Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.046876

ARTICLE

Real-Time Detection and Instance Segmentation of Strawberry in Unstructured Environment

Chengjun Wang[1,2], Fan Ding[2,*], Yiwen Wang[1], Renyuan Wu[1], Xingyu Yao[2], Chengjie Jiang[1] and Liuyi Ling[1]

[1]School of Artificial Intelligence, Anhui University of Science and Technology, Huainan, 232001, China

[2]School of Mechanical Engineering, Anhui University of Science and Technology, Huainan, 232001, China

*Corresponding Author: Fan Ding. Email: 2021200572@aust.edu.cn

Received: 17 October 2023 Accepted: 20 November 2023 Published: 30 January 2024

ABSTRACT

The real-time detection and instance segmentation of strawberries constitute fundamental components in the development of strawberry harvesting robots. Real-time identification of strawberries in an unstructured environment is a challenging task. Current instance segmentation algorithms for strawberries suffer from issues such as poor real-time performance and low accuracy. To this end, the present study proposes an Efficient YOLACT (E-YOLACT) algorithm for strawberry detection and segmentation based on the YOLACT framework. The key enhancements of the E-YOLACT encompass the development of a lightweight attention mechanism, pyramid squeeze shuffle attention (PSSA), for efficient feature extraction. Additionally, an attention-guided context-feature pyramid network (AC-FPN) is employed instead of FPN to optimize the architecture's performance. Furthermore, a feature-enhanced model (FEM) is introduced to enhance the prediction head's capabilities, while efficient fast non-maximum suppression (EF-NMS) is devised to improve non-maximum suppression. The experimental results demonstrate that the E-YOLACT achieves a Box-mAP and Mask-mAP of 77.9 and 76.6, respectively, on the custom dataset. Moreover, it exhibits an impressive category accuracy of 93.5%. Notably, the E-YOLACT also demonstrates a remarkable real-time detection capability with a speed of 34.8 FPS. The method proposed in this article presents an efficient approach for the vision system of a strawberry-picking robot.

KEYWORDS

YOLACT; real-time detection; instance segmentation; attention mechanism; strawberry

1 Introduction

The nutritional value of strawberries is abundant, surpassing that of most fruits on the market due to their high content of vitamins, carotene, and other essential nutrients. Additionally, strawberries possess a robust aroma and a delightful balance between sweetness and sourness, justifying their title as the "Queen of Fruits". Strawberries have a short maturity period and need to be picked in a short time after maturity. Picking at the peak of maturity will cause a serious burden to farmers, and strawberry picking is a labor-intensive industry [1]. To mitigate picking costs, researchers have developed a range of strawberry harvesting robots as substitutes for manual labor [2]. One of the core points of the strawberry-picking robot is strawberry detection. Strawberry detection enables the identification and localization of target strawberries within a given area, providing precise targets for subsequent automated picking. The complexity of the natural environment and the unstructured characteristics of the fruit, however, pose significant challenges to strawberry detection [3,4]. The cultivation methods for strawberries are primarily categorized into table-top, bench-type, elevated planting mode, and high-ridge planting mode [5]. The high ridge planting mode is the predominant method utilized among them; However, it has a more complicated picking environment. The superimposition of strawberries exhibiting varying degrees of ripeness, as well as the impact of foliage, rhizomes, and other factors on strawberry shading and light conditions, significantly influence the detection process of strawberries.

The detection of strawberries has garnered significant research interest over the past decade. The image color segmentation technique employed by Durand-Petiteville et al. is based on a two-dimensional color space [6], enabling the accurate separation of strawberries from strawberry clusters. The RGB channel-based color threshold algorithm was employed by Xiong et al. for strawberry detection [7], yielding an accuracy rate of only 72% with a single image detection time of 2.9 s. Zhao et al. [8] employed an edge detection local operator for the extraction of strawberry contour. Xu et al. [9] proposed a cascade algorithm combining support vector machine (SVM) and histogram of oriented gradients (HOG) for strawberry detection, which extracts and classifies features of strawberries with an accuracy rate of 87%. The aforementioned conventional algorithms, including threshold segmentation, edge detection, and SVM algorithm, primarily employ feature extraction techniques such as strawberry color analysis and edge identification for strawberry recognition. However, these algorithms exhibit limited accuracy and robustness while lacking adaptability to unstructured environments, rendering them unsuitable for practical strawberry-picking detection. The advancement of deep learning has prompted numerous researchers to employ deep learning algorithms for strawberry detection. Sun et al. [10] proposed a strawberry detection model based on improved YOLOv4-Tiny, using GhostNet [11] as the feature extraction network, and integrating multi-scale features to improve the detection accuracy of small targets. The model detection accuracy was 92.62%, and the single image detection time was 5.63 ms. The Rotating YOLO (You Only Look Once) strawberry detection algorithm proposed by Yu et al. [5] achieves an average recognition rate of 94.43% and a single image detection time of 55.56 ms. The aforementioned YOLO-based algorithm, however, utilizes the bounding box to roughly estimate the approximate position of the strawberry, thereby failing to accurately capture its outline and shape features. This limitation is particularly evident in cases involving overlapping strawberries where both missed detection and false detection occur.

The growth pattern of strawberries is characterized by clustered arrangement, resulting in occlusion between the rhizomes, leaves, and strawberries themselves. Accurate localization of target strawberries during harvesting is of utmost importance. In contrast to semantic segmentation and target detection techniques, instance segmentation not only achieves pixel-level classification but also enables precise localization of individual instances [12]. Therefore, instance segmentation currently represents the most effective algorithm for detecting strawberries. The detection of strawberries using Mask-RCNN [13] was initially introduced by Yu et al. [14], who employed ResNet50 as the backbone network and integrated feature pyramid network (FPN) for efficient feature extraction. Notably, this algorithm achieved a remarkable category detection accuracy of 95.78% alongside a segmentation mAP (mean Average Precision) of 45.36. However, the detection speed is only 5 FPS (frames per second). To enhance the detection speed, Pérez-Borrero et al. [15] improved the Mask-RCNN, which introduces a novel deep learning-based strawberry instance segmentation method. This approach eliminates the classifier and bounding box regressor while simplifying the backbone network architecture. Additionally, instead of using non-maximum suppression (NMS), they employ a regional grouping and filtering algorithm for improved efficiency. These modifications collectively streamline the model. Experimental results demonstrate that this method achieves a segmentation mAP of 43.85 with a detection speed of 10 FPS.

The above instance segmentation algorithm, based on Mask RCNN, effectively enhances the accuracy of detection and segmentation. However, its detection speed fails to meet real-time requirements, thereby limiting its applicability in strawberry-picking robots.

To improve the detection speed, Pérez-Borrero et al. [16] proposed a strawberry instance segmentation approach based on a fully convolutional neural network, which includes both the network itself and a grouping filter algorithm. Additionally, by incorporating a centroid prediction network into the U-Net algorithm [17] for each strawberry pixel, the segmentation problem is effectively converted into a precise pixel-level classification task. Transpose of traditional segmentation algorithms into instance segmentation algorithms. The algorithm achieves a seg mAP of 52.61 and operates at a detection speed of 30 FPS. However, it cannot accurately classify strawberries, limiting its segmentation category exclusively to strawberry instances.

The YOLACT [18] algorithm is a real-time instance segmentation method that diverges from the serial approach of Mask R-CNN, instead employing two parallel subtasks. And generates instance masks by predicting distinct coefficients for each instance alongside a shared global mask, thereby obviating the necessity for RoI Align in Mask RCNN to generate local feature maps. Compared to Mask R-CNN, YOLACT exhibits a qualitative improvement in detection speed and can achieve real-time effects. Similar to Mask R-CNN, YOLACT can simultaneously accomplish segmentation, detection, and classification tasks; however, the overall detection accuracy is relatively low.

The attention mechanism can effectively enhance the feature extraction capabilities and is frequently incorporated into backbone networks to bolster network performance. Numerous researchers have employed the attention mechanism in diverse computer vision tasks, including real-time target tracking [19,20], image classification [21], and instance segmentation [22]. Zhang et al. [19] proposed a novel channel and spatial attention mechanism to capture contextual information, which enhances the feature representation of Siamese networks. Chen et al. [21] proposed a parallel mixed attention mechanism integrated into ShuffleNet v2 to effectively suppress irrelevant features and enhance the model's classification capability.

Currently, the real-time detection and segmentation of strawberries remains a formidable challenge due to the intricate and unstructured environment as well as the limited data sets available. Moreover, the presence of foliage often obscures many nearly ripe strawberries, leading to potential misjudgment. Variations in size and shape among strawberries further complicate the accurate segmentation process.

