International Journal of Applied Mathematics in Control Engineering 7 (2024) 137-143

Contents lists available at YXpublications

International Journal of Applied Mathematics in Control Engineering

Journal homepage: http://www.ijamce.com

CS-YOLOv8: Lightweight Mango Detection Algorithm Based on YOLOv8

Xu Fang, Xingguo Song*, Yongjiang Li

Mechanical Engineering School, Southwest Jiaotong University, Chengdu 610031, China

ARTICLE INFO

Article history:
Received 21 March 2024
Accepted 11 June 2024
Available online 15 June 2024

Keywords:
Mango detection
YOLOv8
Lightweight
Partial Convolution

ABSTRACT

To address the issues of high memory usage in mango detection tasks, we propose a lightweight mango detection algorithm based on YOLOv8, called CS-YOLOv8. Firstly, in order to mitigate redundant calculations and memory access, we substitute the backbone of YOLOv8 with FasterNet. Secondly, we propose Channel Shuffle-Partial Convolution (CS-PConv) module, which fuses the channel shuffle mechanism and Partial Convolution (PConv) to enhance the exchange of information between convolutional and non-convolutional channels. Subsequently, CS-PConv is used to replace the 1x1 convolution within the FasterNetBlock to design Channel Shuffle-FasterNetBlock (CS-FasterNetBlock), solving the problem of limited receptive field. Finally, we devise the Shared Parameter Head (SP-Head) by amalgamating the concept of shared weights with the CS-FasterNetBlock, not only diminishing the initial network parameters but also amplifying the important features of mango. We validate the effectiveness of our algorithm on both the MangoYOLO dataset. The experimental results demonstrate that the algorithm proposed in this paper significantly reduces the number of model parameters by 63.3% and decreases GFLOPs by 6.0G, while achieving an improvement of 0.6% in mAP0.5:0.95.

1. Introduction

Mangoes are a commercially significant fruit crop with considerable economic importance, thriving in tropical and subtropical regions globally. Mangoes contain a variety of bioactive compounds including vitamin C, β-carotene, and polyphenols, they have extremely high nutritional value (Tharanathan et al.,2006; Singh et al., 2013; Ntsoane et al., 2019). As labor availability in orchards diminishes, manual picking has become increasingly challenging for growers (Fennimore et al., 2008; Zhou H et al.,2022; Mail M F et al.,2023). Consequently, the adoption of intelligent mango-picking robots represents a promising approach to sustain productivity and maintain product quality. However, the algorithms for mango detection still face the following issues.

Model Complexity and Computational Resource Constraints: Mango picking algorithms typically demand substantial computational power and parameters, resulting in costly hardware deployment. Despite these demands, the actual benefits derived from costly hardware in outdoor mango picking environments do not consistently align with the investment. The significant need for computational resources impedes the widespread adoption of such technologies. Consequently, it is imperative to explore lighter and more efficient algorithms to reduce costs and enhance deplorability in real world settings.

Challenges in Mango Recognition within Agricultural Contexts: The task of mango recognition in agriculture is plagued by the complexity of environmental conditions. For instance, when capturing images of mangoes in natural state presents numerous challenges, including unstable lighting on the mango surfaces, irregular mango shapes, shading caused by overlapping fruits, and shadow variations induced by daily weather conditions (Karkee et al., 2012; Gongal et al., 2015). These uncontrollable and complex factors can significantly diminish the accuracy of mango identification. Consequently, it is essential to develop robust mango recognition algorithms and integrate them into mango-picking robots to effectively accomplish the task of mango detection.

To address the challenges faced in mango picking tasks, researchers have primarily focused on mango detection. In the domain of mango detection, vision researchers have made significant advances. Koirala et al. (2019) have developed a modified YOLO architecture, termed MangoYOLO, which was deeper than YOLOv1 but shallower than YOLOv3 to optimize the speed of mango detection. Xu Z F et al. (2020) conducted experiments with green mangoes and proposed a lightweight green mango detection method based on YOLOv3. J. S. Ignacio et al. (2022) employed the YOLOv5 to identify mangoes and used the CIELAB color space to classify the

* Corresponding author.
E-mail addresses: xg.song@hotmail.com (X. Song)
Digital Object Identifiers: https://doi.org/10.62953/IJAMCE.442383

maturity of the mangoes.

To comprehensively address the challenges associated with mango picking tasks, this study focuses on reducing redundant computations, decreasing the number of algorithmic parameters, and enhancing the feature extraction capabilities of the detection heads. Building on YOLOv8, this study introduces a lightweight mango detection algorithm, called CS-YOLOv8. The main contributions of this work are as follows:

(1) **Replacement of the backbone feature extraction network:** To address the issue of redundant computations caused by the frequent use of regular convolutions in the YOLOv8's backbone, we have adopted FasterNet (chen et al., 2023) as a replacement for the original backbone. FasterNet optimizes the regular convolution structure by using PConv, thereby reducing redundant computations and memory access. This approach significantly decreases the number of algorithmic parameters and computational complexity, while still effectively extracting spatial features.

