agronomy

Article

ADL-YOLOv8: A Field Crop Weed Detection Model Based on Improved YOLOv8

Zhiyu Jia [1], Ming Zhang [2], Chang Yuan [1], Qinghua Liu [2,*], Hongrui Liu [1], Xiulin Qiu [2], Weiguo Zhao [3,4] and Jinlong Shi [1]

1 College of Computer, Jiangsu University of Science and Technology, Zhenjiang 212003, China; 221710701108@stu.just.edu.cn (Z.J.); yuanc@stu.just.edu.cn (C.Y.); 231210703115@stu.just.edu.cn (H.L.); shi_jinlong@163.com (J.S.)
2 College of Automation, Jiangsu University of Science and Technology, Zhenjiang 212003, China; 221210301129@stu.just.edu.cn (M.Z.); qiuxiulin@njust.edu.cn (X.Q.)
3 College of Biotechnology, Jiangsu University of Science and Technology, Zhenjiang 212003, China; wgzsri@126.com
4 Institute of Sericulture, Chinese Academy of Agricultural Sciences, Zhenjiang 212003, China
* Correspondence: liuqh@just.edu.cn; Tel.: +86-139-1455-7059

Abstract: This study presents an improved weed detection model, ADL-YOLOv8, designed to enhance detection accuracy for small targets while achieving model lightweighting. It addresses the challenge of attaining both high accuracy and low memory usage in current intelligent weeding equipment. By overcoming this issue, the research not only reduces the hardware costs of automated impurity removal equipment but also enhances software recognition accuracy, contributing to reduced pesticide use and the promotion of sustainable agriculture. The ADL-YOLOv8 model incorporates a lighter AKConv network for better processing of specific features, an ultra-lightweight DySample upsampling module to improve accuracy and efficiency, and the LSKA-Attention mechanism for enhanced detection, particularly of small targets. On the same dataset, ADL-YOLOv8 demonstrated a 2.2% increase in precision, a 2.45% rise in recall, a 3.07% boost in mAP@0.5, and a 1.9% enhancement in mAP@0.95. The model's size was cut by 15.77%, and its computational complexity was reduced by 10.98%. These findings indicate that ADL-YOLOv8 not only exceeds the original YOLOv8n model but also surpasses the newer YOLOv9t and YOLOv10n in overall performance. The improved algorithm model makes the hardware cost required for embedded terminals lower.

Keywords: weed detection; ADL-YOLOv8; lightweight model; precision agriculture; attention mechanism

check for updates

Citation: Jia, Z.; Zhang, M.; Yuan, C.; Liu, Q.; Liu, H.; Qiu, X.; Zhao, W.; Shi, J. ADL-YOLOv8: A Field Crop Weed Detection Model Based on Improved YOLOv8. *Agronomy* **2024**, *14*, 2355. https://doi.org/10.3390/agronomy14102355

Academic Editors: Marco Fontanelli and Lorenzo Gagliardi

Received: 14 September 2024
Revised: 7 October 2024
Accepted: 10 October 2024
Published: 12 October 2024

1. Introduction

Weed detection is a critical aspect of agricultural production, involving the identification and management of non-crop plants in farmlands. Weeds compete with crops for nutrients, water, and sunlight, thereby affecting the growth rate and yield of crops [1]. Traditionally, chemical pesticides such as herbicides and insecticides have been the primary methods for weed control. However, these chemicals pose potential hazards to the environment and human health while killing weeds. Chemical pesticides can contaminate soil and water sources, affect crop growth and development, and even lead to a decline in crop quality. Moreover, the long-term use of chemical pesticides may enhance weed resistance, gradually reducing the effectiveness of weed control.

Faced with these issues, many countries and regions have embraced more environmentally friendly and sustainable weed control methods. For example, the European Union has banned the use of 20 high-residue pesticides and is gradually phasing out other pesticides. Since 2022, artificial intelligence has begun to shine, and the agricultural field has also tried to apply artificial intelligence technology to identify weeds in the field. For instance, weed identification technology based on digital image processing can collect farmland images and use image processing techniques to analyze these images to identify weeds [2]. Additionally, weed detection technology based on low-altitude drone imagery and visual algorithms can quickly and accurately detect weeds over a large area.

Several companies provide smart agricultural solutions. Rowbot Systems, for example, has developed a product called "Weedbot", an agricultural robot capable of autonomously identifying and removing weeds. The cost of this product varies depending on scale and functionality, typically ranging from tens of thousands to hundreds of thousands of dollars. Blue River Technology, a company dedicated to developing smart agricultural solutions, offers the "WeedSeeker" product. This product uses advanced weed inspection algorithms and can be integrated with tractors or spraying equipment to achieve precise weed control. The product's price depends on its configuration and scale, usually ranging from several thousand to tens of thousands of dollars. John Deere, a well-known agricultural machinery manufacturer, has integrated weed inspection algorithms into some of its products. For instance, their smart tractors are equipped with advanced vision systems that can identify and handle weeds in the field. The product price varies by model and function, ranging from tens of thousands to hundreds of thousands of dollars. Due to the high cost of these devices, they are not currently suitable for widespread popularization. Therefore, there is a significant need to research and apply fast, efficient, and accurate weed detection technologies.

After collecting images in ornamental lawns and sports lawns, Parra et al. (2020) [3] tested 12 edge detection filters. They applied aggregation techniques to the results of three effective filters and determined the thresholds, thus proposing an edge-detection-based method for lawn weed identification. This method has contributed a unique technical approach to lawn weed identification in the development of weed detection. Gée and Denimal (2020) [4] constructed relevant indicators by using RGB images to achieve a spatial assessment of the impact of broadleaf weeds on wheat biomass. This result has provided new ideas for the research on evaluating the impact of weeds on crops based on remote-sensing images in the development of weed detection.

With the continuous enhancement of hardware computing power, deep learning has begun to play an important role in the field of weed detection.

At present, object detection has two detection methods: single-stage and two-stage. Single-stage methods such as YOLO and SSD [5] directly predict the category of the target [6]. The detection speed is fast and can process images and videos in real time, but the accuracy is relatively low, and the detection effect on small and dense targets is poor; two-stage methods such as Faster R-CNN [7] and Mask R-CNN [8] first generate candidate boxes and then classify and regress them. The detection accuracy is high and the effect on small and dense targets is better, but the speed is relatively slow, and the computing resource requirements are large [9].

Currently, more and more researchers are using deep-learning models such as YOLO and Transformer networks to detect and classify field weeds. For example, in 2020, Zhang et al. [10] used YOLOv3-tiny for agricultural weed detection. Eventually, the model obtained an average mean prediction value (mAP) of 72.5% and an intersection-over-union value (IoU) of 80.12%. Hu et al. (2021) [11] proposed a new method for weed detection in precision agricultural cultivation using YOLOv4. This method achieved a detection accuracy of 97%, a recall rate of 81%, and an F1 value of 89%. Wang et al. (2022) [12] proposed a weed detection algorithm named TIA-YOLOv5. This algorithm adopted techniques such as pixel-level synthetic data augmentation, adding Transformer encoder blocks, involution-based channel-feature fusion, and adaptive spatial-feature fusion. On the beet dataset, the algorithm achieved results where the F1 value of weeds reached 70.0%, the average precision (AP) reached 80.8%, and the average precision at an intersection over union of 0.5 (map@0.5) reached 90.0%. This has provided a new method for precision agricultural weed management in the development of weed detection. Gallo et al. (2023) [13] proposed an improved YOLOv7 weed detection algorithm. This algorithm generated numerical indicators of 56.6%, 62.1%, and 61.3% in terms of the average precision at an intersection over union of 0.5 (mAP@0.5), recall rate, and accuracy, respectively. Ding et al. (2024) [14] proposed the RVDR-YOLOv8 model. This model improves the average precision at an intersection over union of 0.5 (mAP50) by 1.7% and the average precision at an intersection over union between 0.5 and 0.95 (mAP50–95) by 1.1%. Its performance is superior to models such as Faster R-CNN, which helps to solve the problem of weed detection under resource-constrained conditions, thereby promoting the development of weed detection and providing support for farmland weed control.

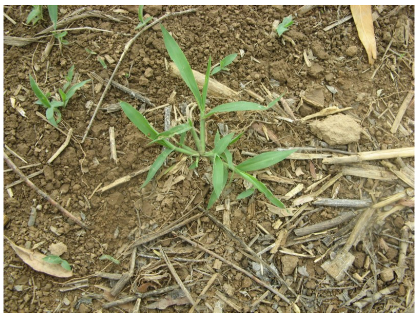
Despite significant advancements in weed detection from previous studies, there remains substantial potential for enhancing both accuracy and speed [15]. The YOLO algorithm, now advanced to YOLOv10, has demonstrated notable improvements in these aspects. This study specifically concentrates on optimizing the YOLOv8n framework to achieve further enhancements in detection accuracy.

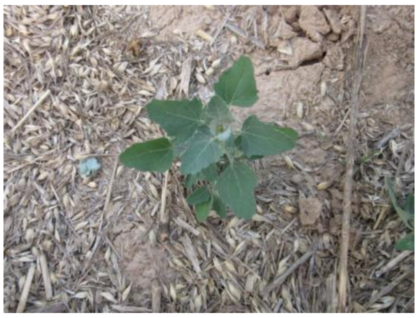
2. Materials and Methods

2.1. Experimental Data

2.1.1. Data Collection and Annotation

Firstly, we used a customized weed dataset, which includes public datasets downloaded from the Internet. For details, see the "corn weed datasets" directory in the following link: https://gitee.com/Monster7/weed-datase/tree/master (accessed on 23 March 2024). In the dataset, there is significant overlap among bluegrass images. Because of this overlap, the model may be misled into focusing on the wrong target contours, so the image data of bluegrass species need to be screened. In addition, we expanded the dataset and introduced a new weed target, Portulaca oleracea. The target data were sourced from the Nanzhuang Corn Planting Base in Qiaodong District, Zhangjiakou City, Hebei Province. They were collected using the camera of a Redmi K50 mobile phone, which has an image resolution of 3000 $\times$ 4000 pixels and is produced by Redmi Technology Co., Ltd. based in Beijing, China. To meet the input requirements of the algorithm, we processed the image pixels and adjusted the size to 480 $\times$ 640. Due to objective reasons such as time and environment, the number of collected target images is relatively small. Although the number of samples for this target is relatively small, due to its highly distinguishable features, we still fully committed to in-depth research and analysis of the existing samples to ensure reliable and effective research results. Eventually, a total of 4398 images of crops and weeds were collected. Our model focuses on detecting five types of weeds and corn seedlings: bluegrass, chenopodium album, cirsium setosum, corn, portulaca oleracea, and sedge. To annotate the weed images, rectangular regions were manually labeled using the LabelImg (The version number is 1.8.1) image annotation software [16]. The type of destination images are shown in Figure 1.