This study applies the speed advantage of YOLACT to real-time instance segmentation of strawberries and designs an efficient YOLACT strawberry instance segmentation and detection algorithm. The main contributions are as follows:

(1) A lightweight and efficient attention module called PSSA module is designed and integrated into ResNet as the backbone net of E-YOLACT;

(2) FPN is replaced with AC-FPN [23] to improve feature representation recognition ability;

(3) FEM is added in the prediction head to further enhance model detection and segmentation capabilities;

(4) Fast-NMS is improved by proposing EF-NMS which effectively reduces the probability of missed detections.

2 Materials and Methods

2.1 Image Data Acquisition

The experimental images were collected from March 01 to March 10, 2023, at the strawberry plantation located in Xinhua Village, Huainan City, Anhui Province, China. The strawberry plantation employs a ridge planting technique and the primary strawberry cultivars include "Hong Yan" and "Feng Xiang", among others, renowned for their vibrant coloration, substantial size, and succulent texture. The image acquisition device used in this study is the iPhone 13, as illustrated in Fig. 1. The mobile phone was positioned on a shooting frame, which was placed at the center of the ditch. The height of the mobile phone from the ground was set to h = 30 ± 15 cm, while the inclination angle ranged between $\theta = 30 \pm 15°$. Under the shooting parameters of a height range of 30 ± 15 cm and an inclination angle range of 30 ± 15°, it aligns with the end effector posture requirements for the strawberry-picking robot. In this configuration, not only can the end effector accurately detect strawberries, but it also allows for easy adjustment of its position to facilitate efficient strawberry picking. A total of 5000 strawberry images were randomly captured at various time periods (morning, afternoon, and evening) and under diverse weather conditions in the strawberry plantation. The images were stored in JPEG format with a resolution of 4032 × 3042.

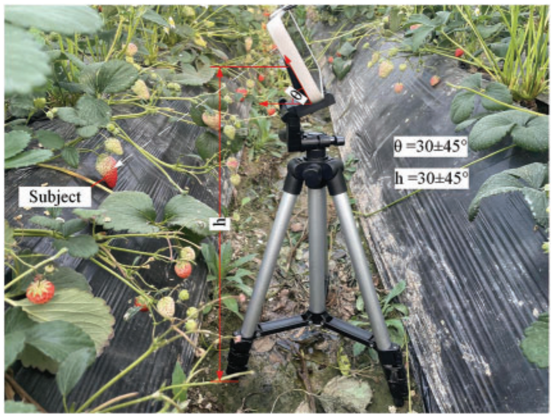

Figure 1: Diagram of the image acquisition process, where h = 30 ± 15 cm and $\theta = 30 \pm 15°$, the white dotted arrow indicates the main subject

2.2 Dataset Construction and Annotation

The custom strawberry dataset was created by randomly selecting 3000 images from the set of captured strawberry images, which included 2800 training sets and 200 validation sets. To test the algorithm's universality, the StrawDI_Db1 dataset [16] is used as the test set in our study. The resolution of the collected images is excessively high, which may adversely impact computational efficiency. Therefore, we resized the images in the dataset to dimensions of 550 ∗ 420 pixels. Utilize Effective interactive segmentation (EISeg) [24] for image annotation, manually segmenting each strawberry within the images, and categorizing them as ripe or unripe, as depicted in Fig. 2. The EISeg is a high-performance tool for interactive segmentation and automatic labeling, which eliminates the need for many contour points to extract segmentation objects. By leveraging pre-trained models, the image is pre-labeled, enabling fine-tuning of target object edges through positive points (within the target object range) and negative points (out-of-range points representing target objects). After comparing the segmentation results obtained from various pre-trained models, we have opted for the Deep High-Resolution Representation Net18_Object-Contextual Representations48 (HRNet18_OCR48) [25,26] static lightweight model architecture.

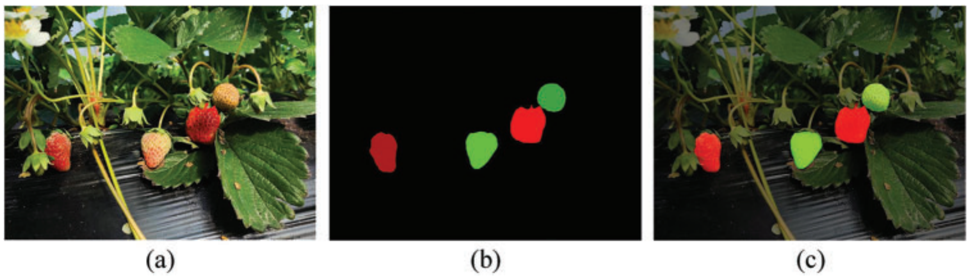

Figure 2: The image annotation process. (a) Original image (b) Mask label image (c) Original image with mask label

2.3 Proposed Methodology

The present study introduces Efficient YOLACT (E-YOLACT), a real-time detection and instance segmentation model for strawberry analysis, which is based on the improvement of YOLACT. This novel approach effectively discerns between mature and immature strawberries while accurately segmenting them. The proposed model framework of E-YOLACT is illustrated in Fig. 3. Compared to target detection and semantic segmentation, instance segmentation poses a more challenging task as it necessitates simultaneous target position detection, pixel categorization, and target segmentation to generate masks. The complexity of the task poses a significant challenge for achieving real-time performance in the instance segmentation algorithm, rendering it unsuitable for integration into the robot detection system. The E-YOLACT framework utilizes the backbone network for image feature extraction and employs the AC-FPN to effectively enhance the receptive field, facilitating the detection of multi-scale features. To improve detection speed, two parallel branch networks were developed, consisting of Prediction Head and Protone. The Prediction Head branch is utilized to generate the confidence, prototype mask coefficient, and anchor location for each candidate frame category. Meanwhile, the Protonet branch is employed to generate K prototype masks for each image. Finally, the prototype mask and its corresponding coefficient are linearly combined, followed by threshold evaluation and output. This approach significantly enhances the model's inference speed.

2.3.1 Backbone Structure

The YOLACT framework utilizes the ResNet101 or ResNet50 backbones for efficient extraction of image features, with ResNet101 consisting of 101 layers and ResNet50 comprising 50 layers [27]. To optimize computational efficiency and reduce parameter complexity, a Bottleneck module is introduced. The design of Bottleneck is illustrated in Fig. 4a. It primarily comprises two 1×1 convolutions, with a 3×3 convolution positioned in the middle. The primary purpose of the 1×1 convolution is to effectively reduce the feature dimension and minimize computational complexity. Taking ResNet50 as an example, Stage 1 comprises three Bottlenecks, while the remaining stages consist of four, six, and three Bottlenecks, respectively. In the case of ResNet101, the Stage 1 to Stage 4 include three, four, twenty-three, and three Bottlenecks, respectively.

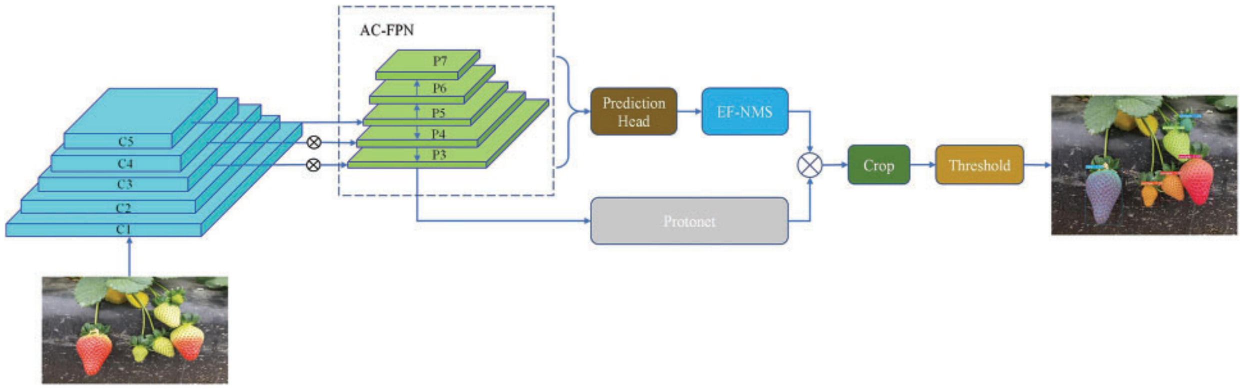

Figure 3: The network structure of the E-YOLACT for visual recognition of strawberries

