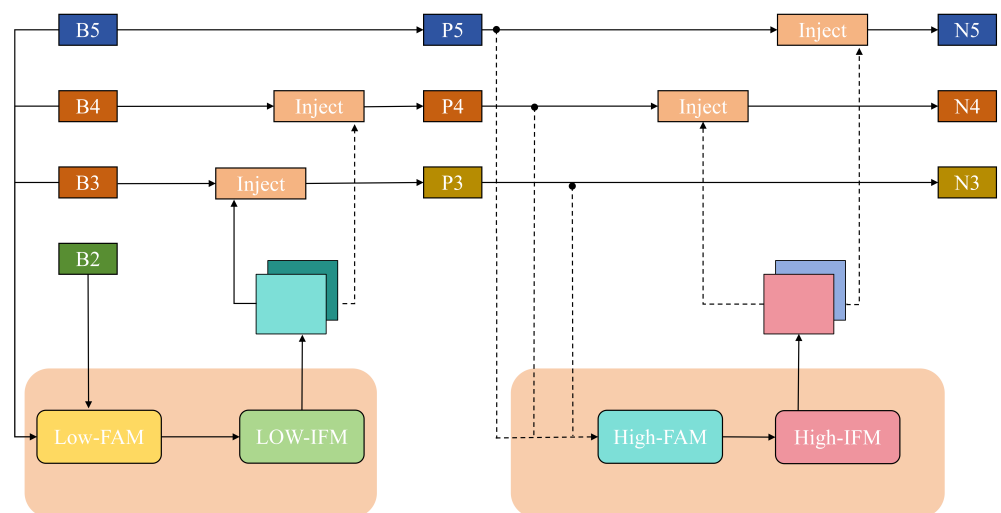


Figure 6. Schematic diagram of the Gather-and-Distribute (GD) mechanism principles.

To achieve real-time blueberry maturity detection on edge devices with limited computational resources, model lightweighting is an indispensable step. Traditional pruning methods, such as L1-Norm pruning or gradient-based sensitivity pruning, typically rely on complex “importance score” calculations, assuming that channels with smaller absolute weights or lower gradient contributions are less important. However, this assumption does not always hold true within the non-convex optimization surfaces of deep neural networks, and the process of calculating scores itself introduces additional computational overhead [35]. Based on the “Lottery Ticket Hypothesis” and the over-parameterization characteristics of neural networks, this study adopts a more concise and efficient Random pruning strategy.

The conceptual framework of Random pruning assumes that in deep Transformer architectures, network robustness and representation capacity depend significantly on the overall topological connectivity of the parameter space rather than the precise magnitude of individual channel weights [36]. For the attention-heavy RT-DETR model, which inherently exhibits feature redundancy, randomly removing a specified proportion of channels serves as a structural optimization mechanism. This approach physically scales down the computational volume and establishes a streamlined sub-network architecture, providing a foundational topology prior to the subsequent fine-tuning phase. Mathematically, the execution of this structural compression and configuration workflow is systematically divided into three distinct operational stages, as follows:

- **Random Mask Generation:** A pruning rate α is predefined. For each convolutional or linear layer in the network, a binary mask vector $M_l \in \{0, 1\}^{C_l}$ following a Bernoulli distribution is generated, where C_l represents the number of channels in that layer and $P(M_{li} = 0) = \alpha_0$.
- **Structural Elimination:** The mask M_l is applied to the corresponding weight matrix to physically eliminate channels with a mask value of 0 and their corresponding input/output connections, thereby reconstructing a “slimmed” network architecture. This process directly reduces the model’s FLOPs and parameter count, rather than merely zeroing out the weights.
- **Fine-tuning:** Since random pruning disrupts the original feature transmission paths, it leads to a temporary decline in model accuracy. Therefore, it is essential to fine-tune the pruned network on the training set. Using an SGD optimizer, the remaining parameters are allowed to re-adapt and seek the optimal solution under the new

topological structure, thereby recovering or even surpassing the performance of the original model.

Compared to other complex pruning algorithms, random pruning exhibits extremely high implementation efficiency and robustness, avoiding the risk of erroneously deleting key features due to “incorrect importance estimation.” The experimental results demonstrate that the fine-tuned random pruning model maintains exceptionally high detection accuracy while significantly reducing the computational load, proving the effectiveness of this strategy for blueberry maturity detection. The principle of random channel pruning is illustrated in Figure 7 [36].

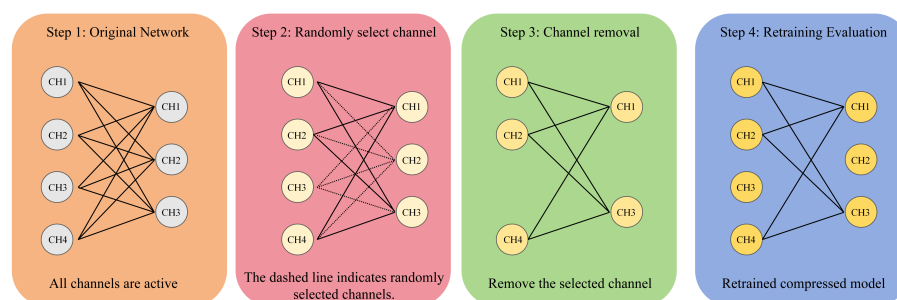


Figure 7. Schematic diagram of the random pruning process. The workflow illustrates the transition from a pre-trained network to a pruned architecture via random mask generation, followed by structural elimination and performance recovery through fine-tuning.