(2) **CS-FasterNetBlock:** In order to solve the problem of accuracy loss caused by FasterNet replacement of the backbone, we revisit the structure of the FasterNetBlock within the FasterNet. Building upon the PConv and integrating the channel shuffle mechanism proposed by the ShuffleNet (Zhang et al., 2018), we have developed a novel convolutional module, termed CS-PConv. In CS-PConv, a channel shuffle mechanism is introduced subsequent to the partial convolution layers, which enhances the intercommunication between convolved and non-convolved channels, thereby augmenting the capability of feature extraction from PConv. Subsequently, we have substituted the 1×1 convolutional layers in the FasterNet Block with CS-PConv, culminating in the design of the CS-FasterNetBlock. This refinement not only expands the receptive field and enriches the diversity of output features but also achieves enhanced detection accuracy with minimal additional cost.

(3) **SP-Head:** In order to solve the problem of large computational resource consumption caused by the multiple use of 3×3 regular convolutions in the detection head in the original YOLOv8, this study introduces the concept of shared parameters (Yerimah et al., 2024) and integrates the CS-FasterNetBlock into the detection head. The new detection head is named SP-Head. By sharing detection heads of YOLOv8, the number of parameters is significantly reduced. With the CS-FasterNetBlock, the receptive field is further improved and the detection head feature extraction is enhanced. In ablation experiments, the SP-Head is not only more lightweight, but also has higher mango detection accuracy.

2. Related Work

2.1 YOLOv8

Published in 2023, YOLOv8 was designed to integrate the most effective features of various real-time object detection models. YOLOv8 offers models in a range of sizes, including YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x, to accommodate diverse application scenarios. To minimize computational redundancy in simpler tasks with complex models, we selected the smallest model, YOLOv8n, after balancing detection accuracy and model parameter quantity. This model consists of three primary components: the Backbone, Neck, and Head, as shown in Fig. 1.

Backbone: YOLOv8 utilizes an enhanced CSPDarknet53 (Mahasin et al., 2022) as its backbone network, responsible for feature extraction tasks, which includes the Conv, C2f, and SPPF modules. The Conv module performs regular convolution, Batch Normalization (BN), and Sigmoid Linear Unit (SiLU) on the input image to produce the output. The C2f structure, designs for learning residual features, employs gradient shunt connections to ensure a rich flow of network information. The SPPF module converts feature maps of arbitrary sizes into fixed-size feature vectors by sequentially passing through three 5×5 maxpool operations, followed by cascading. This design reduces the computational load, effectively decreasing the algorithm's latency.

Neck: The Neck component primarily facilitates the integration of multi-scales features, composed of two main elements: the Feature Pyramid Network (FPN) (Lin et al., 2017) and the Path Aggregation Network (PAN) (Liu et al., 2018). FPN constructs a feature pyramid to extract feature maps and achieve feature fusion across various levels. In contrast, PAN enhances the fusion process by utilizing convolutional layers to merge feature maps while preserving spatial information. The PAN-FPN combines up-sampling and down-sampling techniques in order to fuse shallow positional information with deeper semantic information, enhancing the transmission of localization and semantic features and significantly improving object detection across multi-scales.

Head: The detection head of YOLOv8 employs a widely-used decoupled head structure, comprising two heads: the category detection head and the bounding box detection head. This architecture captures category and location data from objects at various scales via three distinct sets of feature maps, each varying in size.

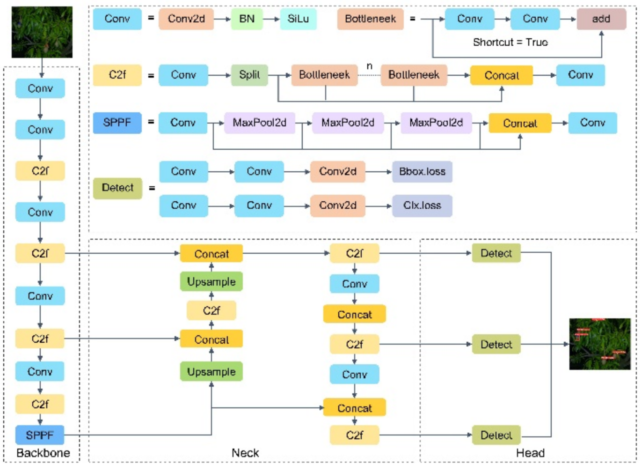

Fig. 1. YOLOv8 structure.