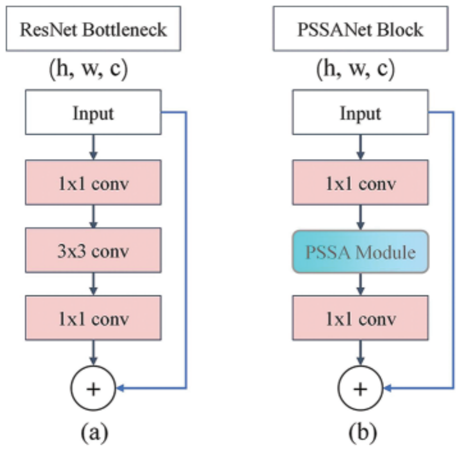

Figure 4: Illustration and comparison of ResNet and our proposed PSSANet blocks

To enhance the performance of ResNet while ensuring computational efficiency, the Bottleneck architecture was improved and PSSANet Block was proposed. The primary enhancement of the PSSANet Block lies in its utilization of the PSSA module to replace the 3 × 3 convolution operation. As illustrated in Fig. 4b, this module not only captures spatial information but also extracts channel-related features. The ResNet50 model is selected for enhancement in this study, wherein the Bottleneck component of ResNet50 is substituted with the PSSANet Block, resulting in the proposal of PSSANet50 as the fundamental architecture for the E-YOLACT.

2.3.2 PSSA Module

The SE mechanism [28] serves as a channel attention mechanism, enabling the model to dynamically adjust channel weights and prioritize valuable channel information. However, it solely focuses on capturing attention in the channel dimension and fails to incorporate spatial attention. The CBAM [29] sequentially integrates the channel attention mechanism and the spatial attention mechanism, enabling comprehensive attention to both spatial and channel aspects of the image. However, this sequential approach results in each subsequent attention mechanism receiving a modified version of its preceding counterpart as the input source. Consequently, feature learning in later stages may be influenced by earlier features, while the high computational complexity of CBAM can impact model training speed and inference time.

To extract richer multi-scale features, inspired by the SA (Shuffle Attention) [30] module and the Pyramid Squeeze Attention (PSA) [31] module, a lightweight and efficient attention mechanism PSSA module is constructed to replace 3×3 convolution. The PSSA module is primarily comprised of five sequential steps, as illustrated in Fig. 5. Firstly, the SPC module is utilized for channel segmentation, followed by conducting multi-scale feature extraction on the feature map of each channel. Furthermore, each feature is bifurcated into two branches along the channel dimension; one branch is dedicated to acquiring channel attention features, while the other focuses on capturing spatial attention features. Specifically, these dual-branch characteristics employ the Channel Attention (CA) Weight module and Spatial Attention (SA) Weight module to extract distinct-scale channel attention and spatial attention, respectively, thereby obtaining corresponding vectors for each scale. Thirdly, the Softmax function is utilized to recalibrate the features of both multi-scale channel and spatial attention vectors. Subsequently, each branch obtains its respective attention weight after undergoing new interactions with multi-scale channels and spatial information. Fourth, the recalibrated weights of the two branches and the corresponding feature maps are element-wise points multiplied, and the output is a multi-scale feature information weighted feature value. The features generated by the last two branches are fused through Concat to generate a feature map with both channel attention and spatial attention weighting. The feature map has richer multi-scale information capabilities.

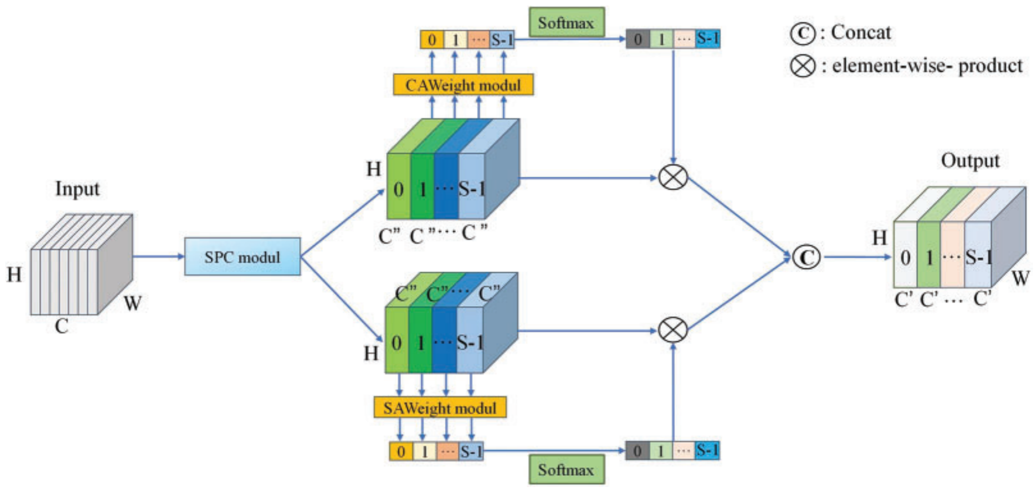

Figure 5: The structure of the proposed Pyramid Squeeze Shuffle Attention (PSSA) module

The SPC operator is utilized for accomplishing multi-scale feature extraction. The design of the SPC operator is illustrated in Fig. 6. Its underlying principle involves targeting the input feature map X and initially dividing it into $[X_0, X_1, \ldots, X_{S-1}]$, comprising a total of S parts, and the number of channels in each part is $C' = C/S$. Each part of the feature map uses multi-scale convolution kernel group convolution to extract feature information of different scales. Group convolution is mainly used to reduce the number of parameters, and the size of the Group is adaptively selected according to the size of the convolution kernel. The specific calculation method of SPC for multi-scale features is presented in formula (1), while the size of the convolution kernel and the Group calculation formula are illustrated in formula (2), respectively.

$$F_i = Conv\left(K_i \times K_i, G_i\right)\left(X_i\right), i = 0, 1, 2, \cdots, S-1 \tag{1}$$

$$\begin{cases} K_i = 2 \times (i+1) + 1, F_i \in R^{C' \times H \times W} \\ G = 2^{\frac{K-1}{2}} \end{cases} \tag{2}$$

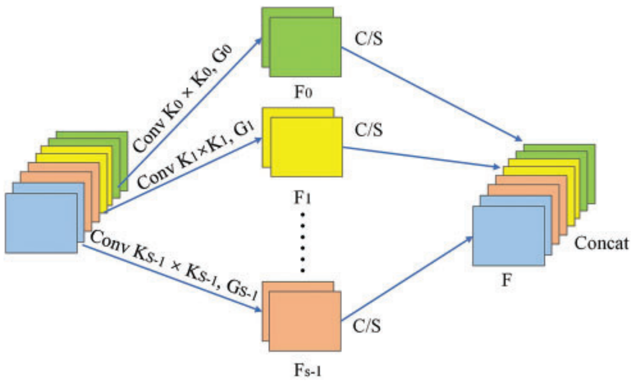

Figure 6: The proposed Squeeze and Concat (SPC) module. The symbol K denotes the size of the convolution kernel, G represents the group size, and F signifies the feature

The SPC module is utilized for acquiring the multi-feature module F_i, which is subsequently partitioned into F_{1i} and F_{2i} based on the channel dimension. The two branches are sent to the channel attention module and spatial attention module in parallel to extract the weights associated with spatial attention and channel attention. The calculation formula for extracting the weights of F_{1i} and F_{2i} is denoted as the formula (3).

$$\begin{cases} Z_{1i} = \text{CAWeight}\left(F_{1i}\right), F_{1i} \in R^{C''} \times H \times W \\ Z_{2i} = \text{SAWeight}\left(F_{2i}\right), F_{2i} \in R^{C'' \times H \times W} \end{cases}, C'' = \frac{C'}{2} \tag{3}$$

$$\begin{cases} Z_1 = Z_{10} \oplus Z_{11} \oplus \cdots \oplus Z_{1S-1} \\ Z_2 = Z_{20} \oplus Z_{21} \oplus \cdots \oplus Z_{2S-1} \end{cases} \tag{4}$$

In formulas (3) and (4), the symbol Z_{1i} and Z_{2i} represent weights for channel attention and spatial attention, respectively; Z_1 and Z_2 represent the complete multi-scale channel attention vector and spatial attention vector obtained by concatenating all multi-scale attention vectors; represents the concatenation operator.