3. Results

3.1. Experimental Environment

The training configuration included an input image size of 640×640 pixels, a batch size of 16, and a total of 200 training epochs, with weight files saved every 10 epochs. Furthermore, the initial learning rate was set to 0.01 with a momentum of 0.937, and the SGD optimizer was employed to optimize the loss function. The experimental environment utilized in this study is detailed in Table 1, and the testing environment is presented in Table 2.

Table 1. Summary of Experimental Environment.

Component	Specifications
CPU	Intel(R) Xeon(R) Gold 5218 CPU @ 2.30 GHz
GPU	NVIDIA GeForce RTX 4060 Ti 16 GB $\times$ 2
Memory	64 GB (Training)
OS	Windows 10
Framework	PyTorch 2.1.2
Language	Python 3.8
CUDA Version	11.8

The software environment for all experiments was based on PyTorch 2.0.0 and Python 3.8. The training configurations included an input image size of 640×640 pixels, a batch size of 16, and 200 training epochs. Additionally, the initial learning rate was set to 0.001, the momentum was 0.937, and to ensure the strict reproducibility of the experiments, the random seed was uniformly set to 42. The NAdam optimizer was utilized to minimize the loss function. To ensure fair evaluation, all compared models were trained from scratch and evaluated using the exact same hyperparameter settings, data augmentation strategies, learning rate schedulers, and training durations.

Table 2. Summary of Testing Environment.

Component	Specifications
Main Configuration	NVIDIA Jetson Orin Nano Super Development Kit (NVIDIA, Santa Clara, CA, USA)
Memory	8 GB
OS	Ubuntu 22.04
Tensor Version	TensorRT 12
Framework	PyTorch 2.1.2
Language	Python 3.10
CUDA Version	11.8
Camera	CLB IMX219 (Sony Corporation, Tokyo, Japan)

3.2. Evaluation Metrics

Mean Average Precision (mAP) is employed to evaluate the average precision of the model across various categories. The mAP represents the mean of the Average Precision (AP) calculated for multiple classes. The formula is defined as follows:

$$AP = \int_0^1 P(R) dR \quad (2)$$

$$mAP = \frac{1}{|Q_R|} \sum_{q \in Q_R} AP(q) \quad (3)$$

where $|Q_R|$ denotes the number of target categories, q represents a specific detection category, and $AP(q)$ signifies the Average Precision value for category q . P and R represent Precision and Recall, respectively.

Parameters (Params): Params are utilized to evaluate the spatial complexity of the model.

Floating Point Operations (GFLOPS): GFLOPS is employed to assess the computational complexity of the model.

3.3. Pruning Experiments

To achieve efficient deployment on embedded devices with extremely limited computational resources, this study conducted rigorous lightweight experiments on the improved model architecture, focusing on the performance of four mainstream pruning strategies: Random, L1-Norm, DepGraph, and Lamp. The core objective is to maximize the compression of model Parameters (Params) and Floating Point Operations (GFLOPS) while simultaneously enhancing detection accuracy (mAP).

The experimental data indicate that while traditional pruning methods effectively reduce model complexity, they are often accompanied by a significant loss in accuracy. Specifically, although the magnitude-based L1-Norm pruning method compressed the parameters to 15.72 M, its mAP@0.5 was only 96.4%. This is primarily because the L1-Norm method relies on the assumption that “smaller absolute weights are less important”; however, in the highly non-convex optimization surfaces of deep neural networks, this assumption does not always hold, leading to the erroneous removal of certain low-weight channels that carry critical semantic information. Similarly, the Lamp method achieved an mAP@0.5 of 96.2% through a scoring mechanism, but its computational overhead (36.9 GFLOPs) remained relatively high, failing to achieve optimal lightweight results. While the DepGraph method performed well in handling structural dependencies, achieving 96.6% accuracy, there remains room for improvement in its compression ratio.

In contrast, the random pruning strategy adopted in this study demonstrates superior performance advantages. This method is based on the “Lottery Ticket Hypothesis” and the over-parameterization characteristics of neural networks, suggesting that network performance depends primarily on the connectivity of the topological structure rather than the numerical values of individual weights. By randomly removing channels and performing fine-tuning, random pruning effectively executes a structured Dropout operation, which not only significantly reduces the computational load but also introduces a regularization effect that prevents model overfitting. The experimental results show that, after random pruning and fine-tuning, the model parameters were reduced to a minimum of 15.52 M and the computational load to 34.0 GFLOPs, while the mAP@0.5 unexpectedly increased to 97.1%. Furthermore, the high-threshold accuracy mAP@0.5:0.95 reached a peak of 74.5%, proving that the random pruning strategy, while removing redundant features, encourages the remaining parameters to find an optimal solution space within the new topology through knowledge fine-tuning, thereby achieving a dual breakthrough in both accuracy and efficiency (see Table ?? and Figure 8).