3. Design of CS-YOLOv8 algorithm

This study addresses the issues of high memory usage in mango detection models by improving the YOLOv8 model in three key areas. Firstly, it is experimentally confirmed that substituting YOLOv8's backbone network with FasterNet significantly reduces both the algorithm's parameter counts and its computational complexity. Secondly, the CS-PConv convolutional module is developed to enhance information exchange among convolutional channels by incorporating the channel shuffle mechanism from the ShuffleNet. The CS-FasterNetBlock is created by substituting the 1×1 convolutional layer in the FasterNetBlock with the CS-PConv, which broadens the receptive field and enhances the model's detection accuracy. Finally, we devise the SP-Head by amalgamating the concept of shared weights with the CS-FasterNetBlock. This SP-Head not only addresses the issue of excessive parameters due to computational redundancy in the target detection head but also elevates the precision of target detection. The improved model is named CS-YOLOv8 and its network structure is shown in Fig. 2.

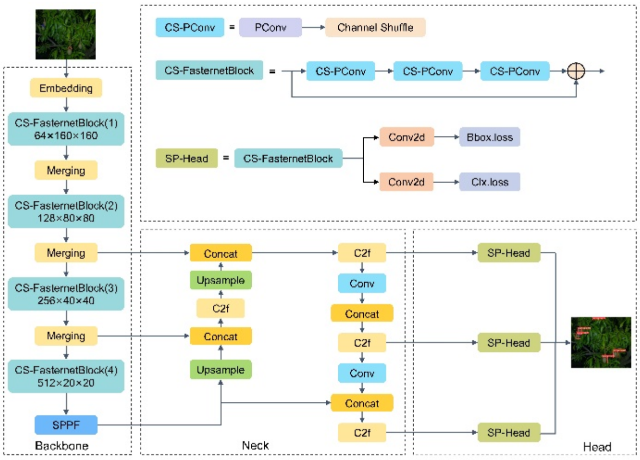

Fig. 2. The CS-YOLOv8 network structure. In its backbone, a channel feature fusion module, CS-FasterNetBlock, is designed to embed into the FasterNet network as the backbone of feature extraction. In its neck, PAN-FPN fuses shallow positional information with deep semantic information. In its head, three shared parameter detection heads with different scales are designed for feature prediction.

3.1 FasterNet and PConv

The extensive use of regular convolutions alongside C2f in the backbone network of YOLOv8 leads to significant redundant computations and a high volume of floating-point operations (FLOPs), resulting in increased model latency. The relationship between latency, FLOPs, and floating-point operations per second (FLOPS) is represented as Eq. (1).

$$Latency = \frac{FLOPs}{FLOPS} \tag{1}$$

FasterNet is a lightweight network proposed by chen et al. in 2023 with a network backbone that reduces latency and improves computational speed by increasing the number of FLOPS while effectively reducing the number of FLOPs. PConv is a convolution module in the FasterNet network, and its working principle is shown in Fig. 3. The PConv module efficiently extracts spatial features by minimizing computational redundancy and memory accesses, thereby reducing both the computational load and parametric volume of the network, while maintaining a constant output feature map and channel size.

Due to the high similarity of feature maps across various channels, employing regular convolution for feature extraction can result in computational redundancy and increased memory usage. The PConv module selectively applies regular convolution operations to certain input channels, leaving the remaining channels unchanged, thereby enhancing the network's capacity to distill key features from numerous similar and redundant feature maps. The FLOPs associated with the PConv module are represented as Eq.(2):

$$FLOPs_{PConv} = h \times w \times k^2 \times c_p{}^2 \tag{2}$$

In the formula, h and w are the width and height of the feature map, k is the size of the convolution kernel, and c_p is the number of channels for conventional convolution. In actual implementation, c_p is 1/4 of c_p, so the FLOPs of PConv is only 1/16 of that of regular convolution.

Chen et al. proposed a FasterNetBlock based on the PConv, comprising a PConv layer followed by two sequentially connected 1x1 convolutional layers. This module is characterized by its simplicity, minimal parameter count, and high processing speed. Its structure is shown in Fig. 3. Furthermore, excessive utilization of normalization and activation layers may result in diminished feature diversity, so only BN and Rectified Linear Unit (ReLU) are added to the two 1x1 convolutional layers of FasterNetBlock. Among these, BN enhances training speed and accuracy, whereas ReLU functions as an activation mechanism to expedite model training and mitigate the issue of gradient vanishing (Chen et al., 2023). Strategically positioning normalization and activation layers between the two 1x1 convolutional layers within each FasterNet module optimizes latency and preserves feature diversity.

In response to the challenges of large model parameters and difficulty in hardware deployment due to computational redundancy in the YOLOv8, apply to mango detection tasks, replacing the backbone network of YOLOv8 with FasterNet can decrease both computational and memory access, thereby rendering the backbone more lightweight.