(**a**)


(**b**)

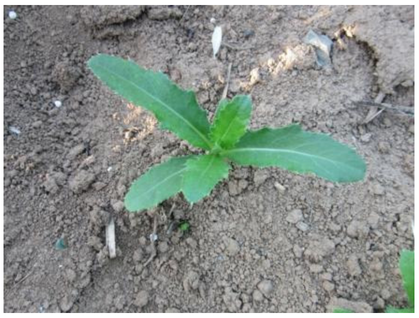

(**c**)

Figure 1. *Cont.*

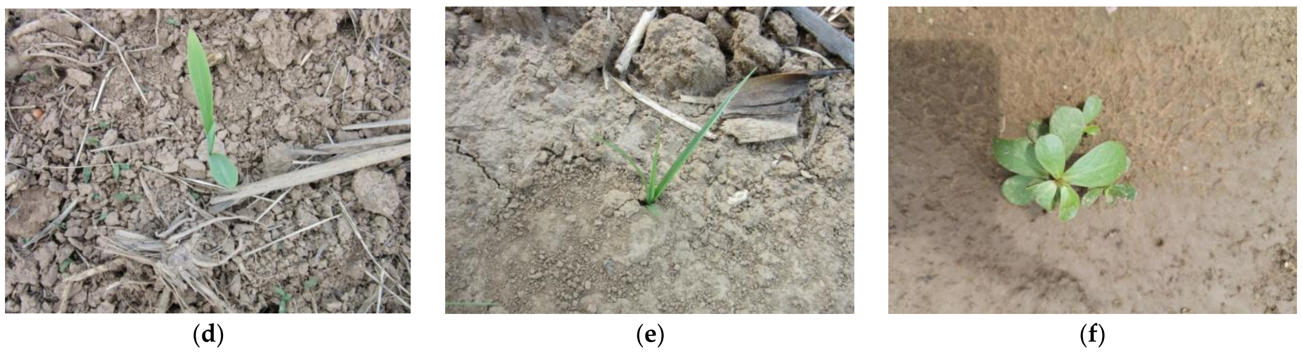


Figure 1. The type of destination in the dataset: (**a**) bluegrass; (**b**) chenopodium album; (**c**) cirsium setosum; (**d**) corn; (**e**) sedge; (**f**) portulaca oleracea.

2.1.2. Data Enhancement

Due to the limited dataset. The network might overly focus on irrelevant information within the images. There is a possibility of overfitting. To mitigate this, data enhancement techniques including random brightness modification, adding blur noise, cropping, and flipping were applied to the dataset. The augmented images are shown in Figure 2.

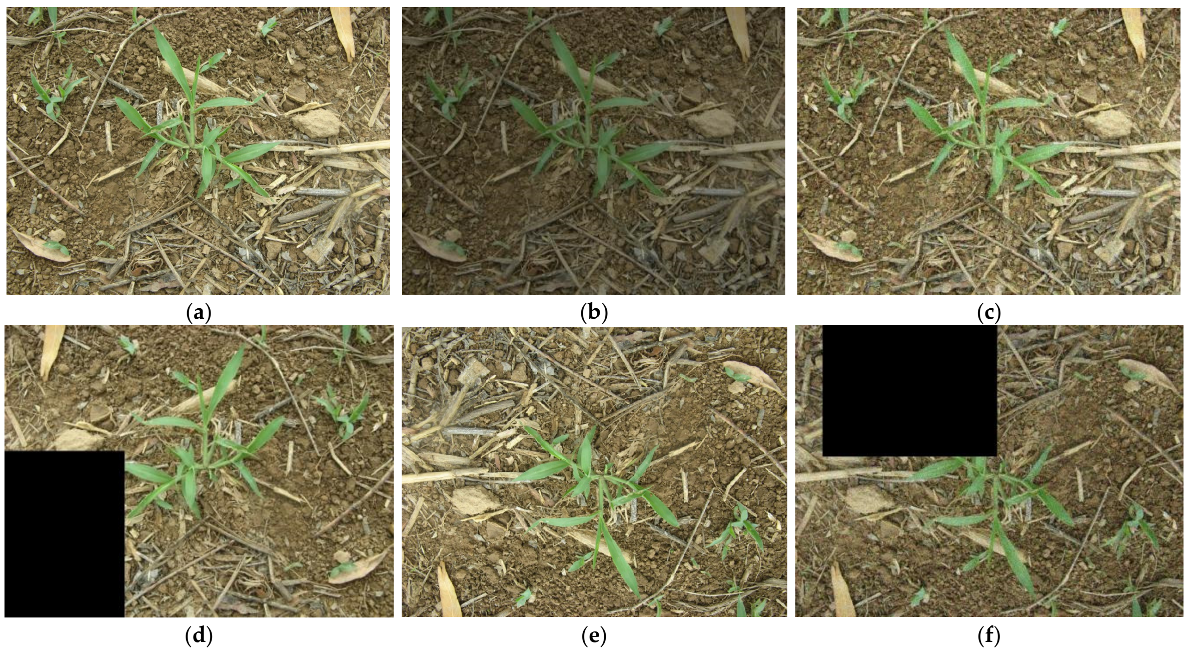


Figure 2. Original and after data augmentation image: (**a**) original image; (**b**) charge light; (**c**) Gaussian noise; (**d**) crop image; (**e**) flipping; (**f**) flipping + crop + Gaussian noise.

The dataset was ultimately expanded to include a total of 7966 images, and it is divided into a training set, a validation set, and a testing set according to the ratio of 7:2:1. This expansion was undertaken with the specific aim of enhancing the model's generalization performance, ensuring it can adapt effectively to a broader range of scenarios. By increasing the dataset's diversity, the robustness of the model is improved, making it more resilient to variations in the input data. Consequently, this strategy also helps to mitigate the risks associated with over-fitting, where the model might otherwise become too narrowly focused on the training data and perform poorly on new, unseen data. Since each image contains multiple targets or various types of targets, we counted the number of anchor boxes for all categories in the dataset. Table 1 shows the detailed categories of the enhanced dataset and the number of anchor boxes for each category, highlighting the diversity and scope of the expanded dataset.

Table 1. The type of weed and corresponding number of anchor boxes after data augmentation.

Category Code	Weed Category	Quantity
0	Bluegrass	2763
1	chenopodium album	1460
2	cirsium setosum	1498
3	Corn	2676
4	Sedge	2542
5	portulaca oleracea	126

2.2. YOLOv8 Algorithm Description

The backbone and neck form the core of YOLOv8, processing the image through Conv and C2f modules. The C2f module, derived from YOLOv7's C3 [17], enhances gradient branches while maintaining a lightweight structure. The SPPF module refines feature maps before passing them to the neck layer. Figure 3 illustrates the base algorithm structure.

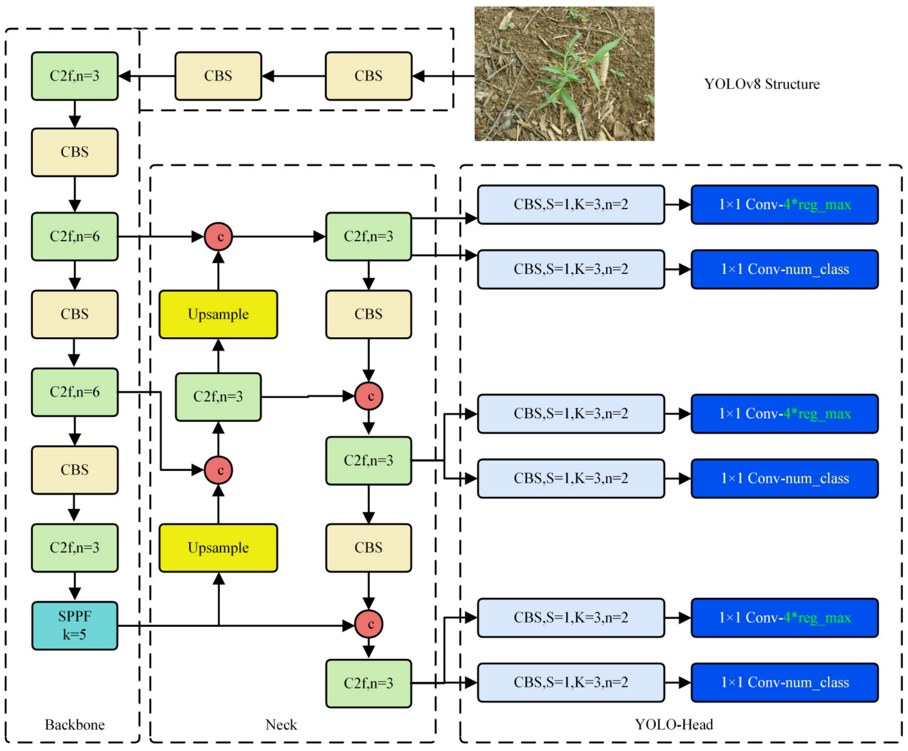


Figure 3. Structure of the source YOLOv8.

The neck module in YOLOv8 leverages FPN [18] and PAN [19] structures to enhance feature fusion. This design integrates high- and low-level features through upsampling and downsampling. This method enhances the network's capability to integrate features from different scales, boosting detection performance for objects of varying sizes.

The detection module of YOLOv8 separates the classification head from the detection head and processes the sample category loss and bounding box loss. The author uses binary cross-entropy loss to obtain the sample category loss. Then, the author uses CIoU to calculate the bounding box loss [20]. In order to balance the effects of the two loss calculation indicators on the detection evaluation, the author uses the task alignment allocator to calculate a more balanced prediction evaluation indicator. This indicator combines the classification loss with the bounding box loss. This indicator optimizes both classification and positioning, effectively reducing the impact of low-quality prediction boxes. The equation of the task alignment metric is given in detail in Equation (1):

$$t = \mathcal{S}^{\alpha} \times \mathcal{U}^{\beta} \tag{1}$$

The task alignment metric equation integrates the classification score (S) and the IoU value (U), with α and β as weighted hyperparameters [21]. The value of t dynamically guides the network towards high-quality anchor prediction. An IoU greater than 0.5 usually indicates a successful detection. The specific equation is provided in Equation (2):

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{2}$$

2.3. Improved YOLOv8 Algorithm

This study adjusts the model structure to minimize parameters and computational complexity. These modifications aim to optimize the model for environments with limited computational resources, achieving significant reductions in parameters and complexity while enhancing detection accuracy.

For instance, a more lightweight and efficient AKConv network replaces the standard convolutional layers stacked before each C2f module in the backbone network of YOLOv8 [22]. Using attention mechanisms is a common approach to improve model accuracy. Attention mechanisms have relatively few parameters and can offer high performance.