Table 3. Performance Comparison of Different Pruning Methods

Method	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)	GFLOPs
L1-Norm	96.4	73.6	15.72	35.9
DepGraph	96.6	72.8	15.81	35.4
Lamp	96.2	74.1	16.01	36.9
Random (Ours)	97.1	74.5	15.52	34.0

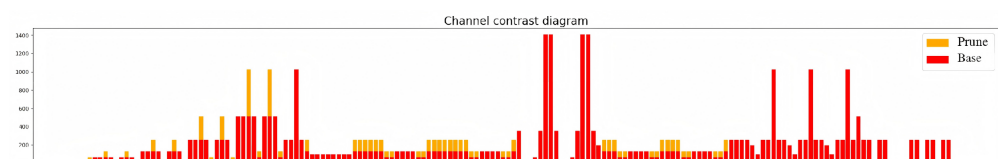


Figure 8. Visualization of Pruned Channels.

3.4. Ablation Studies

To systematically deconstruct the contribution and necessity of each innovative module within the BR-DETR-Prune model, this study designed multi-level ablation experiments. Using the original architecture without any improvements as a baseline, the FasterNet backbone, AIFI-RepBN encoder, GOLD-YOLO neck structure, and random pruning strategy were incrementally integrated, with performance changes at each stage meticulously recorded.

The initial mAP@0.5 of the baseline model was 94.4%, which, while possessing basic detection capabilities, proved insufficient for complex agricultural scenarios. After individually introducing the AIFI-RepBN module, the model accuracy increased to 95.1%. This indicates that by integrating Reparameterized Batch Normalization (RepBN) into the AIFI module, the model not only retains the ability of the attention mechanism to capture long-range dependencies but also enhances focus on key semantics by optimizing feature interaction logic. Furthermore, the independent introduction of the GOLD-YOLO neck, based on the “Gather-and-Distribute” mechanism, further elevated the accuracy to 95.4%. This significant gain validates the effectiveness of the mechanism in unifying the collection and distribution of global features through Low-GD and High-GD branches, successfully addressing the issue of semantic loss caused by the indirect transmission of cross-layer information in traditional FPN structures.

When FasterNet was further introduced as the backbone, although the accuracy for this single improvement was 94.9%, its combination with AIFI-RepBN saw the accuracy rise to

95.8%. This suggests that FasterNet, while reducing redundant computation through PConv, provides richer and more diverse feature map inputs for subsequent feature interactions. The most critical finding was that upon the simultaneous integration of FasterNet, AIFI-RepBN, and GOLD-YOLO, the model mAP@0.5 reached 96.6%. Implementing random pruning on this basis did not lead to a decrease in accuracy; instead, the mAP@0.5 ultimately settled at 97.1%, while significantly compressing the computational load (GFLOPs) to 34.0 (as shown in Table 4).

Table 4. Ablation study of the proposed modules on the blueberry dataset.

No.	FasterNet	AIFI-RepBN	GD	Pruning	mAP@0.5	mAP@0.5:0.95	Params (M)	GFLOPs
1	×	×	×	×	94.4	71.1	19.88	57.0
2	×	✓	×	×	95.1	72.6	20.09	58.3
3	×	×	✓	×	95.4	72.4	22.50	61.4
4	✓	×	×	×	94.9	73.1	16.79	49.5
5	✓	✓	×	×	95.8	73.5	17.00	50.9
6	✓	✓	✓	×	96.6	73.8	19.42	54.1
7	✓	✓	✓	✓	97.1	74.5	15.52	34.0

3.5. Comparative Experiments

To establish the state-of-the-art (SOTA) status of BR-DETR-Prune in the field of object detection, this study conducted a comprehensive horizontal comparison with currently representative SOTA models. The comparison involves various scale variants of RT-DETR (RTDETR-l, R50, R34) and multiple iterations of the YOLO series (YOLOv8m, YOLO11m, YOLO12m, YOLO13l).

In comparison with the homologous RT-DETR series, BR-DETR-Prune demonstrates a clear advantage in both accuracy and efficiency. While the original RTDETR-l achieves an accuracy of 96.5%, it suffers from a high parameter count of 31.99 M and a computational load of 103.4 GFLOPs. In contrast, BR-DETR-Prune achieves a higher accuracy of 97.1% while reducing computational load by approximately 67% and parameters by about 51%. This fully illustrates that the reconstruction of the encoder and feature fusion modules successfully addresses the bottleneck of excessive computational resource consumption. Even compared with RTDETR-R50 (94.8% mAP), the proposed model achieves comprehensive superiority in both accuracy and speed.

BR-DETR-Prune similarly exhibits outstanding performance compared to the YOLO series. For instance, YOLOv8m has an mAP@0.5 of 92.64% with 78.7 GFLOPs; the proposed model exceeds its accuracy by nearly 4.5 percentage points while reducing computational costs by more than half. In summary, with its unique design philosophy, BR-DETR-Prune surpasses heavy-duty models in accuracy and rivals lightweight models in speed, proving its practical value as an optimal solution for blueberry ripeness detection. The detailed comparative results and visualizations are presented in Table 5 and Figures 9–12.

Table 5. Performance Comparison of Different Models.