The design of the CA Weight module and the SA Weight module is illustrated in Figs. 7a and 7b, respectively. The role of the CA Weight module is to weigh the importance of each channel to generate more information. First, use global average pooling to embed global information, and generate a 1 × 1 × C$''$ vector. Subsequently, excitation operations [28] are performed to generate attention weights for individual channels. The SA Weight module is mainly used to weigh the importance of spatial information. Complementary to channel attention, first use the Group Normbalization (GN) [32] to perform normalization processing to obtain spatial statistics, and then perform excitation operations to generate spatial attention weights. The calculation formulas of CA and SA Weights are shown in formulas (5) and (6).

$$\begin{cases} g_{c''} = \dfrac{1}{H \times W} \sum_{k=1}^{H} \sum_{j=1}^{W} F_{1i}(k,j) \\ w_{\mathrm{CA}} = \sigma\left(W_1 \delta\left(W_0\left(g_{c''}\right)\right)\right) \end{cases} \tag{5}$$

$$w_{\mathrm{SA}} = \sigma\left(W_3 \delta\left(W_2\left(GN\left(F_{2i}\right)\right)\right)\right) \tag{6}$$

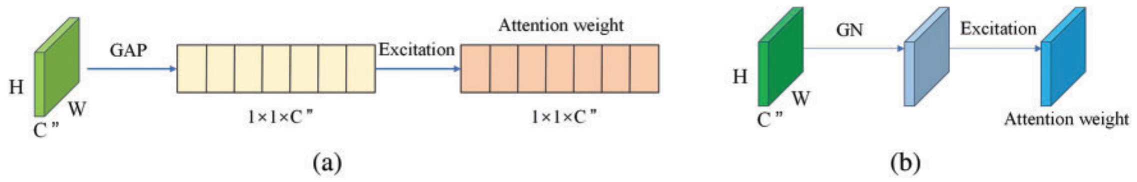

Figure 7: Weight module structure. (a) CA Weight module, (b) SA Weight module

In the above formula, the symbol g_c represents the global average operator, W_0–W_3 represents the fully connected layer (FC), σ represents the Sigmoid activation function, δ represents the use of the Rectified Linear Unit (ReLU) operation [33].

Apply the softmax function to recalibrate and derive new weights, att_{1i} and att_{2i}, for the interaction between local and global attention.

$$\begin{aligned} att_{1i} &= \mathrm{Softmax}\left(Z_{1i}\right) = \frac{exp\left(Z_{1i}\right)}{\sum_{i=0}^{S-1} exp\left(Z_{1i}\right)} \\ att_{2i} &= \mathrm{Softmax}\left(Z_{2i}\right) = \frac{exp\left(Z_{2i}\right)}{\sum_{i=0}^{S-1} exp\left(Z_{2i}\right)} \end{aligned} \tag{7}$$

Subsequently, the newly obtained multi-scale attention weights att_{1i} and att_{2i} are multiplied with their corresponding features F_{1i} and F_{2i} respectively along the channel dimension to generate novel feature representations.

$$\begin{aligned} Y_{1i} &= F_{1i} \odot att_{1i} \\ Y_{2i} &= F_{2i} \odot att_{2i} \end{aligned} \tag{8}$$

where, $\odot$ represents the channel-wise multiplication, the features Y_{1i} and Y_{2i} represent novel characteristics incorporating multi-scale channel attention weights and multi-scale spatial domain attention weights, respectively.

The two novel characteristics are subsequently concatenated to generate a feature Y_i encompassing multi-scale channels and spatial attention weights. Subsequently, the features Y_i from each component are combined to produce the overall feature Y. The Y represents the feature map with multi-scale channels and spatial attention weights generated by the PSSA module.

$$Y_i = \mathrm{Cat}\left(\left[Y_{1i}, Y_{2i}\right]\right) \tag{9}$$

$$Y = \mathrm{Cat}\left(\left[Y_0, Y_1, \cdots, Y_i\right]\right) \tag{10}$$

The model complexity (i.e., network parameters and FLOPs) of PSSANet and ResNet was compared. The parameter amount and FLOPs of PSSANet are 1.2 M and 0.14 G lower than those of ResNet, as presented in Table 1. This comparison demonstrates that PSSANet exhibits a lighter computational load and demands fewer training resources.

Table 1: Comparisons of different attention methods on ImageNet-1k in terms of network parameters (Param.), and floating-point operations per second (FLOPs)

Network	Backbone	Param./M	FLOPs/G
ResNet [27]	ResNet-50	25.56	4.12
PSSANet (Our)		24.76	3.98

2.3.3 AC-FPN

FPN combines low-resolution features with large receptive fields and high-resolution features with small receptive fields to detect objects at different scales by introducing a top-down pathway [34]. However, the top-down pathway merely incorporates additional layers to expand the receptive field without facilitating effective information dissemination, thereby impeding intercommunication of semantic information captured by diverse receptive fields in FPN and subsequently impacting performance.

To address the challenge of achieving effective communication between multi-scale structural fields and ensuring the acquisition of appropriate receptive fields during high-resolution input, the FPN model was substituted with the AC-FPN. The AC-FPN architecture primarily consists of the CEM module and AM module, as illustrated in Fig. 8 [19]. The C5 feature is initially split into two pathways and fed into the dual branches of CEM, wherein one branch consists of dilated convolutions with varying expansion rates to extract features from different receptive fields. The other branch performs an upsampling operation on C5 to preserve the coarse-grained information of the initial input. Then perform a concat operation on the output of the two branches. The utilization of CEM can enhance contextual information and facilitate the detection of objects at multiple scales. To mitigate the superfluous contextual information generated by CEM, AC-FPN introduces the AM module, which primarily captures context correlations through the utilization of a self-attention mechanism.

The C5 is initially processed by both the CEM and AM modules, followed by the fusion of its original features to generate P5. Referring to RetinaNet [35], P2 is not generated, P5 is enlarged by bilinear interpolation and added to C4 after 1×1 convolution to obtain P4. In the same way, P3 is obtained. The P6 feature map is obtained by convolving the P5 feature map and subsequently convolving the P6 feature map to derive the P7 feature map. Utilizing AC-FPN not only ensures enhanced robustness of the mask generated by Protont but also resolves information propagation challenges across distinct receptive fields, thereby improving model performance.

2.3.4 Enhanced Prediction Head

To enhance detection speed, YOLACT simplifies the prediction head and incorporates the concept of shared convolution. Although this design reduces model performance, it effectively promotes computational efficiency. Therefore, FEM is designed and integrated into the prediction head to improve model performance at a slight cost of increased detection time.