Additionally, the upsampling module in the neck module of the source network employs [23] an ultra-lightweight and efficient dynamic upsampler, DySample [24]. DySample features fewer parameters, FLOPs, GPU memory, and latency, while significantly improving accuracy and mean average precision (mAP).

Finally, surprisingly, by cleverly introducing the advanced attention mechanism, the model has become extremely effective in detecting targets, especially for small objects, which can be detected accurately, thereby greatly improving the detection accuracy. This study selects the outstanding LSKA-Attention and integrates it into the original YOLOv8 model. This module not only reduces computational complexity but also improves detection accuracy. Figure 4 illustrates the improved YOLOv8 algorithm.

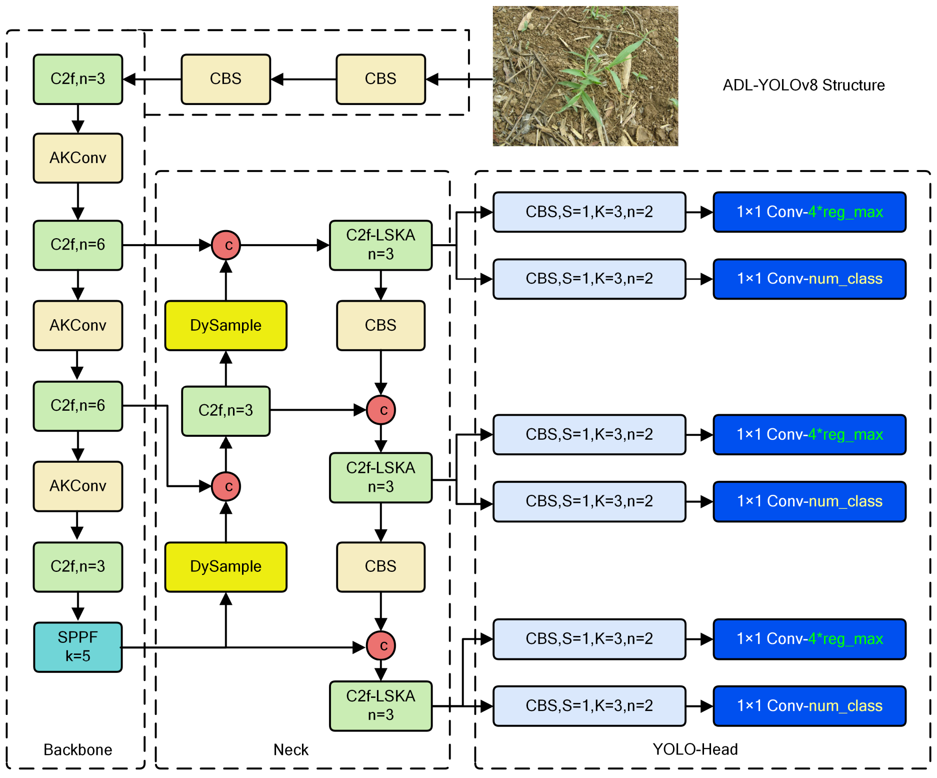


Figure 4. Improved structure of YOLOv8. In YOLO-Head, the asterisk (*) represents arithmetic multiplication to obtain the number of channels of the convolution kernel.

2.3.1. AKConv Model

In the backbone network of YOLOv8, there are five main components: the Stem Layer and Stage Layers 1 through 4. Each of the Stage Layers 1 through 4 contains a convolution module and several C2F modules, with the final layer requiring the output features to pass through an SPPF module. The Stem Layer is the initial part of the network. This reduces the dimensionality of the input image while retaining key information. The four Stage Layers form the core part of the backbone network, particularly Stage Layers 2, 3, and 4, as their output is used as input for the top-down multi-scale feature pyramid structure (FPN) in the neck network.

Each layer needs to use a traditional convolution kernel for downsampling before processing. However, standard convolution operations often rely on square kernels with fixed sampling positions, such as 1×1, 3×3, 5×5, and 7×7. The sampling positions of these regular kernels are immutable, and their sizes cannot dynamically change with the shape of the objects, leading to suboptimal feature extraction for some specific data samples, failing to encompass semantic information at every scale.

To effectively address the aforementioned issues, the standard convolution kernel before each stacked C2F module is improved to create special sampling shapes for convolution operations based on prior knowledge. This is achieved through AKConv, a convolution kernel that dynamically adapts to changes in object shapes via offsets. AKConv belongs to a class of deformable convolution networks that can handle irregularly shaped convolution kernels, such as sizes 5, 7, and 13. AKConv first generates a regular sampling grid, then creates irregular grids for the remaining sampling points, and finally combines them into a complete sampling grid.

Since standard convolution operations often rely on square kernels with fixed sampling positions centered at (0,0), irregular convolutions lack a center point for many sizes. To adapt to the size of the convolutions used, AKConv generates the initial sampling coordinates with the top-left corner (0,0) as the sampling origin, representing the uniform distribution of sampling points without additional offsets. The generated initial sampling coordinates are shown in Figure 5.

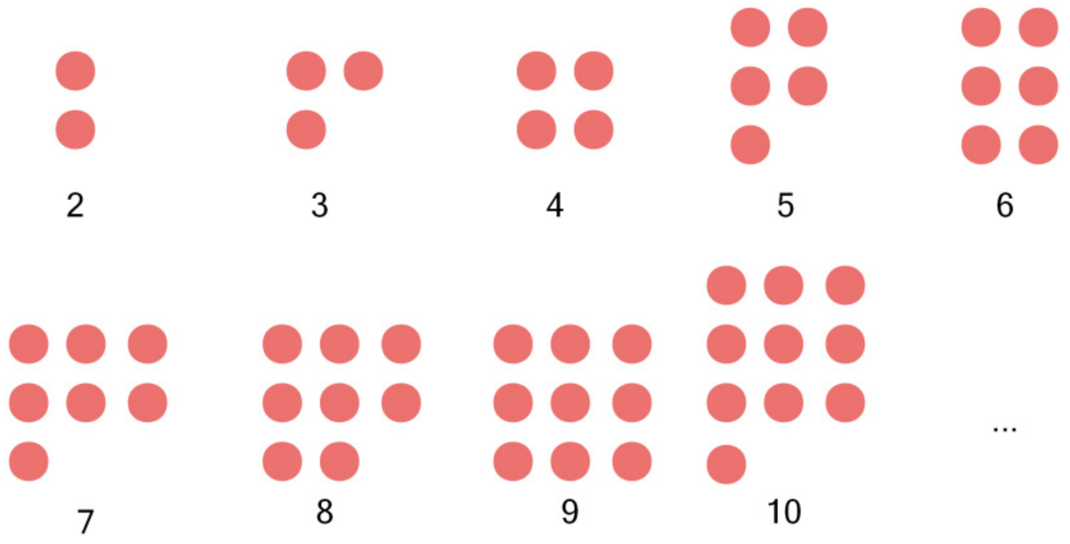


Figure 5. Initial sampling coordinates generated by the algorithm for arbitrary convolution kernel sizes. It provides initial sampling shapes for irregular convolution kernel sizes.

After initializing the sampling grid, AKConv first obtains the offsets through convolution operations. The number of channels of the offset divided by 2 and rounded down to the nearest integer is used as the input to generate the basic offset grid p_n. Then, the base offset points p_n, initial grid points, and offsets are added together to obtain the final sampling coordinates $\overline{p}$. This integration of regular sampling positions, preset base offsets, and dynamically learned offsets allows the sampling to adapt to the changes in the input features. Finally, bilinear interpolation resampling is used to extract features from the corresponding positions, thereby achieving adaptive resampling of the input. The feature map is then reshaped for subsequent feature processing. The principles and specific structure of AKConv convolution operations are shown in Figure 6.

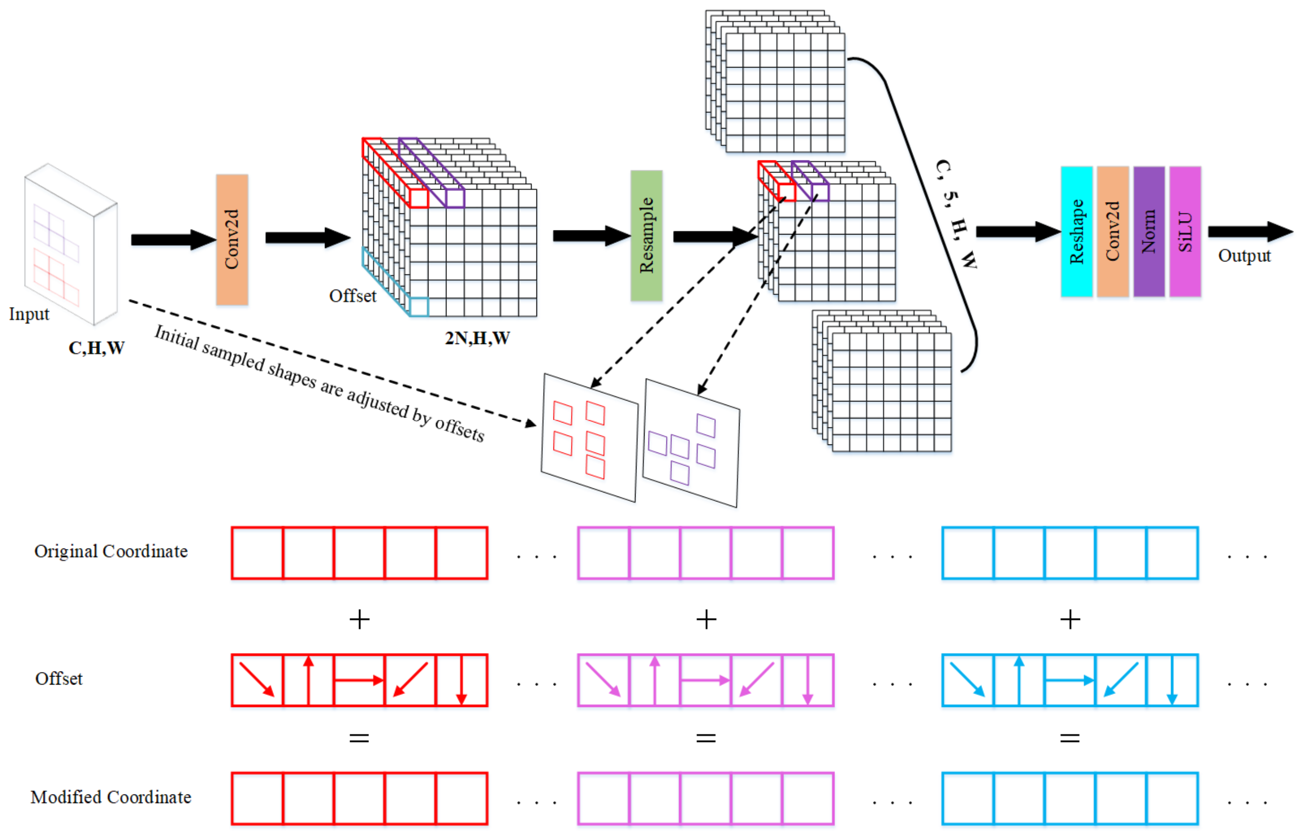


Figure 6. The structure of AKConv.