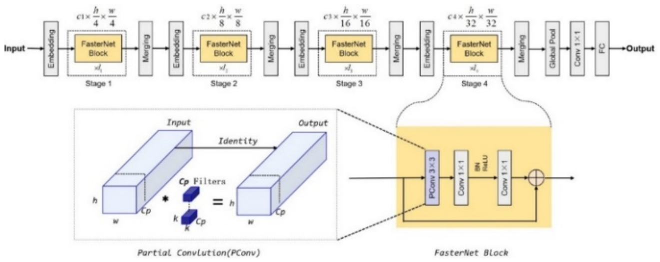

Fig. 3. FasterNet network structure.

3.2 CS-FasterNetBlock

Although PConv applies regular convolution to select input channels for feature extraction while keeping the rest of the channels unchanged, thereby significantly reducing memory access, this convolutional operation is restricted to the channels within each group. Consequently, there is no exchange of information between the convolutional and non-convolutional channels, which diminishes the network's prediction accuracy. Therefore, to address the decrease in accuracy resulting from the substitution of YOLOv8's backbone network with the FasterNet network, this paper proposes the CS-PConv convolution module, which adds the channel shuffle mechanism after PConv. This structure facilitates information exchange between each channel group following partial convolution, boosts the model's capacity to acquire global information, and increases the accuracy of network predictions. Simultaneously, substituting some of the 1x1 convolutions with channel shuffle can significantly reduce the number of parameters and expedite the feature fusion process among convolutional channels. The CS-PConv structure is shown in Fig. 4.

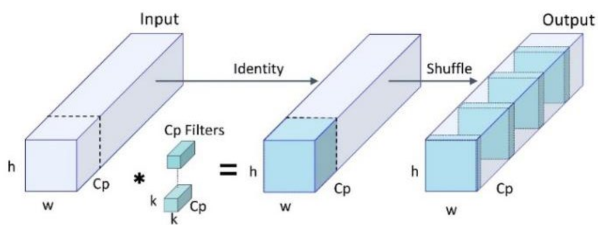

Fig. 4. CS-PConv structure.

The channel shuffle mechanism was first introduced in the ShuffleNet network, and its structure is shown in Fig. 5. This mechanism reorders the channel groupings of the feature map to enhance feature communication across different groups. In traditional grouped convolution, the channels of the feature map are segmented into multiple groups, enabling convolution operations to take place within each group's channels. Since the channels from different groups operate independently, this approach substantially reduces computational complexity. However, this grouping structure also results in information isolation among different group channels, thereby reduce the network's accuracy. Introducing channel shuffle disrupts this isolation, facilitating increased information exchange among groups, which enhances both the performance and accuracy of the network. Divide the number of input channels n equally into group groups, with each group containing n//group of channels. Change the shape of the input channel number to (group, n//group). The obtained new channel can be transposed and then flattened to complete the channel shuffle. In response to the restricted information exchange between convolutional and non-convolutional channels resulting from PConv in the FasterNet network, which diminishes network accuracy, a channel shuffle mechanism has been incorporated following PConv to enhance network accuracy.

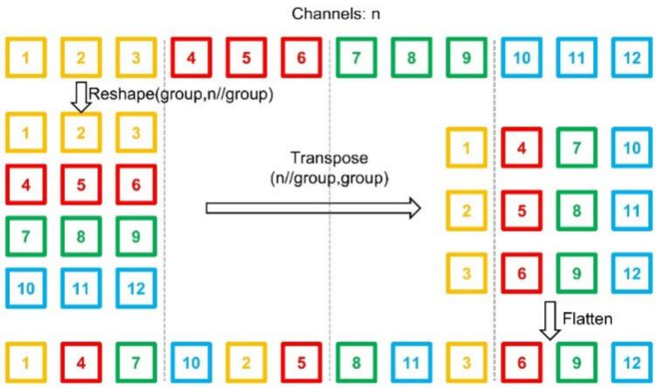

Fig. 5. Channel Shuffle structure.

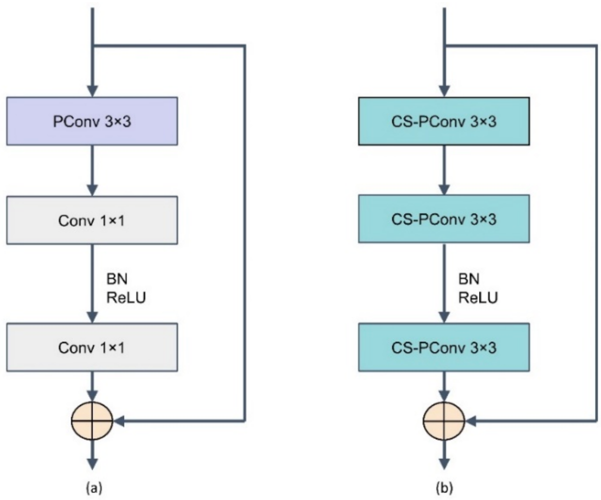

Fig. 6. (a) FasterNetBlock structure (b) CS-FasterNetBlock structure.