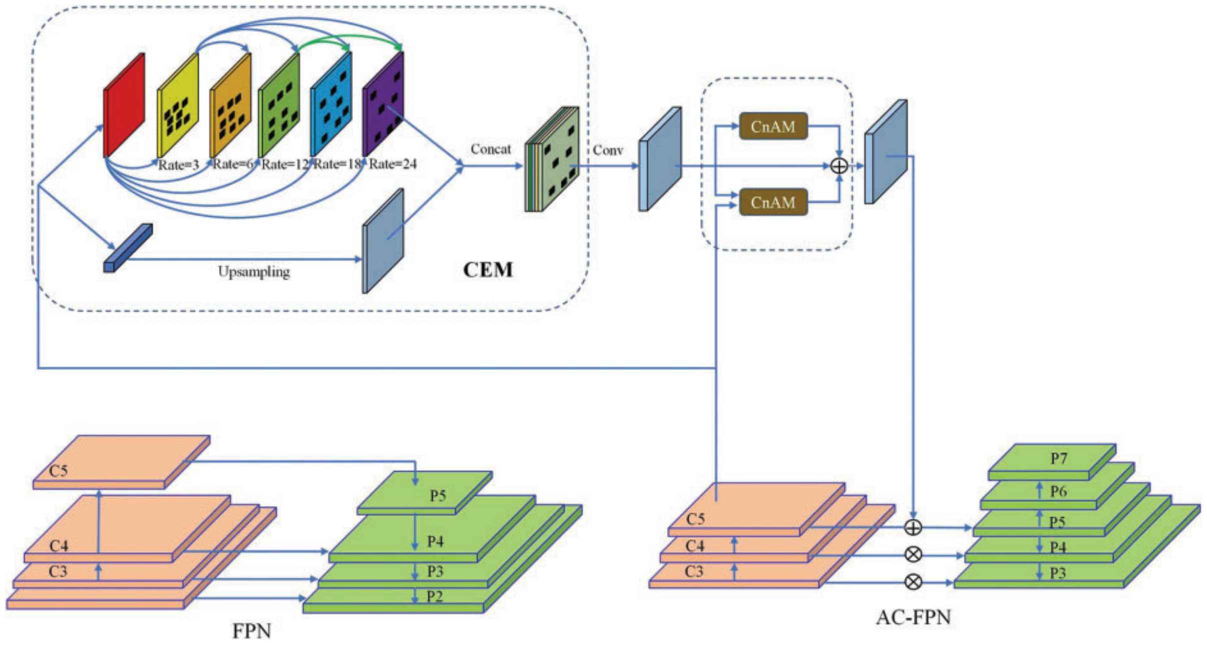

Figure 8: The structure of the AC-FPN

The feature enhancement module consists primarily of three sets of dilated convolutions with varying layers, drawing inspiration from the RFB [36]. As shown in Fig. 9, the output of the 3 × 3 convolution is utilized as the input for the FEM, and initially, feature normalization is performed through a 1 × 1 convolution. Subsequently, the feature map is divided into three segments and individually fed into three branches comprising dilated convolutions with varying layers. The features of the three branches are ultimately integrated through a cascading process. The FEM employs dilated convolution to effectively expand the receptive field, thereby circumventing issues such as resolution reduction and information loss induced by downsampling. Utilizing the FEM can enhance the network's ability to extract information and ensure the generation of high-resolution feature maps, thereby optimizing the positioning and segmentation accuracy of the prediction head.

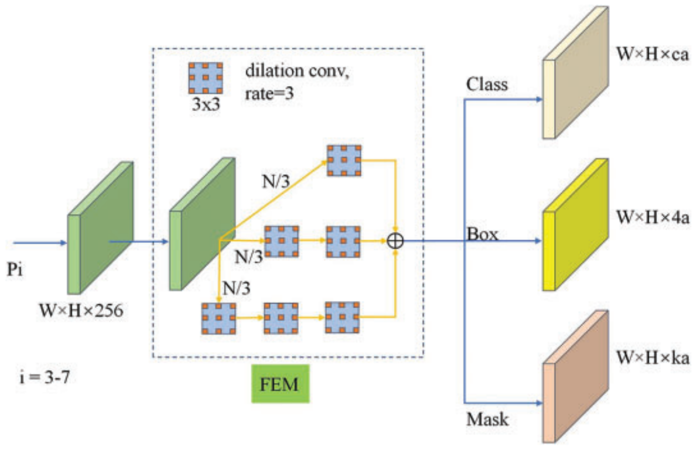

Figure 9: The proposed Feature Enhanced Model (FEM)

2.3.5 Loss Function

The Loss function is utilized to quantify the disparity between the predicted value of the model and the ground truth. The loss function evaluates the disparity between the predicted outcome and the actual label, subsequently providing feedback to the model through backpropagation to optimize parameters and weights. The smaller the value of the loss function, the greater the model's robustness and performance. To effectively optimize the training outcomes of the model, it is crucial to employ diverse types of loss functions tailored for specific tasks. The present study employs a combined loss function L, which consists of linearly weighted classification L_{cls}, mask L_{mask}, and bounding box regression L_{box}, to train the model. The definitions and calculations of L_{box} and L_{cls} are identical to those of $L_{conf\,(x,\,c)}$ and $L_{loc(x,\,l,\,g)}$ [37].

$$L_{mask} = \text{BCE}\left(M, M_{gt}\right) \tag{11}$$

Among them, the M represents the preassembled mask, the M_{gt} denotes the ground truth mask, and pixel-wise binary classification cross-entropy is employed to compute the loss at a per-pixel level.

$$L_{box} = \sum_{\text{i}\in Pos}^{N} \sum_{m\in\{cx,cy,w,h\}} x_{ij}^{k} smooth_{L1}\left(I_{\text{j}}^{m} - \hat{g}_{j}^{m}\right) \tag{12}$$

$$L_{cls} = -\sum_{\text{i}\in Pos}^{N} x_{ij}^{p} \log\left(\hat{c}_{i}^{p}\right) - \sum_{\text{i}\in Neg} \log\left(\hat{c}_{i}^{0}\right) \tag{13}$$

$$\text{smooth}_{L1} = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{14}$$

$$\hat{c}_{i}^{p} = \frac{exp\left(\hat{c}_{i}^{p}\right)}{\sum_{p} exp\left(\hat{c}_{i}^{p}\right)} \tag{15}$$

$$L = L_{cls} + L_{box} + L_{mask} \tag{16}$$

Among them, the i is the prediction box, the g is the ground truth. (cx, xy) is the center of the default box d after compensation (regress to offsets), and (w, h) is the width and height of the default box. The utilization of the $\text{smooth}_{\text{L1}}$ loss effectively mitigates the issue of gradient explosion during the initial stages of training. Additionally, the c represents the softmax loss in a multi-class confidence scenario.

2.3.6 Efficient Fast NMS

The NMS algorithm [38] is employed to suppress non-maximum elements and identify local maximum values. During the evaluation of the detection model, redundant prediction frames can be effectively eliminated through the utilization of NMS. After the conventional NMS generates the regression coefficients for bounding boxes and category confidences, it computes the intersection over union (IoU) between the prediction box with the highest confidence and other prediction boxes. It removes all prediction boxes with an IoU greater than a specified threshold and subsequently selects the prediction box with the highest confidence among those that remain. This process is repeated iteratively by calculating IoU between each remaining prediction box until convergence. The Fast NMS proposed by YOLACT aims to simplify the traditional NMS by employing matrix parallel elimination of prediction boxes, thereby reducing computational load. However, this approach inadvertently eliminates overlapping instances in the prediction frames, leading to a decline in detection performance.

The present article proposes an EF-NMS based on Fast NMS, wherein the initial step involves the computation of a c × n × n matrix to determine the IoU value. The first n detections are sorted in descending order based on the confidence score for each category, and the IoU value is computed between each predicted bounding box. We assign a value of 0 to the lower triangle and diagonal elements of matrix X, while retaining the prediction box with the highest confidence. The maximum IoU value is then obtained based on the column, and for prediction boxes that exceed the predefined IoU threshold N_t, their confidence scores are linearly reduced using a weighted approach. The specific formula for linear weighting is presented in formula (17). Finally, prediction boxes with confidence scores below the threshold S_t are eliminated.

$$S_i = \begin{cases} S_i & IoU\left(M_i, b_i\right) < N_t \\ S_i\left(1 - IoU\left(M_i, b_i\right)\right), & IoU\left(M_i, b_i\right) \geq N_t \end{cases} \tag{17}$$

where, the symbol i denotes the number of iterations performed by the loop on each occasion. The symbol b_i denotes the prediction box to be evaluated in the i-th iteration, M_i represents the highest confidence in the i-th iteration, and S_i represents the confidence score of b_i in the i-th iteration.

3 Experiments and Results

3.1 Experimental Setting and Evaluation Metrics

The experimental hardware platform consists of an Intel(R) Core (TM) i7-12700KF CPU and a GeForce RTX 3060 NVIDIA GPU. The server system employed is Ubuntu 20, while the model is developed using the PyTorch 1.12 framework in conjunction with the Python programming language. The input image resolution is set to 550 × 420, with an initial learning rate of 0.001. Additionally, a weight decay of 5×10^{-4} and a momentum value of 0.9 are employed in the training process. Furthermore, the learning rate is adjusted by multiplying it by 0.1 at iterations 28000, 36000, and 40000, respectively.

The evaluation of the model in this article primarily employs commonly utilized performance metrics such as mean average precision (mAP), AP50, and AP75. The calculation of mAP involves computing the average value of APs across different categories. The AP50 and AP75 metrics correspond to the AP values obtained at IoU thresholds of 50 and 75, respectively. The calculation of AP is intricately linked to the concepts of precision (P) and recall (R). Specifically, AP quantifies the area under the precision-recall curve (P(R)). Precision represents the proportion of samples that are positive among the samples that are predicted to be positive. Recall indicates that the prediction result is the proportion of the actual number of positive samples in the positive sample to the total number of positive samples in the total sample. The calculation formulas of Precision and Recall are shown in formulas (18) and (19), respectively.

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

$$AP = \int_0^1 P(R)\, dR \tag{20}$$

where, the TP means True positive, the FP means False positive, and the FN means False negative. the $P(R)$ represents the $P(R)$ curve.

In the detection task, our given sample classification needs to be combined with the IoU threshold, which quantifies the degree of overlap between two boundaries. The calculation of IoU is illustrated in Fig. 10.