AKConv assigns initial sampling coordinates to convolution operations of any size and adjusts the sample shape with learnable offsets. Compared to the original sample shape, resampling changes the shape of each position [25]. Therefore, AKConv effectively performs the feature extraction process for irregular convolutions. By completing the feature extraction process through irregular convolutions, AKConv can flexibly adjust sample shapes based on offsets, providing more exploratory options for convolution sampling shapes. Unlike standard convolutions and deformable convolutions, which are constrained by the concept of regular convolution kernels, AKConv is more suitable for tasks that require detection of specific shapes (such as elongated tubular structures) or tasks that require convolution operations of arbitrary shapes and sizes.

2.3.2. DySample Upsampling Module in FPN

The model's neck module utilizes an enhanced PAN-FPN structure, improving upon the traditional FPN, which traditionally passes deep semantic information top-down. In YOLOv8, B3-P3 and B4-P4 are concatenated. This operation is performed to strengthen semantic features. However, it can lead to some loss of localization information [26]. To mitigate this, a bottom-up PAN structure is added, enhancing localization through P4-N4 and P5-N5 fusion. This complementary approach ensures better feature learning. The specific architecture is shown in Figure 7.

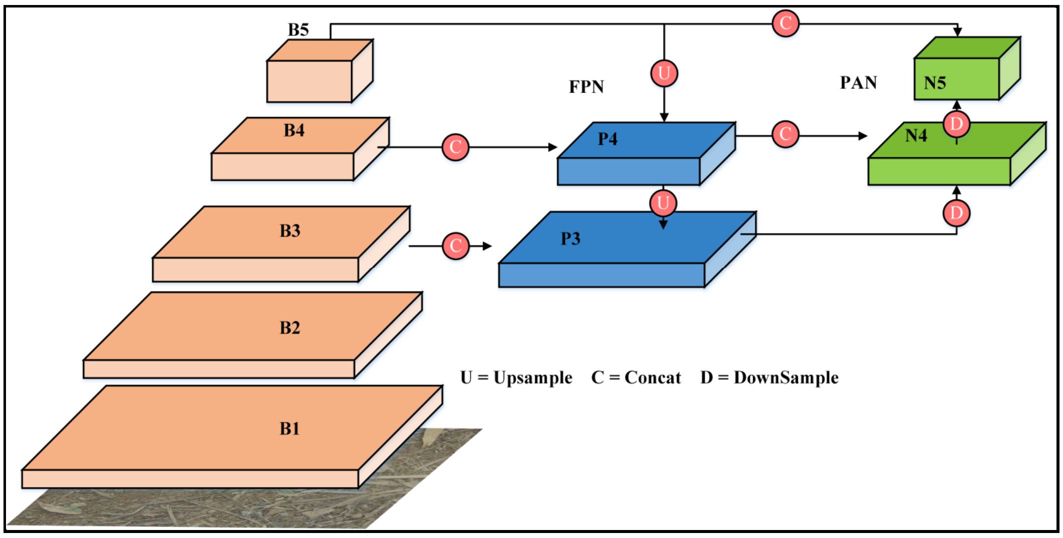


Figure 7. YOLOv8 PAN-FPN.

But the PAN-FPN structure still has limitations. One key issue is its potential inefficiency in handling large-scale feature maps, which may lead to the omission of critical details and a reduction in detection quality. Furthermore, after upsampling and downsampling, some original information in the feature maps is lost. Therefore, the PAN-FPN structure still has potential for enhancement.

The backbone network uses three improved AKConv output convolutions to obtain three feature maps of different scales, which serve as the input for the FPN in the neck module. However, the FPN structure in YOLOv8 uses the traditional nearest-neighbor interpolation algorithm for upsampling. This algorithm can lead to jagged edges in image features, unsmooth details, and semantic consistency issues due to noise introduction. In contrast, dynamic upsamplers address these issues by generating content-aware upsampling kernels through dynamic convolution, but this approach introduces significant computational overhead.

To more effectively resolve the above problems of traditional upsampling, this study reconstructs the upsampling module in the feature merging part of YOLOv8 according to the idea of Learning to Upsample by Learning to Sample (DySample). Dynamic upsampling is introduced into the neck network. Unlike other dynamic upsamplers like CARAFE, FADE, and SAPA, which generate dynamic kernels and guide the upsampling process with higher resolution structures, DySample is ultra-lightweight and efficient. It does not require any additional CUDA packages apart from PyTorch, and it has significantly lower inference latency, memory usage, FLOPs, and parameter count.

DySample's core idea is point sampling. Specifically, it assumes that input features are interpolated into a continuous space through bilinear interpolation and generates content-aware sampling points to resample this continuous mapping. Finally, DySample optimizes the upsampling performance by controlling the initial sampling positions, adjusting the offset movement range, and dividing the upsampling process into several independent groups. Grid sampling is illustrated in Figure 8.

In the aforementioned sampling generator, there are two primary steps: linear transformation and pixel rearrangement. First, the input feature map X [27] is processed through a linear layer with input and output channel dimensions of C and $2s^2$, respectively [28]. This is followed by pixel transformation, which produces an offset map O of size $2 \times sH \times sW$. Subsequently, grid sampling is performed, where the spatial sampling of the input image X is conducted based on the computed offsets to generate new coordinates. Finally, bilinear interpolation is applied to the image based on the calculated grid coordinates, resulting in the final feature map.

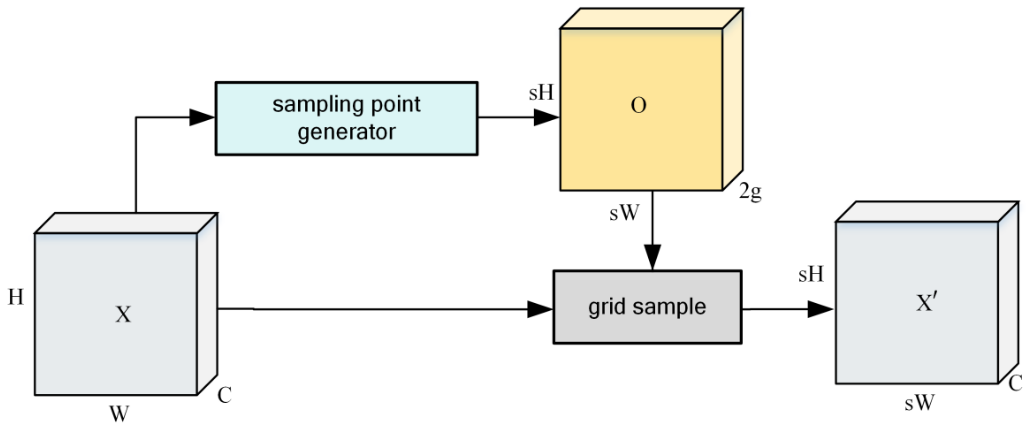


Figure 8. Sampling-based dynamic upsampling.

However, the initial offset positions shared by s^2 upsampling points in the sampling set ignore positional relationships, and the unconstrained roaming range of the offsets can lead to disordered sampling points. To address these two issues, the algorithm optimizes by grouping the samples and adjusting the movement range of the offsets.

The initial sampling positions are shown in Figure 9a. Since the initial sampling positions for each upsampling point are the same, the positional relationships between adjacent points are ignored, leading to uneven sampling position distribution. Therefore, "bilinear initialization" is introduced, as shown in Figure 9b. This changes the initial positions so that when the offset is zero, bilinear interpolation results can be obtained. To facilitate understanding of the bilinear interpolation in the below figure, four colors are used here to distinguish the sampling range of each sampling point.

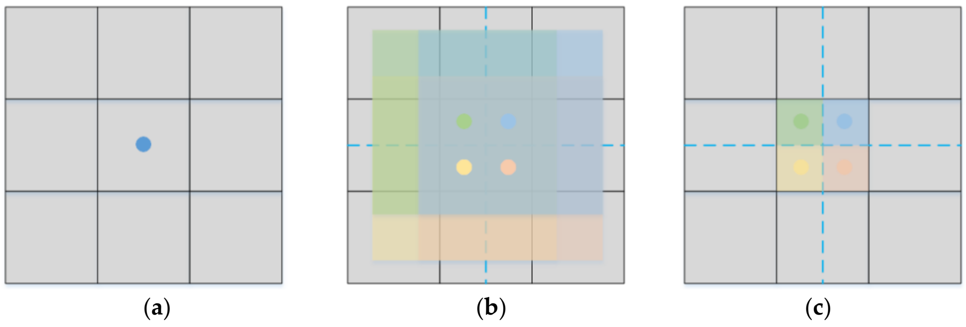


Figure 9. (**a**) In the case closest to initialization, all 4 offsets share the same initial position, ignoring positional relationships. (**b**) In bilinear initialization, the initial positions are separated to achieve uniform distribution. However, without offset modulation, the offset ranges typically overlap. (**c**) The offset ranges are constrained to reduce overlapping.

This improvement enhances the model's performance. However, the normalization of the offset range might cause adjacent sampling positions to overlap, affecting boundary prediction and generating output artifacts. To mitigate this, we multiply the offset by 0.25, which meets the theoretical marginal condition between overlapping and non-overlapping [29], as shown in Figure 9c. This factor, known as the "static range factor," constrains the walking range of sampling positions locally, reducing overlap and further improving performance. We rewrite Equation (3) as follows:

$$\mathcal{O} = 0.25 linear(\mathcal{X}) \quad (3)$$

The implementation principle of the improved static sampling set generator is illustrated in Figure 10.