In this study, we reexamined the structure of FasterNetBlock. This block incorporates two 1x1 convolutional layers following PConv, which reduce the number of parameters and accelerate training speed. However, the receptive field of 1x1 convolutions is relatively small, limiting the acquisition of global features. To address these issues, this paper proposes CS-FasterNetBlock based on FasterNetBlock and CS-PConv, as shown in Fig. 6. Firstly, replacing PConv with CS-PConv enhances the information exchange between convolutional and non-convolutional channels, and improves the model's ability to obtain and interact with global information. Secondly, CS-PConv is used to replace the two 1x1 convolutional layers in the FasterNetBlock to increase the receptive field and reduce the loss of effective mango feature information, thereby optimizing the accuracy of the model.

3.3 SP-Head

The head of YOLOv8 utilizes a popular decoupled head structure, distinguishing the classification head from the detection head. The detection head identifies the category and position information of objects by using three identical detection modules on three feature maps of varying sizes (Qu et al., 2023). Each detection module comprises a category detection branch and a bounding box detection branch. The specific structure is shown in Fig. 7.

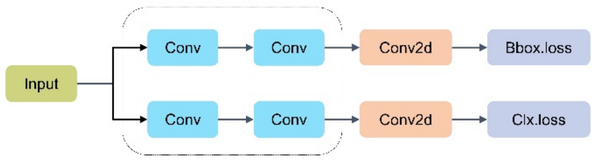

Fig. 7. Structure of the original YOLOv8 head.

We reexamine the Head section of YOLOv8 and discover that it is consists of three detection modules, each featuring three detection branches. Each branch includes two 3x3 convolutional layers and one 1x1 convolutional layer. Consequently, there are 12 layers of 3x3 convolutions in the Head section of YOLOv8. Due to the use of a large amount of regular convolution in the detection module, this part of the parameter counts accounts for 25.0% of the entire network architecture. Considering that the focus of this paper is on mango picking tasks, which are conducted in outdoor environments, lower computational complexity and lightweight models are more advantageous for actual hardware deployment. Based on the concept of shared parameters, we redesign the Head section of YOLOv8. Replace multiple regular convolutions with CS-FasterNetBlock, which aims to further reduce the parameter count and maximize the utilization of convolution operations, thereby reducing computational redundancy in the detection head. Simultaneously, CS-FasterNetBlock expands the receptive field of the detection head and enhances its feature extraction ability for mango target. In addition, the structure of the category detection branch and bounding box detection branch remains unchanged. We rename this new detection head as SP-Head, as shown in Fig. 8.

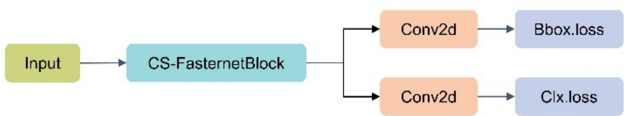

Fig. 8. SP-Head Structure.

4. Experimental results and analysis

4.1 Experimental environment

The experimental environment configuration for this paper is shown in Tab. 1. The training phase of this experiment adopts unified parameters. The Stochastic Gradient Descent (SGD) (Ruder et al., 2017) optimizer is used, with an initial learning rate of 0.01, weight attenuation coefficient of 5×10-4, momentum parameter of 0.937, batch size set to 2, and training epochs of 300 epochs.

Tab. 1. Experimental parameter configuration.

Name	Configuration
Operating system	Windows 10
Development Languages	Python 3.8.17
Frameworks	Pytorch1.13.1+cuda11.7
CPU	Inter Core i7-7700
GPU	GeForce GTX 1660 Ti (8G)

4.2 Experimental data

This paper uses the open dataset MangoYOLO (Koirala et al., 2023) proposed by the Central Queensland University (CQU) as the mango object detection dataset. This dataset aims to detect mango fruits in tree canopy images. A total of 1730 images were obtained using a 5-megapixel RGB digital camera and 720W LED floodlighting on an agricultural multifunctional vehicle running at a speed of 6 kilometers per hour at night. These images are randomly divided into a training set (1300 images), a validation set (215 images), and a testing set (215 images).

4.3 Evaluation Metrics

In this study, Precision(P), Recall(R), and mAP0.5:0.95 are used as evaluation indicators for object detection. The calculation method for mAP is represented as Eq. (6).