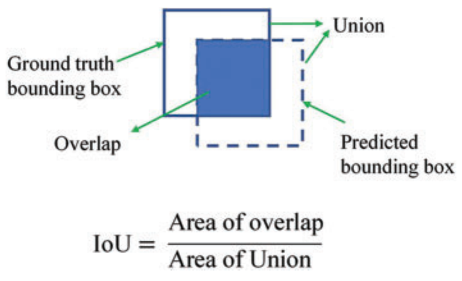
Union
Ground truth bounding box
Overlap
Predicted bounding box
IoU = Area of overlap / Area of Union

Figure 10: The definition of IoU

3.2 Results of Strawberry Detection and Segmentation

To validate the performance of the proposed E-YOLACT algorithm for strawberry detection and instance segmentation, a total of 200 strawberry images were utilized for rigorous testing and evaluation. The comprehensive results are presented in Tables 2 to 4. The visualization of three representative strawberry images is illustrated in Fig. 11.

Table 2: COCO AP results with different models in this study on custom dataset

	Method	mAP (%)	AP_{50} (%)	AP_{75} (%)
Box	YOLACT	64.4	89.1	69.9
	YOLACT++	71.4	90.6	79.7
	Mask-RCNN	76.9	91.7	84.7
	E-YOLACT	77.9	94.5	88.7
Mask	YOLACT	62.2	87.8	64.7
	YOLACT++	68.4	89.2	76.4
	Mask-RCNN	78.7	94.6	89.4
	E-YOLACT	76.6	93.6	88.3

Table 3: Precision rate with different models in this study on custom dataset

Method	Precision (%)		
	Ripe	Unripe	Overall
YOLACT	94.7	88.7	91.7
YOLACT++	95.6	87.6	91.6
Mask-RCNN	95.3	89.6	92.5
E-YOLACT	96.5	90.47	93.5

Table 4: FPS with different models in this study on custom dataset

Evaluation parameter	Method			
	YOLACT	YOLACT++	Mask-RCNN	E-YOLACT
FPS	42.3	35.6	9.7	34.8

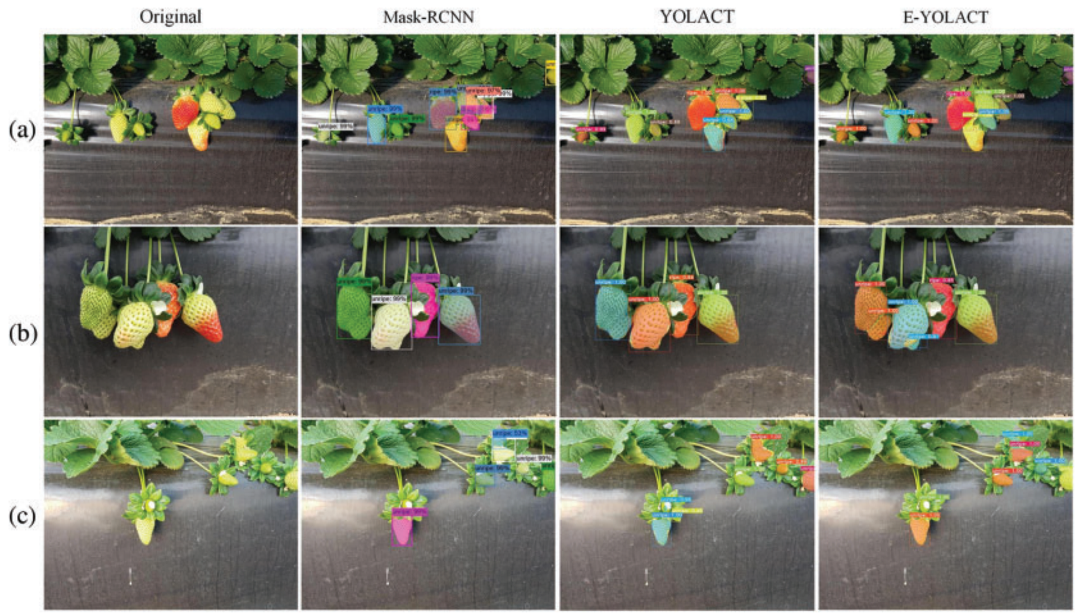

Figure 11: Detection results of three representative strawberry images using different instance segmentation models. The column of original is images of strawberries in unstructured environments for different cases. Mask R-CNN is the result of mask R-CNN algorithm detection. YOLACT is the result of detection by the original YOLACT algorithm. The E-YOLACT is the result of the algorithm detection proposed in this article

As depicted in Fig. 11, the three images (a) to (c) depict distinct growth conditions of strawberries. These images serve as representative examples of challenging strawberry detection scenarios and effectively evaluate the model's performance. The strawberries in (a) have different sizes, which can be effectively used to test the model's ability to detect multi-scale objects. While employing the Mask R-CNN algorithm, there are instances of misdetections where a single strawberry is segmented into two separate entities; However, Mask R-CNN accurately segmented the edges of the strawberries for those instances that were correctly detected. Although YOLACT does not have the problem of false detection compared to Mask R-CNN, its segmentation accuracy is extremely low, and cannot accurately segment the edges of strawberries in the detection frame. The E-YOLACT effectively avoids the issue of false detection encountered by Mask R-CNN and can accurately segment the edges of each strawberry. In (b), each strawberry is superimposed and exhibits uniform size. Overlapping strawberries are easily detected as a single entity; both Mask R-CNN and YOLACT erroneously detect the overlapping strawberries as one, whereas the E-YOLACT is capable of effectively and accurately segmenting each overlapping strawberry, demonstrating its proficiency in accurately segmenting and detecting overlapping objects. In (c), the strawberries are mainly concentrated at the edge of the image, exhibiting a uniform green color similar to that of the background. Additionally, some strawberries are partially obscured by leaves and petals, which may lead to potential misidentification as part of the background. Despite the satisfactory performance of Mask R-CNN in contour segmentation, it still exhibits missed detections and fails to capture small strawberries concealed by petals. however, YOLACT can detect strawberries covered by petals, but the segmentation accuracy needs improvement as the generated mask fails to fully encompass the entire strawberry. The E-YOLACT can effectively and accurately detect each strawberry in the picture and segment it accurately. Although there are very few strawberries that are not completely covered by the mask, it has been greatly improved compared to YOLACT.

3.3 Comparison with Other Instance Segmentation Methods

To further analyze the performance of the E-YOLACT model proposed in this article, we employ evaluation metrics including AP, AP50, AP75, FPS, and category accuracy. Additionally, a comparative analysis will be conducted with classic instance segmentation algorithms such as YOLACT, YOLACT++, and Mask-RCNN. The four models were trained and assessed using the same dataset and experimental conditions, as well as a standardized test set. The parameter results are presented in Tables 2 to 4. In the strawberry image test conducted, the Box-map values of original YOLACT, YOLACT++, and Mask RCNN were 64.4, 71.4, and 76.9, respectively; however, the E-YOLACT achieved a Box-map value of 77.9 which is higher than all classic algorithms mentioned above. Compared to the original YOLACT, the E-YOLACT showed an improvement of nearly 21%. Additionally, the E-YOLACT also outperformed the above-mentioned classic algorithms in terms of Box-AP50 and Box-AP75. However, upon comparing the Mask-map values, it is evident that Mask-RCNN achieves the highest performance with a score of 78.7, while YOLACT and YOLACT++ attain scores of 62.2 and 68.4, respectively, both falling short of surpassing the 70 mark. Despite the E-YOLACT's lower Mask-map value compared to Mask RCNN, it still exhibits a commendable score of 76.6, significantly higher than that of YOLACT and YOLACT++. The same trend can be observed for AP50 and AP75 metrics.

Although Mask RCNN's average precision (AP) score is commendable, its FPS falls significantly behind the other three algorithms, with a mere 9.7. In contrast, the YOLACT achieves a maximum FPS of 42.3, while both YOLACT++ and the E-YOLACT exhibit comparable performance with respective FPS values of 35.6 and 34.8, effectively meeting the real-time detection requirements. Regarding Category Precision, including ripe Precision, unripe Precision, and comprehensive Precision, all four algorithms achieve a performance exceeding 90%, with the E-YOLACT achieving the highest scores of 96.5%, 90.47%, and 93.5%, respectively. Furthermore, as depicted in Fig. 11, it is evident that the E-YOLACT effectively distinguishes ripe strawberries.