Model	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Params (M)	GFLOPs	FPS
BR-DETR-Prune	97.1	74.5	15.52	34.0	69.5
RTDETR-l	96.5	73.8	31.99	103.4	40.5
RTDETR-R50	94.8	73.3	41.96	129.6	36.7
RTDETR-R34	94.8	72.9	31.11	88.8	46.2
YOLO13l	90.2	66.7	27.6	64.0	63.8
YOLO12m	93.7	70.7	20.10	67.1	67.4
YOLO11m	93.5	69.8	20.03	67.7	67.2
YOLOv8m	92.64	66.98	25.84	78.7	66.4

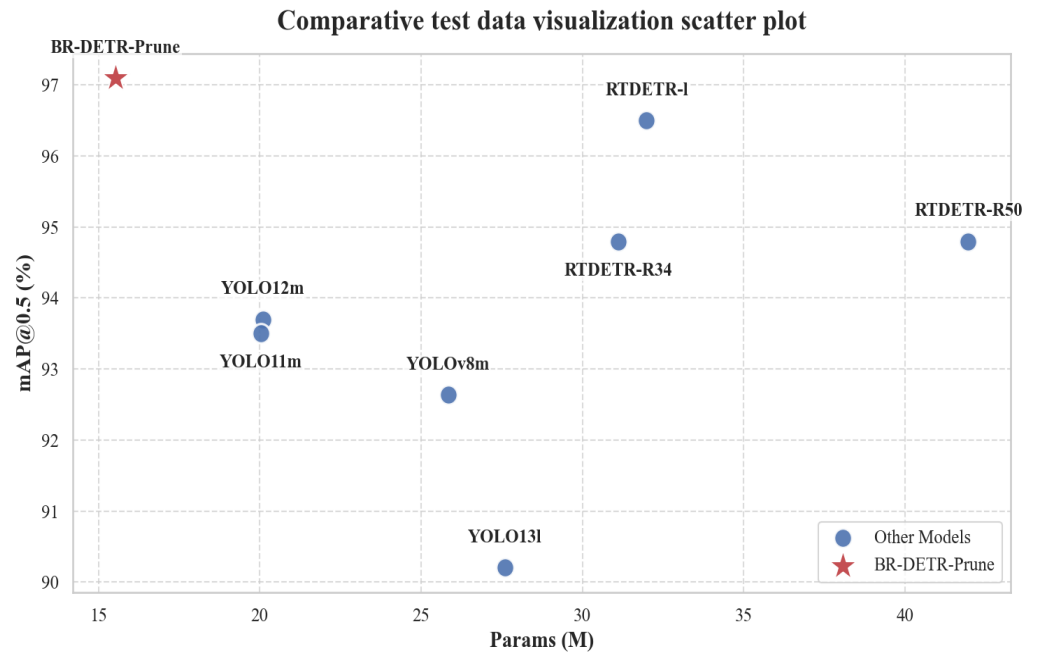


Figure 9. Scatter plot of accuracy vs. number of parameters.

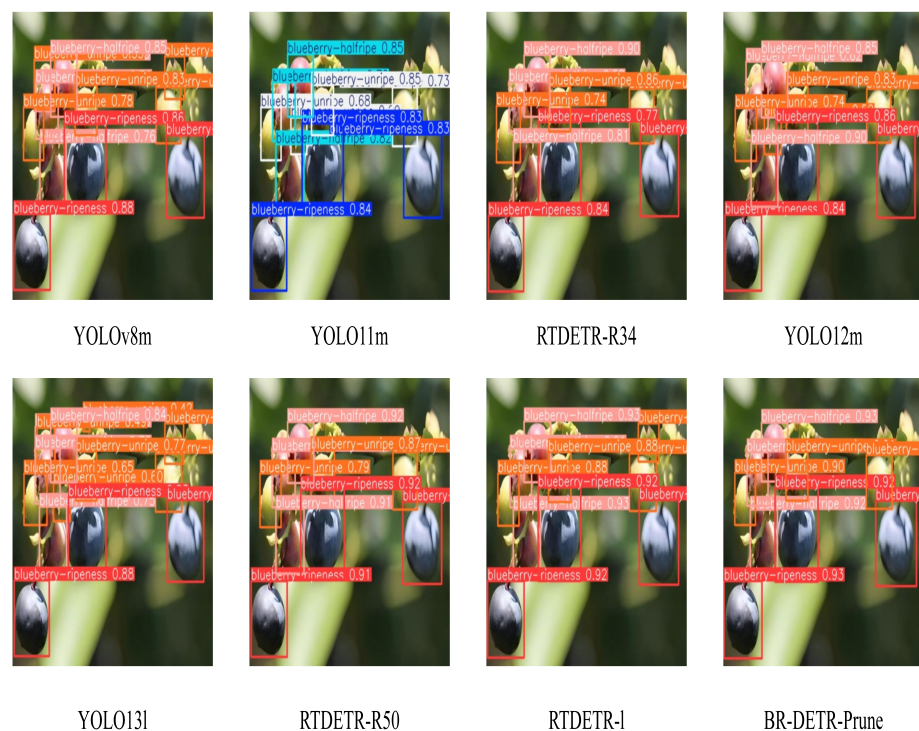


Figure 10. Detection results of different models.