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

$$AP = \int_0^1 Precision\, d\, Recall \tag{5}$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \tag{6}$$

In the Eq. (3), TP represents the number of samples predicted as positive for a positive class, FP represents the number of samples predicted as positive for a negative class, and FN represents the number of samples predicted as negative for a positive class. In the Eq. (5), The P-value and R-value are two contradictory parameters. Ideally, a detection model with both high P-value and R-value would have better performance. Therefore, it is necessary to combine the Average Precision (AP) to combine these two evaluation indicators and evaluate the performance of the model. AP is equal to calculating the integral of P corresponding to each R. The mAP value can be obtained by adding up the AP values of detection results from different categories and dividing them by the total number of categories (denoted as N). mAP0.5 represents the average precision of the detection model when its IoU is set to 0.5, and mAP0.5:0.95 represents the average precision of the detection model when its IoU is set to 0.5 to 0.95 (with an interval of 0.5 values).

In addition, this study selects GFLOPs and Params as evaluation metric to measure the size of the model.

4.4 YOLOv8 version comparative study

To cope with different detection tasks, YOLOv8 provides models of different sizes at n/s/m/l/x scales, with different focuses on processing speed and recognition accuracy, providing diverse choices to meet different computing resources and real-time processing needs. In order to avoid computational redundancy of complex models for simple tasks, we conduct a comparative experiment on five versions of YOLOv8, and the experimental results are shown in Tab. 2.

By conducting a comparative study of different versions of YOLOv8, we find that while other models slightly improve recognition accuracy compared to YOLOv8n, the increase is minimal. Among them, YOLOv8l has the highest accuracy but only improves by 0.5%. However, the number of parameters and GFLOPs of YOLOv8l is 20 times that of YOLOv8n. This significant increase in model complexity is disadvantageous for deployment in hardware environments with limited computational resources, such as in agricultural scenarios. Further analysis shows that the R of the other four versions of YOLOv8 are not as good as that of YOLOv8n. This phenomenon can be attributed to the increased number of convolutional kernels in the network, which, while enhancing the depth of feature extraction, also leads to the loss of some mango features in the original images, thereby reducing the recall rate. In agricultural scenarios with limited hardware computational resources, using complex models for simple tasks results in computational redundancy. Therefore, to balance detection accuracy and model parameters, we ultimately choose the smallest model, YOLOv8n, as the detection model. This choice not only ensures high detection accuracy but also effectively reduces the consumption of computational resources, making it well-suited for the practical needs of agricultural scenarios.

Tab. 2. YOLOv8 version comparative experiment. n/s/m/l/x are different sizes of YOLOv8 models, which change the depth of the network and the number of convolutional kernels.

Model	P/%	R/%	mAP0.5:0.95/%	Params/M	GFLOPs/G
YOLOv8n	96.0	96.6	73.3	3.0	8.2
YOLOv8s	96.3	96.1	73.7	11.1	28.4
YOLOv8m	95.8	96.0	73.5	25.9	79.1
YOLOv8l	96.3	96.1	73.8	43.6	165.4
YOLOv8x	95.9	96.0	73.7	68.1	257.4

4.5 Ablation study

In this section, we will validate the effectiveness of our respective modules on the algorithm through three ablation experiments, specifically targeting Fasternet, CS-FasternetBlock, and SP-Head. The number of iterations for each ablation experiment is the same.

As shown in Tab. 3, replacing the backbone of YOLOv8 with FasterNet significantly reduces the number of parameters by 43.3%, which is more conducive to model hardware deployment. However, this significant reduction in parameter quantity comes at a cost, resulting in a slight decrease in mAP0.5:0.95 at the mango detection from 73.3% to 71.8%. In order to further improve the accuracy of the model, we reexamine the FasternetBlock and reconstruct the module CS-FasternetBlock to enhance the communication between channel information and improve the receptive field. The reconstruction of this module makes the mAP0.5:0.95 increase by 1.3%, and the parameter quantity remains unchanged. Finally, by integrating the concept of shared parameters with the CS−FasternetBlock module, a new shared detection head SP-Head is reconstructed. Compared to the original network, YOLOv8 achieves a 63.3% reduction in parameter count, a decrease of 6.0 GFLOPs, and an improvement of 0.6% in mAP0.5:0.95.

Tab. 3. Ablation experiment. Where A represents Fasternet, B represents SC-Fasternet Block, and C represents SP-Head.

A	B	C	P/%	R/%	mAP0.5:0.95/%	Params/M	GFLOPs/G

			96.0	96.6	73.3	3.0	8.2
✓			95.1	95.6	71.8	1.7	5.1
✓	✓		95.4	95.8	73.1	1.7	5.1
✓	✓	✓	96.4	96.6	73.9	1.1	2.2

The red square in Fig. 9 indicates the predicted result of mango target recognition. Based on the comparison of effects in Fig. 9, the following conclusions can be drawn. 1) The improved algorithm demonstrates higher confidence in mango detection. For instance, in the first, second, third, and fourth rows, the confidence scores for mango predictions by the enhanced model surpass those of YOLOv8 and are more accurately aligned with the labeled positions. This improvement is attributed to the SP-Head's feature extraction module, CS-FasternetBlock, which focuses more on mango-specific features and refines their extraction. 2) In terms of identifying occluded mango targets, our improved algorithm significantly outperforms the original network. It accurately detects targets obscured by leaves, enhancing the recognition accuracy of occluded targets. For example, in the second and third rows, the proposed algorithm successfully identifies these occluded targets. 3) The proposed method accurately recognizes mango targets in both simple and complex backgrounds, as demonstrated in the first and fourth rows. Moreover, our model significantly reduces the parameter count and computational complexity of YOLOv8, making it more suitable for deployment on hardware with limited computational resources.