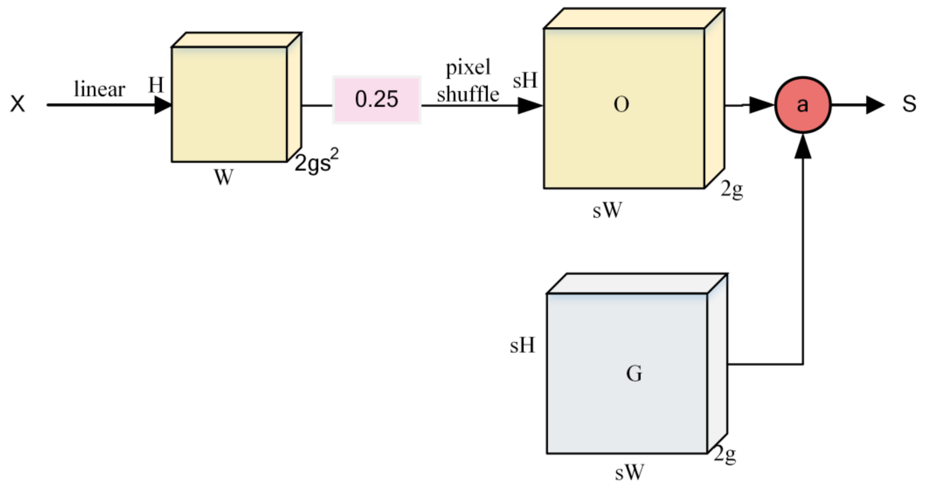


Figure 10. Static sampling set generator.

To increase the flexibility of the offsets, a point-to-point dynamic range factor is generated through linear mapping of the input features combined with a sigmoid function. This makes the offset range more adaptable to different situations, further fine-tuning performance. The implementation principle of the improved dynamic sampling set generator is illustrated in Figure 11. In this setup, the range factor is first generated and then used to modulate the offsets [30]. The σ represents the sigmoid function. The final results for both types of sampling sets are the aggregate of the generated offsets and the initial grid positions. The feature map is then divided into multiple groups by channels, with each group dividing the same set of offsets. This approach improves model performance, especially when the number of groups is set to 4, showing a significant enhancement in performance.

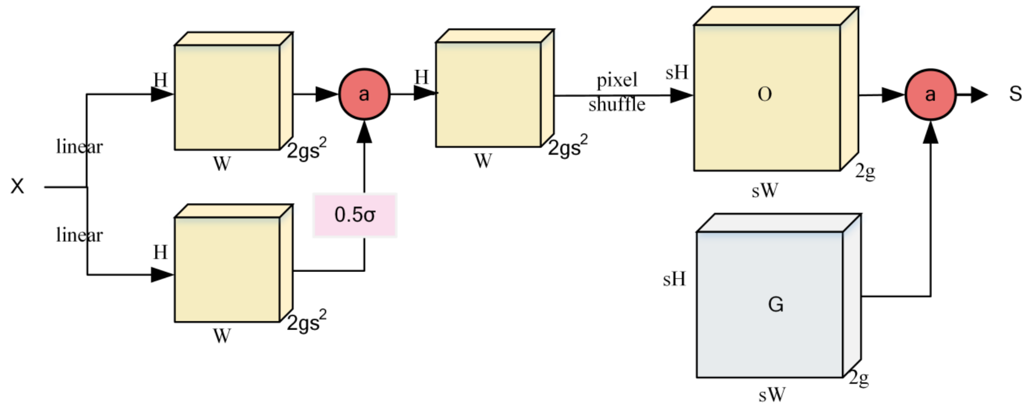


Figure 11. Dynamic sampling set generator.

Furthermore, according to the above discussion, the offset generation method can be divided into two approaches: first linear mapping and then pixel rearrangement (LP) and first pixel rearrangement and then linear mapping (PL).

The upsampling module in YOLOv8's feature fusion component introduces DySample. It not only significantly reduces inference latency, memory usage, FLOPs, and parameter count but also enhances the source network's detection accuracy. This algorithm not only addresses the issues of jagged edges and uneven details that may arise from traditional nearest-neighbor interpolation upsampling algorithms through continuous and context-aware point sampling but also prevents the semantic consistency issues that may result from noise introduced by conventional upsampling algorithms. Consequently, the PAN-FPN module's capability to fuse multi-scale features has been optimized, providing more accurate feature maps for subsequent model detection modules and enhancing the model's detection capabilities.

2.3.3. LSKA-Attention

In the head detection network of YOLOv8, the three feature maps that need to be detected, sized 80 × 80 × 256, 40 × 40 × 512, and 20 × 20 × 1024, respectively, are processed through two (3,1) CBS convolution modules. These are then followed by a (1,1) convolution module with a channel number of 4*reg_max to obtain the anchor box loss. Additionally, they pass through two (3,1) CBS convolution modules followed by a (1,1) convolution module with a channel number of num_class to obtain the classification loss. These three detection feature maps are obtained after feature extraction and enhancement through a C2f convolution module following feature fusion in the PAN-FPN module.

Although the inclusion of the AKConv and DySample algorithms has already improved the model's feature extraction capability, further enhancements are needed to ensure that the detection network selectively focuses its limited resources on critical information during feature processing. Therefore, a C2f-LSKA-Attention module is proposed, which integrates a large kernel separable convolution attention mechanism (LSKA-Attention) into the C2f module. This allows the model to dynamically allocate weights to different parts of the input [31], focusing on important information in the feature map while ignoring or de-emphasizing unimportant information during subsequent detection.

The core idea of LSKA-Attention is similar to the Vision Transformer (VIT), employing the currently popular Transformer self-attention mechanism module. However, LSKA possesses channel adaptability that standard CNN and Transformer self-attention modules do not have. LSKA constructs the attention module by introducing a large kernel size (i.e., large kernels), allowing each attention head to cover a larger image area and capture a broader range of contextual information [32].

In addition, the separation of large kernels is proposed, which solves the problem of significant computational and memory overhead when processing high-resolution images. Then, we introduce the principles of LSKA step by step through a simple example.

Firstly, the authors designed a simple large kernel convolution, as shown in Figure 12a. Its output expression is given by the following:

$$\mathrm{Z^C} = \sum_{\mathrm{H,W}} \mathrm{W^C_{k\times k}} * \mathrm{F^C} \tag{4}$$

$$\mathrm{A^C} = \mathrm{W_{1\times 1}} * \mathrm{Z^C} \tag{5}$$

$$\overline{\mathrm{F}}^{\mathrm{C}} = \mathrm{A^C} \otimes \mathrm{F^C} \tag{6}$$

Here, $*$ denotes convolution, and $\otimes$ signifies the Hadamard product [33]. As the kernel size grows, the computational cost of depth-wise convolution within the large kernel module increases quadratically.

To reduce the high computational cost [34] of large kernel depth-wise convolutions, the authors decomposed them into cascaded one-dimensional horizontal and vertical depth-wise convolutions combined with a 1 × 1 convolution (Figure 12b). However, they observed that increasing the kernel size in LKA-trivial leads to a rise in computational complexity proportional to k^2.

To address the quadratic increase in computational cost due to large kernel sizes, the authors drew on the original LKA design from the VAN network, which incorporates standard depth-wise convolution, dilated depth-wise convolution, and 1 × 1 convolution (Figure 12c). This large kernel decomposition mitigates the computational burden associated with depth-wise convolution. The LKA output is obtained through this approach:

$$\overline{\mathrm{Z}}^{\mathrm{C}} = \sum_{\mathrm{H,W}} \mathrm{W^C_{(2d-1)\times(2d-1)}} * \mathrm{F^C} \tag{7}$$

$$\mathrm{Z^C} = \sum_{\mathrm{H,W}} \mathrm{W^C_{\lfloor \frac{k}{d} \rfloor \times \lfloor \frac{k}{d} \rfloor}} * \overline{\mathrm{Z}}^{\mathrm{C}} \tag{8}$$

$$A^C = W_{1\times1} * Z^C \quad (9)$$

$$\overline{F}^C = A^C \otimes F^C \quad (10)$$

where d is the dilation rate, and $\overline{Z}^C$ denotes the output from a depth-wise convolution with a kernel size of $(2d-1) \times (2d-1)$, capturing local spatial details [35]. This helps offset the grid effect of the subsequent depth-wise dilated convolution, which uses a kernel size of $\left\lfloor \frac{k}{d} \right\rfloor \times \left\lfloor \frac{k}{d} \right\rfloor$. The dilated depth-wise convolution is meant to seize overall space-related data from the output $\overline{Z}^C$. However, when the kernel size increases to more than 23 × 23, it still results in very high computational complexity and memory usage.

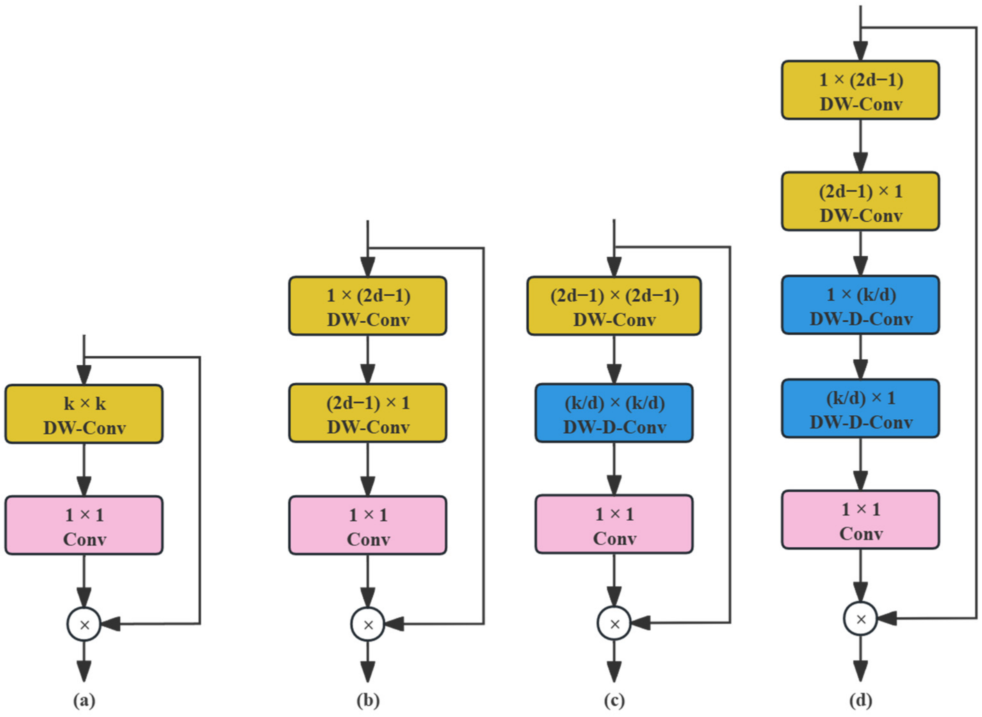


Figure 12. (**a**–**d**) Comparison of different designs for large kernel attention modules.