3.6. Deployment Experiments

To verify the applicability and real-time detection capability of the BR-DETR-Prune model in real-world agricultural scenarios, a hardware testing system based on an embedded edge computing platform was constructed. The experimental hardware employs the NVIDIA Jetson Orin Nano as the edge inference core, which is highly suitable for integration into agricultural inspection robots or picking manipulators.

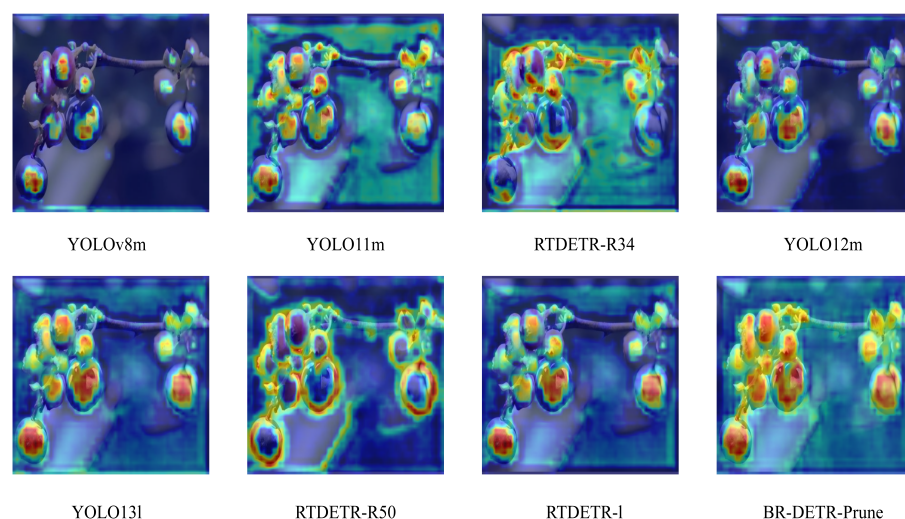


Figure 11. Heatmap visualization of different models.

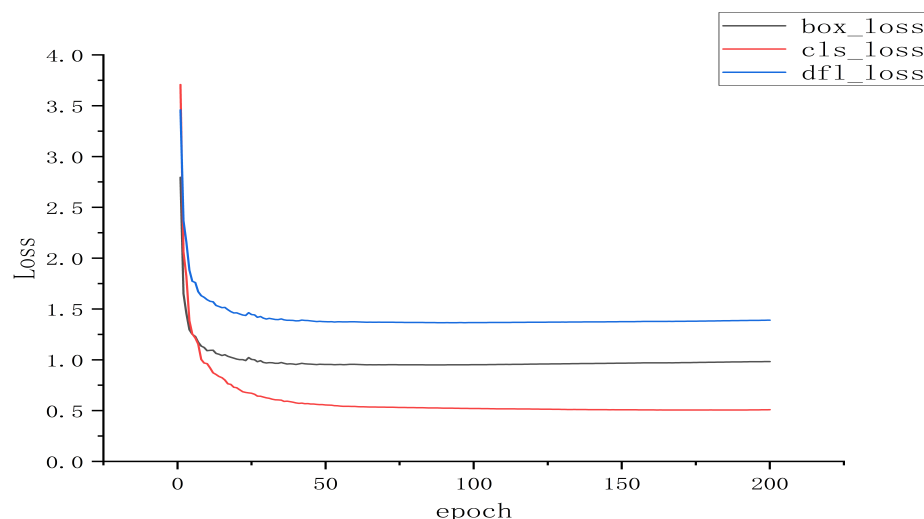


Figure 12. Training loss curves.

During the deployment phase, the TensorRT inference engine was utilized to accelerate the trained PyTorch model, quantizing the model weights from FP32 to FP16 precision. The experimental results are illustrated in Figure 13. In the video stream real-time detection task, the lightweight model achieved an inference speed of 20.5 FPS at the edge. This frame rate meets the real-time visual feedback requirements of most agricultural automation equipment. Furthermore, the detection boxes remain tight and accurate under dense occlusion and varying lighting conditions. This proves that the BR-DETR-Prune model successfully overcomes the bottleneck that prevents Transformer architectures from being deployed on low-power edge devices, demonstrating significant engineering application value.

3.7. Discussion

The BR-DETR-Prune model proposed in this study demonstrates competitive comprehensive performance in blueberry maturity detection, offering a feasible solution for handling dense fruit occlusion and multi-scale variations. The performance of this model is primarily attributed to the collaborative optimization at the architectural level and the targeted model compression strategy. To address the dense occlusion caused by the clustered growth of blueberries, traditional CNN architectures, which rely heavily on localized convolution kernels, are prone to missed detections or misclassifications in over-

lapping regions. In contrast, the RT-DETR architecture introduced in this study utilizes the Multi-Head Self-Attention mechanism to capture global contextual information and incorporates a “Gather-and-Distribute” (GD) mechanism to fuse multi-scale features, which helps to enhance the model’s representation capabilities for overlapping and micro targets. Meanwhile, the collaborative design of the FasterNet backbone and the AIFI-RepBN encoder enhances feature representation via structure re-parameterization while maintaining hardware-friendly inference speeds. Furthermore, the experimental results indicate that the Random channel pruning strategy can, to some extent, exert a regularization effect through random structural elimination, reducing parameters while preserving model robustness.