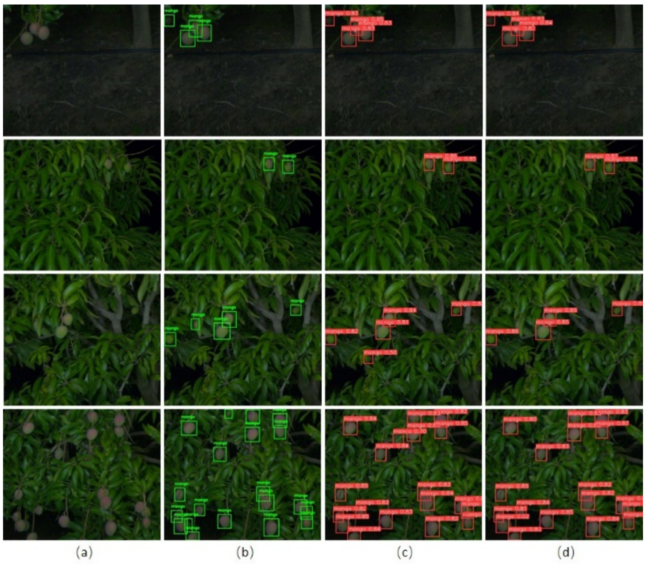

Fig. 9. Example results of our model on On-tree mango instance segmentation dataset. (a)image, (b)label image, (c)YOLOv8, (d)CS-YOLOv8.

4.6 Comparative experiments

In this section, we aim to illustrate the advantages of CS-YOLOv8 in mango detection. Regarding mango recognition, we evaluate several advanced object detection models, comprising lightweight models such as YOLOv5n (Jocher et al., 2022), YOLOv6 (Li et al., 2022) and YOLOv7-Tiny (Wang et al., 2023). These models underwent training and testing utilizing the MangoYOLO dataset, with evaluation metrics including P, R, and mAP0.5:0.95 for object detection.

As shown in Tab. 4, we evaluate our model on the MangoYOLO dataset. Based on the results, the following conclusions can be drawn. 1) The CS-YOLOv8 network achieved a mAP0.5:0.95 metric of 73.9% on the MangoYOLO dataset. Compared to lightweight networks YOLOv5n, YOLOv6 and YOLOv7-tiny, there is a slight advantage in target recognition accuracy. The reason for this is that the CS-FasterNetBlock module can enhance information exchange between channels and greatly improve the receptive field, enhancing the backbone network's ability to extract target recognition features. 2) The CS-YOLOv8 network has reached the lowest level in terms of parameter count and GFLOPs. Compared to the lightest YOLOv5n in the YOLO series, GFLOPs have decreased by 4.9G. This is because the SP-Head reduces information redundancy, significantly reduces parameters, and integrates the CS-FasterNetBlock module, without causing a decrease in its accuracy.

Tab. 4. Comparative experiment on MangoYOLO dataset.

Model	P/%	R/%	mAP0.5:0.95/%	Params/M	GFLOPs/G
YOLOv5n	96.7	94.9	73.1	2.5	7.1
YOLOv6	95.4	96.4	73.2	3.1	8.4
YOLOv7-tiny	95.1	94.6	72.9	6.1	13.3
YOLOv8n	96.0	96.6	73.3	3.0	8.2
CS-YOLOv8	96.4	96.6	73.9	1.1	2.2

5. Conclusions

The experimental results have proven that our proposed CS-YOLOv8 model can effectively solve the problems of high device memory usage. Our model can effectively detect mango, while maintaining lightweight and without the need for geometric estimation of picking point characteristics. This model has undergone various optimizations and enhancements compared to the original YOLOv8. Through effective analysis of on YOLOv8 version comparative experiments, module ablation experiments, and detection algorithm comparison experiments, we reduce the number of model parameters by 63.3% compared to the original network and decrease GFLOPs by 6.0G and an improvement of 0.6% in mAP0.5:0.95. The future research focus will be on how to better control network parameters and deploy models on embedded devices, while significantly improving the detection accuracy of mango detection.