Finally, the authors applied the method shown in Figure 12b, where the two-dimensional weight kernels from depth-wise and dilated depth-wise convolutions are divided into two sequential one-dimensional separable kernels. This results in an equivalent enhanced LKA structure. This yields the LSKA (Large Kernel Separable Attention) module, as shown in Figure 12d. Its output expression is given by the following:

$$\overline{Z}^C = \sum_{H,W} W^C_{(2d-1)\times1} * \left(\sum_{H,W} W^C_{1\times(2d-1)} * F^C \right) \quad (11)$$

$$Z^C = \sum_{H,W} W^C_{\lfloor \frac{k}{d} \rfloor \times 1} * \left(\sum_{H,W} W^C_{1\times\lfloor \frac{k}{d} \rfloor} * \overline{Z}^C \right) \quad (12)$$

$$A^C = W_{1\times1} * Z^C \quad (13)$$

$$\overline{F}^C = A^C \otimes F^C \quad (14)$$

This module adopts a cascaded horizontal and vertical kernel design, which not only retains the model's ability to capture long-range dependencies but also further reduces memory and computational complexity.

The aforementioned improvements enable the model to concentrate its limited resources on critical information. Consequently, this approach provides more accurate feature maps for the detection module, thereby enhancing the capabilities of the detection network.

2.4. Experimental Parameters and Evaluation Metrics

2.4.1. Experimental Parameters

Experimental basic parameters are categorized into hardware parameters and algorithm parameters. To ensure the comparability of experimental results, different algorithm innovations must be executed under the same hardware environment to eliminate the effects of hardware variations [36]. Throughout the entire training process of all experiments, consistent hyperparameters must be applied to each algorithm.

The hardware equipment of the experiment includes CPU, GPU, and RAM. The graphics card used is RTX 3080, which has a video memory capacity of 10 GB. In order to provide a temporary data storage area when the algorithm is running, the experiment also uses a 40 GB memory capacity. The software part uses the Ubuntu 20.04 operating system equipped with the PyTorch 1.11.0 framework.

The implementation language of the algorithm is Python3.8. The algorithm part uses the YOLOv8n model as the baseline network, which has many customized hyperparameters. The learning rate of this experiment is 0.01, and the model input size is 640 $\times$ 640. For the efficient progress of the experiment, it is necessary not only to make full use of the video memory to increase the training speed of the model but also to ensure that the total memory occupation of each batch of processed pictures does not exceed the size of the video memory to prevent the model training from crashing. Therefore, through the initial tests, it is determined that the training batch size, epoch, and workers are 40, 300, and 24, respectively. α and β in Equation (1) are set to 0.5 and 7.5, respectively [37].

2.4.2. Model Evaluation Indicators

This experiment adopts three detection indicators that are widely used in classification, detection, and recognition tasks. First, precision refers to the proportion of samples predicted as positive examples that are actually positive examples [38]. Its definition equation is as follows (15):

$$P = \frac{TP}{TP + FP} \tag{15}$$

A higher precision means that a higher proportion of the positive examples predicted by the model are indeed positive examples [39], that is, the prediction results of the model are more accurate. Then, recall refers to the proportion of samples that are correctly predicted as positive examples among samples that are actually positive examples. Its definition equation is as follows (16):

$$R = \frac{TP}{TP + FN} \tag{16}$$

Recall measures the ability of the model to detect all positive examples. A higher recall means that the model can better find all samples that are actually positive examples. The last indicator is mAP, which is a comprehensive evaluation indicator that evaluates the overall accuracy based on the overlap of predictions with the actual box. Its definition equation is as follows (17):

$$mAP = \frac{\sum_{i=1}^{Q} AP_i}{Q} \times 100\% \tag{17}$$

The average precision of each category is represented as AP_i. Its definition equation is as follows (18):

$$AP_i = \frac{\frac{TP}{TP+FP}}{N} \tag{18}$$

This study also takes into account computational cost (GFLOPS) and model size, where lower values mean better performance and efficiency.

3. Results

3.1. Ablation Experiment

The ablation experiments were conducted to assess the effectiveness of the improved YOLOv8 algorithm in weed detection tasks by progressively incorporating different enhancement modules to evaluate their impact on model performance. Each enhancement demonstrated performance improvements, particularly in recall and mAP metrics. The introduction of AKConv effectively reduced model complexity while maintaining strong performance. The replacement with DySample enhanced precision and recall, indicating its advantages in the upsampling process. The integration of the LSKA method further optimized the model's efficiency and performance [40]. Detailed experimental results and performance improvements are presented in Table 2. As shown in the following table, we use $\surd$ to indicate the introduction of a certain module in the original model.

Table 2. Results of the ablation experiment.

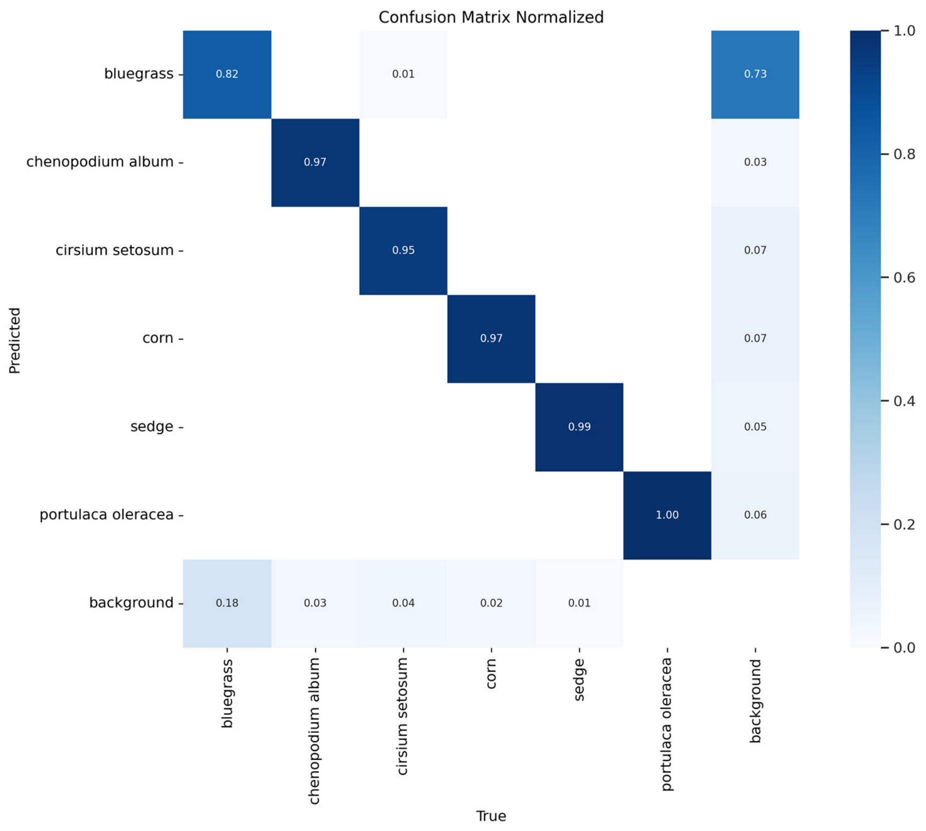
AKConv	DySample	LSKA-Attention	Precision (%)	Recall (%)	mAP50 (%)	mAP50:95 (%)	Model Size (MB)	GFLOPS
			89.93	86.16	91.64	72.53	5.96	8.2
√			88.34	89.01	93.43	73.12	5.51	7.8
	√		91.37	87.79	94.42	73.20	5.98	8.2
		√	88.60	88.77	93.52	74.57	5.37	7.6
√	√		89.49	88.56	93.72	75.20	5.52	5.4
√		√	90.81	86.44	92.41	70.38	10.16	5.2
√	√	√	92.13	88.61	94.71	74.43	5.02	7.3

Overall, the enhanced YOLOv8 model combines AKConv, DySample, and LSKA. This enhanced model outperforms the original YOLOv8 in detection accuracy. It also outperforms the original YOLOv8 in computational efficiency. In addition, it has fewer parameters than the original YOLOv8. This improved model showed a 2.2% increase in precision, a 2.45% boost in recall, a 3.07% rise in mAP@0.5, and a 1.9% gain in mAP@0.95. The model size was reduced by 15.77%, and computational complexity was decreased by 10.98%.

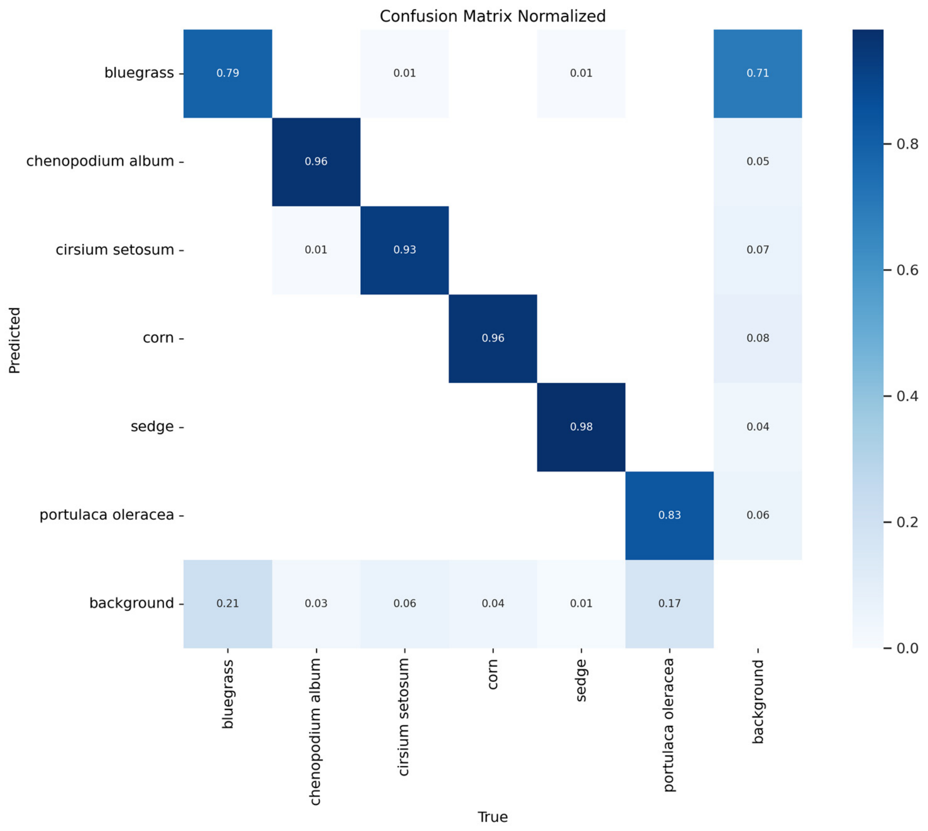
3.2. Confusion Matrix

Deep-learning models are often viewed as complex and somewhat mysterious black-box systems because their inner workings and decision-making processes are not always easy to understand or explain. Despite their strong performance across various tasks, their decision-making and reasoning processes are difficult to interpret. Understanding the explainability of these complex and powerful deep-learning models is of extreme essential significance in diverse significant areas like medical diagnosis, where accurate and explainable results can mean the difference between life and death, and autonomous driving, where safety and reliability depend on understanding the decisions made by the model.