Figure 13. Field test on Jetson Orin Nano Super.

In terms of detection accuracy, BR-DETR-Prune exhibits certain technical advantages compared to several existing modified YOLO-based studies. For instance, the BMDNet-YOLO proposed by Sun et al. enhanced robustness via progressive transfer learning, achieving an mAP50 of 95.6% [10]; the CES-YOLO model presented by Yuan et al. reported an mAP50 of 91.22% [11]; and the GLL-YOLO model designed by Xu et al. for fruit blocking issues yielded a comprehensive mAP of approximately 94.51% [12]. In comparison, the model in this study achieved an mAP@0.5 of 97.1% on the current dataset. This suggests that, under dense clustered growth conditions, the global self-attention mechanism can provide effective global context to decouple overlapping targets, thereby alleviating the limitations that traditional convolutional neural networks might encounter due to the lack of long-range dependency capture, to some extent. Nevertheless, since the dataset scale and experimental conditions in this study are relatively specific, the aforementioned comparative conclusions still require validation across a broader range of practical environments.

Regarding real-time performance and edge deployment capabilities, the results of this study indicate the model’s application potential on specific hardware platforms. For example, although Wu et al. implemented video stream counting by combining STF-YOLO with ByteTrack, their detection accuracy was limited to 79.7% mAP50 [13]. In this study, the model parameters were compressed to 15.52 M via Random pruning, achieving an inference speed of 20.5 FPS during testing on the NVIDIA Jetson Orin Nano edge computing platform. This performance provides a valuable balancing approach to mitigating the common

contradiction in agricultural automation where “high-precision models are difficult to deploy, and lightweight models lack sufficient accuracy,” suggesting the feasibility of applying reasonably optimized and compressed Transformer architectures to resource-constrained agricultural equipment. However, it should be noted that because the current evaluation was conducted on a specific edge hardware platform, its generalized inference speed across diverse computational architectures warrants further investigation.

Although BR-DETR-Prune demonstrated favorable results within the current experimental scope, several limitations remain in this study. When dealing with extremely complex dynamic illumination (such as severe backlighting), insufficient data for night-time supplementary lighting scenarios, or severe leaf-like interferences (as shown in Figure 14), the generalization ability of the model still exhibits deficiencies, necessitating more extensive environmental datasets for collaborative optimization. Furthermore, the current detection network relies primarily on the external visual phenotypic features of visible light images, which remains limited in terms of non-destructive detection of internal fruit quality (such as sugar content and acidity).



Figure 14. Visual examples of severe leaf-like interferences and fruit occlusion under natural orchard environments.

Therefore, future work will focus on introducing multispectral or hyperspectral imaging techniques to integrate internal physicochemical information, alongside conducting multi-batch experiments under more diverse climatic and geographical conditions. Meanwhile, collaborative optimization considering the motion planning and target localization errors of specific harvesting robotic arms will be pursued to construct a more robust and comprehensive agricultural perception system.

Finally, it is worth noting that due to computational resource constraints, the models in this study were evaluated based on single-run training with a fixed random seed (Seed = 42). While the substantial performance improvements demonstrate the effectiveness of the proposed method, future work will involve multiple independent runs with varied seeds to conduct a more comprehensive statistical analysis of the model’s robustness.

4. Conclusions

Addressing challenges such as dense fruit occlusion, complex backgrounds, and the limited computational power of edge devices in natural environments, this study proposes BR-DETR-Prune, a lightweight detection model tailored for edge computing. Based on the RT-DETR framework, the model enhances feature extraction efficiency and multi-scale perception by incorporating the FasterNet lightweight backbone, a reconstructed AIFI-RepBN encoder, and a “Gather-and-Distribute” (GD) feature fusion mechanism. Furthermore, the innovative application of the Random channel pruning strategy successfully achieved significant model compression while simultaneously improving overall performance.

The experimental results demonstrate that BR-DETR-Prune achieved an mAP@0.5 of 97.1% on a custom blueberry dataset, outperforming mainstream models such as YOLOv8, YOLO11, and the original RT-DETR in terms of accuracy. With only 15.52 M param-

ters and a computational load of 34.0 GFLOPs, the model reached a real-time inference speed of 20.5 FPS on the NVIDIA Jetson edge computing platform. This research not only provides an efficient and reliable visual perception solution for intelligent blueberry harvesting and yield estimation, but also offers a new theoretical basis and technical reference for the lightweight deployment of Transformer architectures on resource-constrained agricultural equipment.

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

Funding: This research was funded by the Jilin Province Science and Technology Development Program Project, grant number 20250201059GX.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding authors.