Acknowledgements

This work is supported by the Fundamental Research Funds for the Central Universities (No. 2682022KJ015), State Key Laboratory of Robotics and Systems (HIT) (SKLRS-2020-KF-13).

References

Tharanathan R N, Yashoda H M, Prabha T N. Mango (Mangifera indica L.),"The king of fruits"—An overview[J]. Food Reviews International, 2006, 22(2): 95-123.

Singh Z, Singh R K, Sane V A, et al. Mango-postharvest biology and biotechnology[J]. Critical Reviews in Plant Sciences, 2013, 32(4): 217-236.

Ntsoane, M.L., Zude-Sasse, M., Mahajan, P., Sivakumar, D.: Quality assesment and postharvest technology of mango: A review of its current status and future perspectives. Scientia Horticulturae. 249, 77–85 (2019)

Fennimore S A, Doohan D J. The challenges of specialty crop weed control, future directions[J]. Weed Technology, 2008, 22(2): 364-372.

Zhou H, Wang X, Au W, et al. Intelligent robots for fruit harvesting: Recent developments and future challenges[J]. Precision Agriculture, 2022, 23(5): 1856-1907.

Mail M F, Maja J M, Marshall M, et al. Agricultural harvesting robot concept design and system components: A review[J]. AgriEngineering, 2023, 5(2): 777-800.

Karkee M, Zhang Q. Mechanization and automation technologies in specialty crop production[J]. Resource Magazine, 2012, 19(5): 16-17.

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

Koirala, A., Walsh, K.B., Wang, Z., McCarthy, C.: Deep learning–Method overview and review of use for fruit detection and yield estimation. Computers and electronics in agriculture. 162, 219–234 (2019)

Xu, Z.-F., Jia, R.-S., Sun, H.-M., Liu, Q.-M., Cui, Z.: Light-YOLOv3: fast method for detecting green mangoes in complex scenes using picking robots. Appl Intell. 50, 4670–4687 (2020). https://doi.org/10.1007/s10489-020-01818-w

Ignacio, J.S., Eisma, K.N.A., Caya, M.V.C.: A YOLOv5-based Deep Learning Model for In-Situ Detection and Maturity Grading of Mango. In: 2022 6th International Conference on Communication and Information Systems (ICCIS). pp. 141–147 (2022)

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12021–12031 (2023)

Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6848–6856 (2018)

Yerimah, L.E., Ghosh, S., Wang, Y., Cao, Y., Flores-Cerrillo, J., Bequette, B.W.: Shared Parameter Network: An efficient process monitoring model. Computers & Chemical Engineering. 181, 108522 (2024). https://doi.org/10.1016/j.compchemeng.2023.108522

Mahasin, M., Dewi, I.A.: Comparison of cspdarknet53, cspresnext-50, and efficientnet-b0 backbones on yolo v4 as object detector. International Journal of Engineering, Science and Information Technology. 2, 64–72 (2022)

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature Pyramid Networks for Object Detection, http://arxiv.org/abs/1612.03144, (2017)

Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path Aggregation Network for Instance Segmentation, http://arxiv.org/abs/1803.01534, (2018)

Chen, B., Dang, Z.: Fast PCB Defect Detection Method Based on FasterNet Backbone Network and CBAM Attention Mechanism Integrated With Feature Fusion Module in Improved YOLOv7. IEEE Access. 11, 95092–95103 (2023). https://doi.org/10.1109/ACCESS.2023.3311260

Qu, W., Zhong, S., Wu, Y., Cao, X.: Development of a real-time pen-holding gesture recognition system based on improved YOLOv8. In: 2023 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML). pp. 1035–1039 (2023)

Ruder, S.: An overview of gradient descent optimization algorithms, http://arxiv.org/abs/1609.04747, (2017)

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., Kwon, Y., Michael, K., Fang, J., Yifu, Z., Wong, C., Montes, D.: ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation. Zenodo. (2022)

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., Wei, X.: YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications, http://arxiv.org/abs/2209.02976, (2022)

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. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7464–7475 (2023)

Xu Fang, received his bachelor's degree from Southwest Forestry University, China, in 2022. Currently, he is studying for a master's degree with the Tangshan Institute, Southwest Jiaotong University, China. His current research interests include object detection, semantic segmentation and robotic arm trajectory planning.

Xingguo Song, Ph.D., graduated from Harbin Institute of Technology, School of Mechanical and Electrical Engineering, majoring in Mechanical Design and Theory, is a visiting scholar at Rice University and a postdoctoral fellow at Johns Hopkins University, USA. His main research interests are intelligent robotics, UAV path planning, bionic robotics, and computer vision.

Yongjiang Li, received his bachelor's degree from Southwest Jiaotong University in 2022. He is currently pursuing a master's degree at the School of Mechanical Engineering at Southwest Jiaotong University. His current research interests include robot software architecture, ROS robotics, and path planning.