In this experiment, both the enhanced and original YOLOv8 models were validated by examining their confusion matrices, as shown in Figure 13, to gain insights into their performance.



(**a**)



(**b**)

Figure 13. (**a**) The improved YOLOv8 model. (**b**) The original YOLOv8 model.

The confusion matrix [41] clearly shows that both models have cases where the actual target cannot be detected (false negatives) and also incorrectly identified negative cases as positives (false positives). Detailed analysis reveals that the original YOLOv8 model performs poorly in recognizing portulaca oleracea (purslane) and bluegrass, with accuracies of only 79% and 83%, respectively. In contrast, the proposed improved YOLOv8 model significantly enhances the recognition accuracy of portulaca oleracea and bluegrass by 3% and 17%, respectively. Additionally, other categories, which already had high accuracy in the original network, saw improvements as well: chenopodium album, corn, and sedge each improved by 1%, and cirsium setosum improved by 2%.

3.3. Comparison of Other Models

To further assess the practical applicability and usefulness of the proposed algorithm, this study undertook a comprehensive comparison between the ADL-YOLOv8 model and widely used object detection models including SSD, Faster R-CNN, YOLOv5n, YOLOv5s, YOLOv7-tiny, YOLOv7, YOLOv9t [42], and YOLOv10n [43]. The experiment, which was conducted under the same dataset and parameters and carried on for 300 iterations, clearly highlighted the outstanding effectiveness and superiority of ADL-YOLOv8. Table 3 presents the comparison of precision, recall, mAP@50, mAP@50:95, model size [44], and GFLOPs for each model.

Table 3. Comparison of different algorithms.

Model	Precision (%)	Recall (%)	mAP50 (%)	mAP50:95 (%)	GFLOPS	Model Size (MB)
SSD	41.26	48.31	61.69	37.67	68.2	105
Faster R-CNN	58.00	62.31	81.38	52.64	78.8	628.7
YOLO v5n	89.03	85.85	92.40	73.26	7.1	5.04
YOLO v5s	88.65	89.54	94.90	75.40	23.8	17.6
YOLO v7_tiny	86.50	83.20	86.10	63.90	13.2	12.3
YOLO v7	85.40	85.20	89.40	68.20	105.2	74.8
YOLO v9t	91.47	87.51	93.72	75.88	7.6	4.43
YOLO v10n	89.74	87.32	93.09	74.35	8.2	5.5
ADL-YOLOv8	92.13	88.61	94.71	74.43	7.3	5.02

Although YOLOv5s showed 0.19%, 0.97%, and 0.93% higher mAP@50, mAP@50:95, and recall rates, respectively, compared to the proposed algorithm, its precision, computational load, and model size were 3.48% lower, 226.0% larger, and 250.6% larger, respectively. Therefore, the results of the comparison clearly demonstrate that the improved ADL-YOLOv8 model exhibits the highest level of overall performance in terms of detection accuracy, recall rate, mAP@50, and mAP@50:95 when contrasted with other networks. Excluding YOLOv9t, which is rather closely comparable, and the non-lightweight YOLOv5s, it can be observed that the mAP@50 of the ADL-YOLOv8 model is, respectively, 33.02%, 13.33%, 2.31%, 8.61%, 5.31%, and 1.62% superior to that of the other six models. At the same time, the mAP@50:95 of the ADL-YOLOv8 model is, respectively, 36.76%, 21.79%, 1.17%, 10.53%, 6.23%, and 0.08% greater than that of the other six models.

The improved model presents a compact form. With a size of just 5.02 MB, it is significantly 0.94 MB smaller than YOLOv8n. As a result, it stands as one of the most diminutive models available, occupying the position that is second only to YOLOv9t. The ADL-YOLOv8 model also demands fewer computational resources, with GFLOPs at 7.3, which is 0.9 lower than the already efficient YOLOv8n. The ADL-YOLOv8 model has nearly 8.43 times higher GFLOPs than SSD, 9.79 times higher than Faster R-CNN, 2.26 times higher than YOLOv5s, and nearly 13.41 times higher computational efficiency than YOLOv7. Considering the need for real-time processing and model lightweighting while ensuring detection accuracy, the following three aspects are analyzed based on Table 3.

- Real-Time Performance

Although the GFLOPs of the proposed algorithm are not the lowest among similar models, reaching 7.3 GFLOPs, it is still significantly lower than most high-load models and remains closely aligned with designs that have extremely high computational efficiency, such as YOLO v9t. This allows the algorithm to ensure real-time performance while demonstrating a thoughtful balance between speed and efficiency in the algorithm design, making it an optimal choice for real-time application scenarios.

- Lightweight Design

Despite the proposed algorithm model size being 5.02 MB, slightly larger than YOLO v9t's 4.43 MB, this value still highlights significant achievements in lightweight design in the overall comparison. Specifically, when assessing the model's portability and resource efficiency, the proposed algorithm not only surpasses large models such as Faster R-CNN (628.7 MB) and YOLO v7 (74.8 MB) but also shows minimal differences compared to leading lightweight designs like YOLO v5n (5.04 MB) and YOLO v10n (5.5 MB). Therefore, the proposed algorithm is an ideal choice for edge computing and mobile device deployment, fully reflecting the delicate balance in algorithm design between accuracy and resource conservation.

- Accuracy

Among all the listed algorithms, the proposed algorithm exhibits outstanding performance in precision (P), recall (R), and mAP@0.5% and mAP@0.5:0.95% metrics, reaching 92.13%, 88.61%, 94.71%, and 74.43%, respectively. These values are significantly higher than other algorithms, such as the latest YOLO v9t algorithm (precision: 91.47%; recall: 87.51%; mAP@0.5%: 93.72%; mAP@0.5:0.95%: 75.88%) and YOLO v10n algorithm (precision: 89.74%; recall: 87.32%; mAP@0.5%: 93.09%; mAP@0.5:0.95%: 74.35%), demonstrating the superior detection accuracy of the proposed algorithm.

The proposed algorithm achieves high-precision detection results while maintaining low computational load (7.3 GFLOPs) and small model size (5.02 MB). This makes it competitive with YOLO v9t and YOLO v10n in terms of real-time performance and lightweight design, while also significantly surpassing all comparison objects in detection accuracy. Therefore, for the field of weed detection, the proposed algorithm is an ideal solution, capable of providing excellent detection performance without compromising on real-time and lightweight requirements.

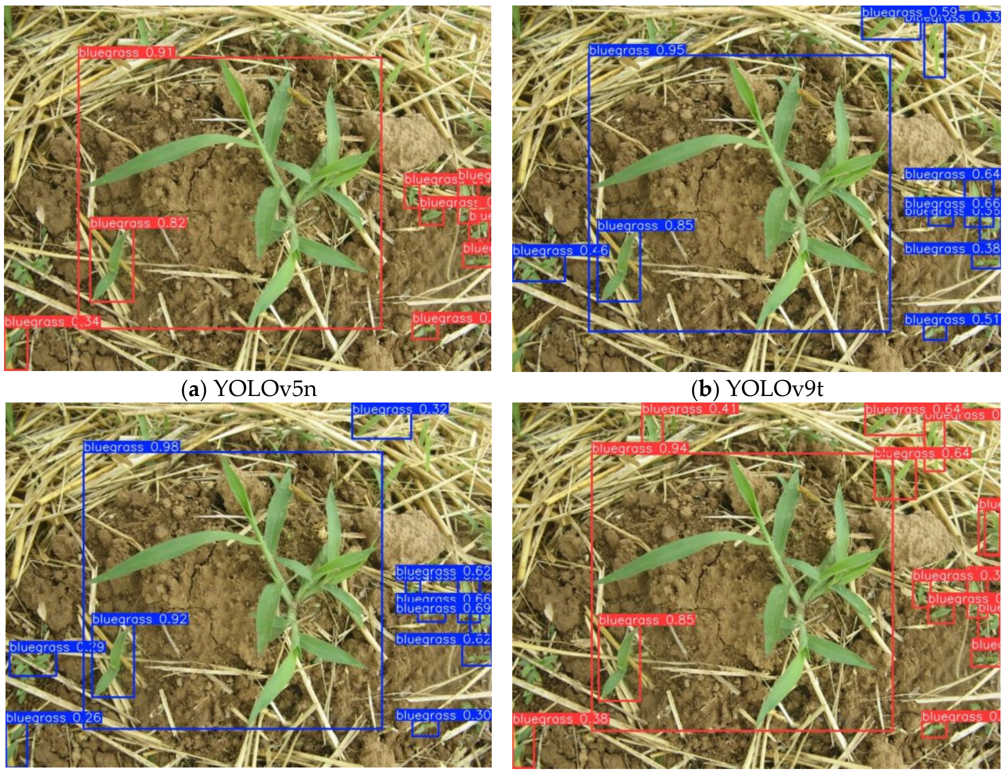
In summary, the ADL-YOLOv8 model that has been developed in this study offers extremely high accuracy in detecting weeds while keeping the additional parameters to an absolute minimum. As a result, it effectively improves the inference speed. The optimized YOLOv8 model also features a reduced memory footprint and lower computational demands, making it well-suited for deployment on embedded devices. This can facilitate the accelerated implementation of smart weeding equipment and precise pesticide application by drones in agriculture.

3.4. Different Models' Detection Visualization Results Analysis

The evaluation metrics of the Faster R-CNN, SSD, YOLO v7_tiny and YOLO v7 algorithms are far lower than those of the original YOLOv8. Moreover, the model size and GFLOPs of YOLOv5s are relatively large and fail to meet the basic requirements of this experiment.

To evaluate the practical effectiveness of the ADK-YOLOv8 model in weed detection, we utilized pre-trained ADK-YOLOv8, YOLOv5n, YOLOv9t, and YOLOv10n models to detect various types of weeds. This validation primarily tested the accuracy of these four models in detecting small and occluded targets. In the visualization results, blue and red bounding boxes indicate bluegrass, pink and light blue bounding boxes indicate chenopodium album, orange and white bounding boxes indicate cirsium setosum, and orange-yellow bounding boxes indicate corn, while yellow and dark blue bounding boxes indicate sedge. Due to the simplicity of the features of corn, sedge, and portulaca oleracea, the differences in performance among the four models in detecting these weeds were not significant, and thus, they were excluded from the visualization validation.