Compared with YOLACT, although the calculation amount and model size of E-YOLACYT increase, and the detection speed decreases, the performance of the model has been greatly improved. The performance of the E-YOLACYT model can reach or even exceed Mask RCNN. The detection speed, although not comparable to that of YOLACT, significantly surpasses that of Mask RCNN. Among strawberry instance segmentation models with equivalent performance, the E-YOLACT exhibits the highest detection speed.

3.4 Ablation Experiments

To validate the efficacy of various modules and methods proposed in this study for enhancing the performance of YOLACT, the PSSANet50, AC-FPN, FEM, and EF-NMS are sequentially integrated into the YOLACT framework. The evaluation is conducted based on three metrics: Box-mAP, Mask-mAP, and FPS. The experimental results are presented in Table 5. When we replaced ResNet50 with PSSANet50 integrated with the PSSA module, despite a decrease in FPS by 2.7, both Box-mAP and Mask-mAP increased by 6.1 and 9.1, respectively. This substantiates the efficacy of the PSSA module in enhancing feature extraction capabilities; After replacing FPN with AC-FPN, the Box-mAP and Mask-mAP of the model exhibited a respective increase of 3.8 and 2.4. However, there was a corresponding decrease in FPS by 3.4. The utilization of AC-FPN effectively augmented the receptive field and enhanced model performance. The performance enhancement is relatively modest when compared to the decrease in speed; The introduction of the EFM module in the prediction head resulted in a negligible 0.6 drop in model FPS, indicating minimal impact on detection speed. However, both Box-mAP and Mask-mAP exhibited a significant increase of 2.1. By adopting the EF-NMS instead of Fast-NMS as proposed in this study, the model's Box-mAP increased by 1.5 and the Mask-mAP improved by 0.8, thereby enhancing the efficiency of NMS. However, employing EF-NMS resulted in a decrease of 0.8 FPS for the model, which did affect detection speed but with negligible significance. In conclusion, it is evident that each module and method proposed in this article significantly enhances the performance of the YOLACT model.

Table 5: Ablation study of each component in our proposed method on custom dataset

PSSANet50	AC-FPN	FEM	EF-NMS	Box-mAP	Mask-mAP	FPS
				64.4	62.2	42.3
√				70.5	71.3	39.6
√	√			74.3	73.7	36.2
√	√	√		76.4	75.8	35.6
√	√	√	√	77.9	76.6	34.8

4 Discussion

The existing strawberry instance segmentation algorithm faces several challenges, with real-time performance being the most critical one. For instance, references [14,15] proposed the strawberry instance segmentation model based on Mask-RCNN, achieving frame rates of 5 and 10 FPS, respectively, which fall significantly short of meeting real-time requirements. Although the strawberry segmentation algorithm proposed by [16] can reach 30 FPS, meeting the temporal constraints, its applicability is limited to strawberry segmentation only. However, the strawberry-picking robot requires the ability to determine whether the strawberries are ripe during the picking process. We propose an advanced algorithm called E-YOLACT for strawberry detection and instance segmentation. This algorithm not only fulfills the real-time requirements of the vision system employed by the strawberry-picking robot but also effectively identifies whether strawberries are ripe.

The E-YOLACT is an improved version of YOLACT, it is a one-stage algorithm that boasts a simpler model architecture compared to the two-stage Mask RCNN, resulting in significantly faster detection speeds. However, comparative experiments have shown that YOLACT's performance falls short of that achieved by Mask RCNN. To enhance performance, we propose the PSSA module and integrate it into ResNet to augment the performance of YOLACT. The PSSA module effectively incorporates both spatial and channel information during feature extraction, thereby enhancing Backbone's feature extraction capability, and the model complexity of PSSANet is lower than ResNet, but it uses multiple parallel branches spliced with concat. Although accuracy improves, the number of branches affects the GPU's parallel computing rate and efficiency. Element-wise operation reduces FLOPs but occupies GPU computing time. As can be seen from Fig. 11, YOLACT, YOLACT++, and Mask RCNN will miss detection when detecting overlapping strawberries. Considering this issue arises due to NMS, we have made enhancements based on Fast-NMS and proposed the EF-NMS. Our ablation experiments demonstrate that the EF-NMS model significantly improves performance. Furthermore, in conjunction with Fig. 11, the E-YOLACT equipped with the EF-NMS effectively detects overlapping strawberries. However, it is evident from Fig. 11 that the segmentation performance of the E-YOLACT falls short compared to Mask RCNN, with some strawberry masks exhibiting incomplete coverage. The majority of these strawberries are situated at the periphery of the image and primarily consist of unripe ones. Although this situation occurs relatively infrequently, it can result in positioning errors for mature strawberries due to inadequate segmentation accuracy. Consequently, it hampers the robot's ability to reach the ideal picking point during strawberry harvesting, leading to unsuccessful picking or even surface damage to the strawberries. The effectiveness of the E-YOLACT in identifying mature strawberries can be observed in Table 4 and Fig. 11. Although there are occasional errors in strawberry identification, the primary misjudgment occurs when mature strawberries are mistakenly classified as immature ones due to the similarities between some unripe and ripe strawberries. However, since the strawberry-picking robot only selects ripe strawberries during harvesting, this does not impact the practical application of the model, the missed ripe strawberries may rot and cause economic losses if they are not picked for a long time.

The E-YOLACT model, similar to YOLACT, employs a one-stage instance segmentation approach, which effectively minimizes computational overhead and ensures efficient object detection speed. The E-YOLACT framework adopts PSSANet50, as proposed in this article, instead of ResNet50. PSSANet50 incorporates both channel and spatial attention mechanisms, enabling effective multi-scale feature extraction. In comparison to ResNet50, it exhibits superior capabilities in extracting features. The FEM module is incorporated into the prediction head of the E-YOLACT, addressing the issue of diminished detection and segmentation performance resulting from the shared convolution employed in YOLCAT prediction. However, it should be noted that the segmentation performance still falls short when compared to Mask-RCNN. The EF-NMS algorithm employed in the E-YOLACT employs multiple iterations to effectively suppress overlapping boxes, thereby mitigating the issue of false detections encountered in Mask-RCNN and YOLACT (multiple overlapping strawberries are erroneously identified as a single entity). In conclusion, the E-YOLACT demonstrates effective real-time detection of strawberries in an unstructured environment.

Despite improvements in the E-YOLACT's performance through module integration and refinement, it still incurs additional time costs and is limited to high computational load GPUs due to its large network model parameters. Additionally, accurately detecting strawberry ripeness remains a challenge. The lightweight network models are characterized by low complexity and high computational efficiency. For example, Chen et al. [21] proposed a lightweight garbage classification model GCNet with only 1.3 M parameters, achieving an impressive detection time of only 105 ms. Considering the benefits of lightweight networks, we propose integrating the PSSA model into ShuffleNet or MobileNet as the backbone network in future research to effectively reduce model complexity while ensuring accuracy. Moreover, the segmentation performance of the E-YOLACT has not met expectations. To address this concern, we propose integrating FPN input features with other layers to fuse shallow and deep features, aiming to enhance overall segmentation performance by alleviating feature information loss caused by occluded strawberries. The accuracy of the E-YOLACT in detecting low-maturity strawberry categories is relatively limited. To address this issue, we plan to enhance the prediction head by incorporating a lightweight attention mechanism or feature classifier module specifically designed to emphasize strawberry color information.

5 Conclusion

The E-YOLACT model is proposed in this paper to address the real-time detection and segmentation of strawberries in unstructured environments. To enhance the feature extraction capabilities of the backbone network, we propose the PSSA attention mechanism and integrate it into the Bottleneck of ResNet, thereby effectively optimizing the extraction capabilities of spatial and channel features. To solve the problem of communication obstruction between semantic information of different receptive fields in FPN, we introduced AC-FPN. To mitigate the potential reduction in detection performance caused by YOLACT's shared convolution-based prediction head, we propose the integration of the FEM based on multi-layer dilated convolution to bolster the predictive capabilities. To address the missed detection problem caused by Fast NMS, we propose the EF-NMS as a solution that effectively enhances the performance of prediction boxes. Compared to the original YOLACT, the E-YOLACT demonstrates superior performance in detection and segmentation. Specifically, the E-YOLACT achieves box mAP scores of 77.9, 94.5, and 88.7 for AP50 and, AP75, respectively; mask mAP scores of 76.6, 93.6, and 88.3 for AP50, AP75, respectively; category accuracy of 93.5%; and the model's FPS up to 34.8. The E-YOLACT proposed in this article effectively detects and segments strawberries; however, it cannot currently digitally assess strawberry maturity. It can only classify extreme ripeness and immaturity, and the network model of the E-YOLACT is slightly larger with room for improvement in segmentation accuracy. The future work aims to achieve maturity detection and further streamline the network model while enhancing its performance.