Acknowledgments: We would like to express our sincere gratitude to the handling editors and the anonymous reviewers for their invaluable comments, rigorous evaluation, and constructive suggestions. We are particularly grateful to the editorial team for their professional guidance and efficient coordination throughout the peer-review process, which significantly contributed to enhancing the theoretical depth and overall presentation of this manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang, Y.; Li, X.; Ge, Y.; Hu, Y.; Zhu, R.; Yang, X.; Chen, S.; Peng, H.; Wang, C. Blueberry Leaf Polysaccharide/Gelatin Composite Gel: Preparation, Characterization, and Formation Mechanism. *Int. J. Biol. Macromol.* **2025**, *304*, 141020. [\[CrossRef\]](#)
2. Gai, R.; Liu, Y.; Xu, G. TL-YOLOv8: A Blueberry Fruit Detection Algorithm Based on Improved YOLOv8 and Transfer Learning. *IEEE Access* **2024**, *12*, 86378–86390. [\[CrossRef\]](#)
3. Nguyen, H.D.; McHenry, B.; Nguyen, T.; Zappone, H.; Thompson, A.; Tran, C.; Segrest, A.; Tonon, L. Accurate Crop Yield Estimation of Blueberries using Deep Learning and Smart Drones. *J. Big Data Artif. Intell.* **2025**, *3*, 5–24. [\[CrossRef\]](#)
4. Arellano, C.; Sagredo, K.; Muñoz-Schick, C.E.; Govan, J.E. Deep learning models to detect wax bloom on blueberry fruits from images. *Chil. J. Agric. Res.* **2025**, *85*, 601–610. [\[CrossRef\]](#)
5. Cao, J.; Bao, W.; Shang, H.; Yuan, M.; Cheng, Q. GCL-YOLO: A GhostConv-Based Lightweight YOLO Network for UAV Small Object Detection. *Remote Sens.* **2023**, *15*, 4932. [\[CrossRef\]](#)
6. Khan, Z.; Liu, H.; Shen, Y.; Zeng, X. Deep Learning Improved YOLOv8 Algorithm: Real-Time Precise Instance Segmentation of Crown Region Orchard Canopies in Natural Environment. *Comput. Electron. Agric.* **2024**, *224*, 109168. [\[CrossRef\]](#)
7. Khan, A.T.; Jensen, S.M.; Khan, A.R. Advancing Precision Agriculture: A Comparative Analysis of YOLOv8 for Multi-Class Weed Detection in Cotton Cultivation. *Artif. Intell. Agric.* **2025**, *15*, 182–191. [\[CrossRef\]](#)
8. You, H.; Li, Z.; Wei, Z.; Zhang, L.; Bi, X.; Bi, C.; Li, X.; Duan, Y. A Blueberry Maturity Detection Method Integrating Attention-Driven Multi-Scale Feature Interaction and Dynamic Upsampling. *Horticulturae* **2025**, *11*, 600. [\[CrossRef\]](#)
9. Shi, L.; Wei, Z.; You, H.; Wang, J.; Bai, Z.; Yu, H.; Ji, R.; Bi, C. OMC-YOLO: A Lightweight Grading Detection Method for Oyster Mushrooms. *Horticulturae* **2024**, *10*, 742. [\[CrossRef\]](#)
10. Sun, H.; Wang, R.-F. BMDNet-YOLO: A Lightweight and Robust Model for High-Precision Real-Time Recognition of Blueberry Maturity. *Horticulturae* **2025**, *11*, 1202. [\[CrossRef\]](#)
11. Yuan, J.; Fan, J.; Sun, Z.; Liu, H.; Yan, W.; Li, D.; Liu, H.; Wang, J.; Huang, D. Deployment of CES-YOLO: An Optimized YOLO-Based Model for Blueberry Ripeness Detection on Edge Devices. *Agronomy* **2025**, *15*, 1948. [\[CrossRef\]](#)
12. Xu, Y.; Li, H.; Zhou, Y.; Zhai, Y.; Yang, Y.; Fu, D. GLL-YOLO: A Lightweight Network for Detecting the Maturity of Blueberry Fruits. *Agriculture* **2025**, *15*, 1877. [\[CrossRef\]](#)
13. Wu, N.; Wu, J.; Wang, Z.; Zhao, Y.; Xu, X.; Wang, Y.; Skobelev, P.; Mi, Y. Maturity Detection and Counting of Blueberries in Real Orchards Using a Novel STF-YOLO Model Integrated with ByteTrack Algorithm. *Front. Plant Sci.* **2025**, *16*, 1682024. [\[CrossRef\]](#)