Acknowledgement: We express our gratitude to the members of the Strawberry-Picking Robot Research Team from the School of Artificial Intelligence, Anhui University of Science and Technology for their invaluable advice and assistance throughout this study.

Funding Statement: This research was funded by Anhui Provincial Natural Science Foundation (No. 2208085ME128); the Anhui University-Level Special Project of Anhui University of Science and Technology (No. XCZX2021-01); the Research and the Development Fund of the Institute of Environmental Friendly Materials and Occupational Health, Anhui University of Science and Technology (No. ALW2022YF06); and Anhui Province New Era Education Quality Project (Graduate Education) (No. 2022xscx073).

Author Contributions: Study conception and design: Chengjun Wang, Fan Ding, Yiwen Wang; writing-original draft: Chengjun Wang, Fan Ding; methodology: Chengjun Wang, Fan Ding; writing-review & editing: Yiwen Wang, Renyuan Wu; data curation: Renyuan Wu, Xingyu Yao; validation: Yiwen

Wang, Chengjie Jiang, Liuyi Ling; project administration: Renyuan Wu, Liuyi Ling. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] L. Jie, S. Jiao, X. Wang and H. Wang, "A new type of facility strawberry stereoscopic cultivation mode," *China Agricult*, vol. 24, no. 2, pp. 61–68, 2019.

[2] S. Yamamoto, S. Hayashi, H. Yoshida and K. Kobayashi, "Development of a stationary robotic strawberry harvester with a picking mechanism that approaches the target fruit from below," *Japan Agricultural Research Quarterly*, vol. 48, no. 3, pp. 261–269, 2014.

[3] C. W. Bac, E. J. Henten, J. Hemming and Y. Edan, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014.

[4] D. I. Patrício and R. Rieder, "Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review," *Computers and Electronics in Agriculture*, vol. 153, pp. 69–81, 2018.

[5] Y. Yu, K. Zhang, H. Liu, L. Yang and D. Zhang, "Real-time visual localization of the picking points for a ridge-planting strawberry harvesting robot," *IEEE Access*, vol. 8, pp. 116556–116568, 2020.

[6] A. Durand-Petiteville, S. Vougioukas and D. C. Slaughter, "Real-time segmentation of strawberry flesh and calyx from images of singulated strawberries during postharvest processing," *Computers and Electronics in Agriculture*, vol. 142, pp. 298–313, 2017.

[7] Y. Xiong, C. Peng, L. Grimstad, P. J. From and V. Isler, "Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper," *Computers and Electronics in Agriculture*, vol. 157, pp. 392–402, 2019.

[8] J. Zhao, G. Pan and W. Guo, "Application of edge detection local operator to strawberry contour extraction," *Journal of Agricultural Mechanization Research*, vol. 2008, no. 6, pp. 184–186, 2008.

[9] Y. Xu, K. Imou, Y. Kaizu and K. Saga, "Two-stage approach for detecting slightly overlapping strawberries using HOG descriptor," *Biosystems Engineering*, vol. 115, no. 2, pp. 144–153, 2013.

[10] J. Sun, Y. Chen, X. Zhou, J. Shen and X. Wu, "Fast and accurate recognition of the strawberries in greenhouse based on improved YOLOv4-Tiny model," *Transactions of the Chinese Society of Agricultural Engineering*, vol. 38, no. 18, pp. 195–203, 2022.

[11] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu *et al.,* "GhostNet: More features from cheap operations," in *2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, pp. 1577–1586, 2020.

[12] D. Tian, Y. Han, B. Wang, T. Guan, H. Gu *et al.,* "Review of object instance segmentation based on deep learning," *Journal of Electronic Imaging*, vol. 31, no. 4, pp. 41205, 2021.

[13] K. He, G. Gkioxari, P. Dollar and R. Girshick, "Mask R-CNN," in *2017 IEEE Int. Conf. on Computer Vision (ICCV)*, Venice, Italy, pp. 2980–2988, 2017.

[14] Y. Yu, K. Zhang, L. Yang and D. Zhang, "Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN," *Computers and Electronics in Agriculture*, vol. 163, pp. 104846, 2019.

[15] I. Pérez-Borrero, D. Marín-Santos, M. E. Gegúndez-Arias and E. Cortés-Ancos, "A fast and accurate deep learning method for strawberry instance segmentation," *Computers and Electronics in Agriculture*, vol. 178, pp. 105736, 2020.

[16] I. Pérez-Borrero, D. Marín-Santos, M. J. Vasallo-Vazquez and M. E. Gegundez-Arias, "A new deep-learning strawberry instance segmentation methodology based on a fully convolutional neural network," *Neural Computing and Applications*, vol. 31, pp. 15059–15071, 2021.

[17] O. Ronneberger, P. Fischer and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*, pp. 234–241, 2015.
[18] D. Bolya, C. Zhou, F. Xiao and Y. J. Yong, "YOLACT: Real-time instance segmentation," in *2019 IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, Seoul, Korea (South), pp. 9156–9165, 2019.
[19] D. Zhang, Z. Zheng, M. Li and R. Liu, "CSART: Channel and spatial attention-guided residual learning for real-time object tracking," *Neurocomputing*, vol. 436, pp. 260–272, 2021.
[20] D. Zhang, Z. Zheng, T. Wang and Y. He, "HROM: Learning high-resolution representation and object-aware masks for visual object tracking," *Sensors*, vol. 20, no. 17, pp. 4807, 2020.
[21] Z. Chen, J. Yang, L. Chen and H. Jiao, "Garbage classification system based on improved ShuffleNet v2," *Resources Conservation and Recycling*, vol. 178, pp. 106090, 2022.
[22] D. Wang and D. He, "Fusion of mask RCNN and attention mechanism for instance segmentation of apples under complex background," *Computers and Electronics in Agriculture*, vol. 196, pp. 106864, 2022.
[23] J. Cao, Q. Chen, J. Guo and R. Shi, "Attention-guided context feature pyramid network for object detection," arXiv preprint arXiv:2005.11475, 2020.
[24] M. Xian, F. Xu, H. D. Cheng, Y. Zhang and J. Ding, "EISeg: Effective interactive segmentation," in *2016 23rd Int. Conf. on Pattern Recognition (ICPR)*, Cancun, Mexico, pp. 1982–1987, 2016.
[25] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng *et al.,* "Deep high-resolution representation learning for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.
[26] Y. Yuan, X. Chen and J. Wang, "Object-contextual representations for semantic segmentation," in *Computer Vision–ECCV 2020*, pp. 173–190, 2020.
[27] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, pp. 770–778, 2016.
[28] J. Hu, L. Shen and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 7132–7141, 2018.
[29] S. Woo, J. Park, J. Y. Lee and I. S. Kweon, "CBAM: Convolutional block attention module," in *Computer Vision–ECCV 2018 (ECCV 2018)*, Munich, Germany, pp. 3–19, 2018.
[30] Q. L. Zhang and Y. B. Yang, "SA-Net: Shuffle attention for deep convolutional neural networks," in *ICASSP 2021–2021 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, pp. 2235–2239, 2021.
[31] H. Zhang, K. Zu, J. Lu, Y. Zou and D. Meng, "EPSANet: An efficient pyramid squeeze attention block on convolutional neural network," in *Computer Vision–ACCV 2022*, Springer International Publishing, pp. 541–557, 2023.
[32] Y. Wu and K. He, "Group normalization," in *Computer Vision–ECCV 2018. ECCV 2018*, vol. 128, Springer International Publishing, pp. 742–755, 2020.
[33] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML'10: Proc. of the 27th Int. Conf. on Int. Conf. on Machine Learning*, Madison, WI, USA, pp. 807–814, 2010.
[34] T. Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan *et al.,* "Feature pyramid networks for object detection," in *2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 936–944, 2017.
[35] T. Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2018.
[36] S. Liu, D. Huang and Y. Wang, "Receptive field block net for accurate and fast object detection," in *Computer Vision–ECCV 2018*, Springer International Publishing, pp. 404–419, 2022.
[37] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed *et al.,* "SSD: Single shot multiBox detector," in *Computer Vision–ECCV 2016*, Springer International Publishing, pp. 21–37, 2016.
[38] A. Neubeck and L. van Gool, "Efficient non-maximum suppression," in *18th Int. Conf. on Pattern Recognition (ICPR'06)*, Hong Kong, China, pp. 850–855, 2006.