14. Deng, B.; Lu, Y.; Li, Z. Detection, Counting, and Maturity Assessment of Blueberries in Canopy Images Using YOLOv8 and YOLOv9. *Smart Agric. Technol.* **2024**, *9*, 100620. [\[CrossRef\]](#)
15. Zhao, Y.; Li, Y.; Xu, X. Object Detection in High-Resolution UAV Aerial Remote Sensing Images of Blueberry Canopy Fruits. *Agriculture* **2024**, *14*, 1842. [\[CrossRef\]](#)
16. MacEachern, C.B.; Esau, T.J.; Schumann, A.W.; Hennessy, P.J.; Zaman, Q.U. Detection of Fruit Maturity Stage and Yield Estimation in Wild Blueberry Using Deep Learning Convolutional Neural Networks. *Smart Agric. Technol.* **2023**, *3*, 100099. [\[CrossRef\]](#)
17. Chen, J.; Kao, S.; He, H.; Zhuo, W.; Wen, S.; Lee, C.-H.; Chan, S.-H.G. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 12021–12031.
18. Chen, K.; Du, B.; Wang, Y.; Wang, G.; He, J. The Real-Time Detection Method for Coal Gangue Based on YOLOv8s-GSC. *J. Real-Time Image Proc.* **2024**, *21*, 37. [\[CrossRef\]](#)
19. Li, Y.; Adamczewski, K.; Li, W.; Gu, S.; Timofte, R.; Gool, L.V. Revisiting Random Channel Pruning for Neural Network Compression. *arXiv* **2022**, arXiv:2205.05676. [\[CrossRef\]](#)
20. Jiang, F.; Lu, K.; Wang, W.; Pan, X.; Zhang, J.; Chen, P.; Wang, B. FPDNet: A Fast and High-Precision Detection Network for Hot-Rolled Strip Surface Defects. *Measurement* **2025**, *253*, 117309. [\[CrossRef\]](#)
21. Liu, L.; Li, P.; Wang, D.; Zhu, S. A Wind Turbine Damage Detection Algorithm Designed Based on YOLOv8. *Appl. Soft Comput.* **2024**, *154*, 111364. [\[CrossRef\]](#)
22. Joine, M.; Sakar, E.H. Floral and Fruit Phenology in Three Young Olive (*Olea europaea* L.) Cultivars Grown in Central Northern Morocco. Part I: Reproductive Phenological Calendar and Agroclimatic Requirements According to the Olive Specific BBCH Scale. *Appl. Fruit Sci.* **2025**, *67*, 116. [\[CrossRef\]](#)
23. Santos, R.F.; Chagas, P.C.; Moura, E.A.; Moraes Lima, M.E.; Rocha Araújo, M.C.; Chagas, E.A. Phenological Growth Stages Based on the BBCH Scale and Thermal Requirements of Spondias Dulcis Parkinson. *Appl. Fruit Sci.* **2025**, *67*, 207. [\[CrossRef\]](#)
24. Li, X.; Cai, M.; Tan, X.; Yin, C.; Chen, W.; Liu, Z.; Wen, J.; Han, Y. An Efficient Transformer Network for Detecting Multi-Scale Chicken in Complex Free-Range Farming Environments via Improved RT-DETR. *Comput. Electron. Agric.* **2024**, *224*, 109160. [\[CrossRef\]](#)
25. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs Beat YOLOs on Real-Time Object Detection. In Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024; pp. 16965–16974.
26. Saltık, A.O.; Allmendinger, A.; Stein, A. Comparative Analysis of YOLOv9, YOLOv10 and RT-DETR for Real-Time Weed Detection. *arXiv* **2024**, arXiv:2412.13490.
27. Shi, R.; Zhang, L.; Wang, G.; Jia, S.; Zhang, N.; Wang, C. GD-Det: Low-Data Object Detection in Foggy Scenarios for Unmanned Aerial Vehicle Imagery Using Re-Parameterization and Cross-Scale Gather-and-Distribute Mechanisms. *Remote Sens.* **2025**, *17*, 783. [\[CrossRef\]](#)
28. Shi, Q.; Hu, M.; Suganthan, P.N.; Katuwal, R. Weighting and Pruning Based Ensemble Deep Random Vector Functional Link Network for Tabular Data Classification. *Pattern Recognit.* **2022**, *132*, 108879. [\[CrossRef\]](#)
29. Guo, A.; Sun, K.; Zhang, Z. A Lightweight YOLOv8 Integrating FasterNet for Real-Time Underwater Object Detection. *J. Real-Time Image Proc.* **2024**, *21*, 49. [\[CrossRef\]](#)
30. Chen, X.; Wang, G. FP-RTDETR: Enhancing Infrared Ship Detection with Multi-Scale Feature Fusion and Lightweight Design. *J. Supercomput.* **2025**, *81*, 984. [\[CrossRef\]](#)
31. Kong, X.; Chen, X. CLEAR-DETR: Cross-Weather Low-Visibility Enhanced Atmospheric Recognition Transformer for Robust Traffic Sign Detection. *Signal Image Video Process.* **2026**, *20*, 73. [\[CrossRef\]](#)
32. Xu, C.; Liao, Y.; Liu, Y.; Tian, R.; Guo, T. Lightweight Rail Surface Defect Detection Algorithm Based on an Improved YOLOv8. *Measurement* **2025**, *242*, 115922. [\[CrossRef\]](#)
33. Dai, S.; Bai, T.; Zhao, Y. Keypoint Detection and 3D Localization Method for Ridge-Cultivated Strawberry Harvesting Robots. *Agriculture* **2025**, *15*, 372. [\[CrossRef\]](#)
34. You, H.; Wang, H.; Wei, Z.; Bi, C.; Zhang, L.; Li, X.; Yin, Y. VBP-YOLO-Prune: Robust Apple Detection under Variable Weather via Feature-Adaptive Fusion and Efficient YOLO Pruning. *Alex. Eng. J.* **2025**, *128*, 992–1014. [\[CrossRef\]](#)
35. Yang, X.; Wang, Z.; Dong, M. PRE-YOLO: A Lightweight Model for Detecting Helmet-Wearing of Electric Vehicle Riders on Complex Traffic Roads. *Appl. Sci.* **2024**, *14*, 7703. [\[CrossRef\]](#)
36. Song, B.; Chen, J.; Liu, W.; Fang, J.; Xue, Y.; Liu, X. YOLO-ELWNet: A Lightweight Object Detection Network. *Neurocomputing* **2025**, *636*, 129904. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.