As shown in Figure 14, the results of the YOLOv5n, YOLOv9t, YOLOv10n, and ADK-YOLOv8 networks in detecting a large number of small targets are displayed. In images with complex backgrounds and multiple small targets, YOLOv5n, YOLOv9t, and YOLOv10n demonstrate notable deficiencies in extracting small target features. These models all exhibit missed detections, identifying 9, 10, and 11 bluegrass targets, respectively. However, the ADK-YOLOv8 model exhibits significant superiority in handling similar scenarios, with only one occluded target missed and successfully detecting the remaining small targets, resulting in a total of 15 detected weed targets.



(**a**) YOLOv5n (**b**) YOLOv9t

(**c**) YOLOv10n (**d**) ADK-YOLOv8

Figure 14. Detection of a large number of small targets.

The following shows the performance of the four models when detecting both the ash and bluegrass targets. YOLOv5n not only missed one bluegrass target but also produced a false positive; YOLOv9t missed three bluegrass targets; and YOLOv10n missed two bluegrass targets. In contrast, the improved ADK-YOLOv8 network successfully detected all targets without any missed detections or false positives, as shown in Figure 15.

Next is their ability to detect occluded prickly lettuce targets. The image contains a complete target and an occluded target. The results show that both YOLOv9t and YOLOv10n fail to detect the occluded target, showing significant missed detections. In contrast, YOLOv5n and the improved ADK-YOLOv8 successfully detect all targets, including the occluded ones, as shown in Figure 16.

Through the above visual analysis, ADK-YOLOv8 is significantly better than other models in handling complex backgrounds and occluded targets. Its high accuracy is attributed to its improved network architecture and more efficient feature extraction mechanism. And it also solves the problem of high missed detection rate of mainstream lightweight models in small target detection in weeds. In addition, ADK-YOLOv8 performs well in complex scenarios with multiple targets, significantly reducing missed detections and false positives, demonstrating excellent adaptability and robustness. This performance advantage enhances the model's reliability and usefulness in real-world applications, especially in agricultural weed detection tasks that require high-accuracy capabilities.

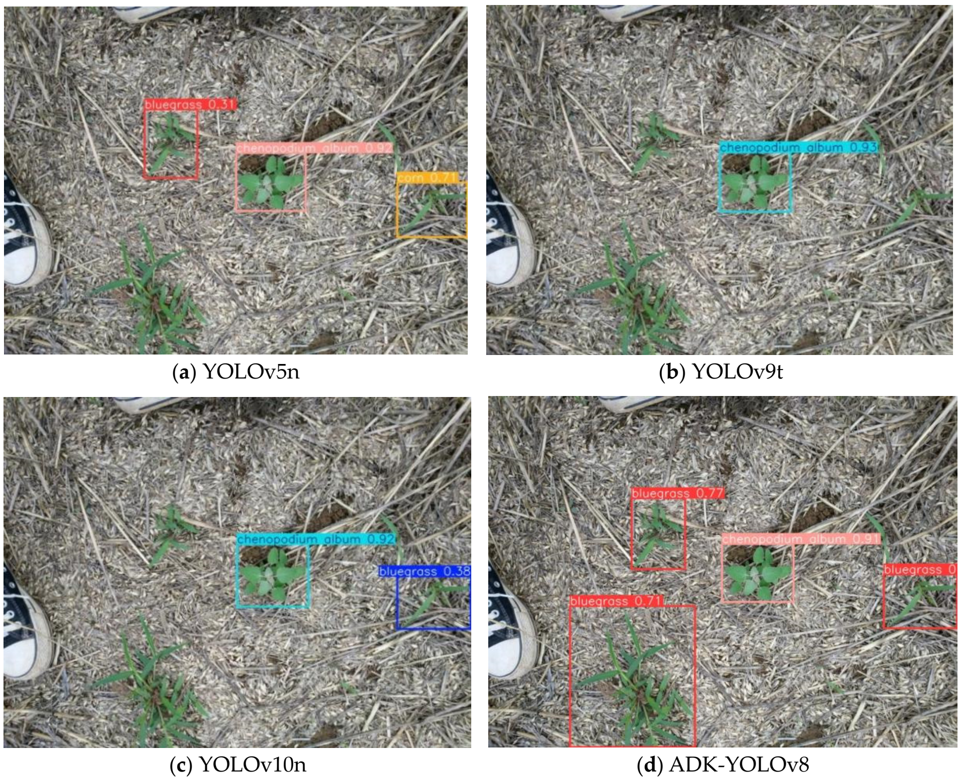


Figure 15. Multi-object detection.

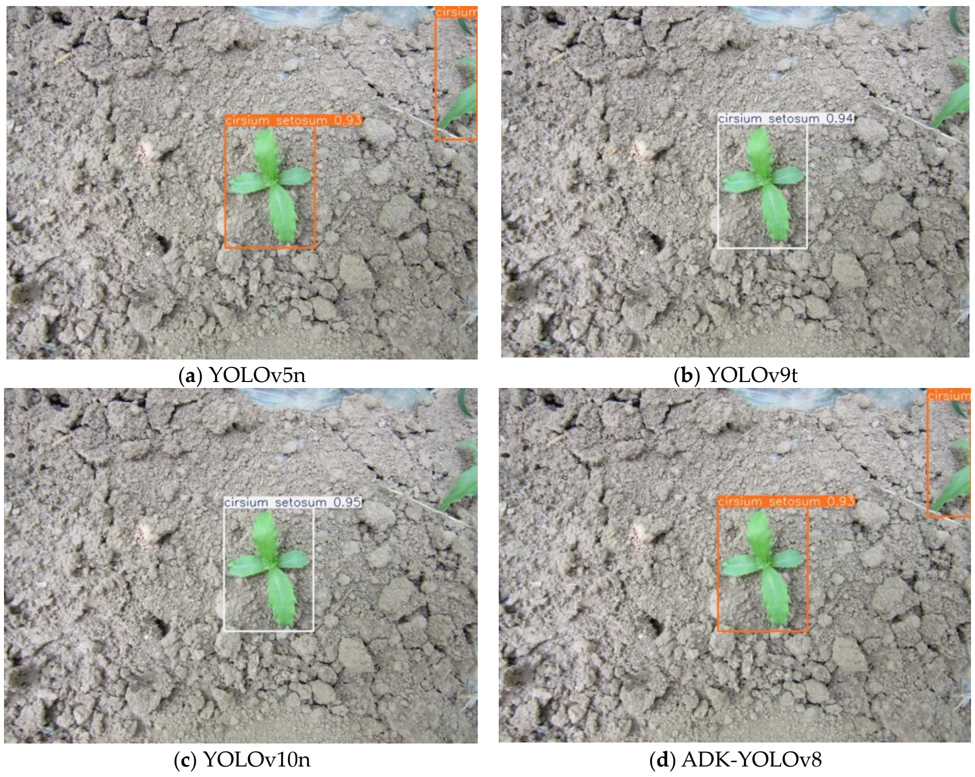


Figure 16. Detection of occluded targets.

4. Discussion

4.1. Comparison among Different YOLO Versions

In the comparative experiment with mainstream models, we newly carried out two experiments aiming to compare the performance of the latest target detection models, YOLOv11-S, RVDR-YOLOv8 and YOLOv8-ADL, in terms of accuracy, recall rate, mean average precision (mAP), model size, and giga floating-point operations per second (GFLOPS). Table 4 shows the comparison results of this experiment. It can be seen from the comparison results that our research has significant advantages in weed detection.

Table 4. The comparison between the improved model and the latest YOLO version as well as YOLO—variant models.

Model	Precision (%)	Recall (%)	mAP50 (%)	mAP50:95 (%)	GFLOPS	Model Size (MB)
RVDR-YOLOv8	91.88	87.24	93.34	73.64	7.4	5.43
YOLO v11s	92.14	88.32	94.09	74.35	7.2	5.27
ADL-YOLOv8	92.13	88.61	94.71	74.43	7.3	5.02

4.2. Current Research Challenges

Compared with the original network, the ADL-YOLOv8 model proposed in this article has achieved substantial improvements in various indicators of weed detection. Furthermore, it successfully achieves the goal of making the model lightweight. This feature opens up the possibility for affordable and cost-effective deployment on embedded devices or mobile devices. The model becomes more suitable for these types of devices. This not only expands the application scope of the model but also provides more convenience for users who need to detect weeds in various environments. However, there are still two challenges in the current experiment.

1. Variable appearance of weeds:

Weeds show significant differences in appearance under different lighting conditions and growth environments.

Weeds may grow in the shadows or be occluded by other vegetation, resulting in blurry images and increasing the difficulty of weed detection.

2. Improvement needed in the detection accuracy of small weeds or those with indistinct features:

Although the model has been improved in terms of small-object detection, the detection accuracy for very small weeds or those with textures very similar to the surrounding environment still needs to be enhanced. Because small-sized weeds may occupy only a tiny part of the image and have indistinct features, the identification is rather difficult.

4.3. Future Research Priorities

In future research, to address these limitations, specific data augmentation algorithms will be designed, considering the following aspects: 1. Simulating lighting changes: augmenting the dataset with images simulating different lighting conditions. 2. Simulating occlusion: adding samples where weeds are partially obscured by other vegetation. 3. Collecting diverse samples: gathering samples from various environments, including different times of day (morning and evening), different weather conditions (sunny, cloudy, and rainy), and different types of soil and background environments.

5. Conclusions

Precise herbicide application results in weed detection playing a crucial role in modern agriculture. It can not only effectively reduce the indiscriminate use of pesticides, thereby reducing agricultural production costs at the economic level, but also has great significance in environmental protection. It greatly alleviates the burden on the environment and strongly promotes the development of agriculture towards green and sustainable directions. Based on this, this paper proposes an improved weed detection model named ADL-YOLOv8.

We assume that the ADL-YOLOv8 model mainly introduces dynamic feature extractors and advanced attention mechanisms. These measures are aimed at achieving a lightweight model and improving its accuracy. Moreover, in the discussion section, by comparing with the latest YOLOv11s and the recent RVDR-YOLOv8, it is concluded that our experimental results support this hypothesis. The main research achievements are as follows:

1. During the model construction process, this method adopts the AKConv network. This network is lightweight and efficient and has unique advantages in processing targets of specific shapes. For example, when dealing with targets with elongated tubular structures, the AKConv network can better adapt to the shape characteristics of the targets, thereby extracting relevant feature information more accurately and effectively enhancing the performance of the entire model. This optimization for targets of specific shapes enables the model to handle complex-shaped weed targets more easily, providing a strong guarantee for the accuracy of weed detection.
2. In the neck part of the model, its upsampling module adopts a super-lightweight and efficient dynamic upsampler named DySample. This dynamic upsampler plays an indispensable role in the model and has a significant effect on improving the accuracy and mean average precision (mAP) of the model. Through this special upsampling method, the model can restore the detailed information in the image more accurately, making the features of weed targets in the image more obvious, thereby improving the accuracy of the model in weed detection and making the weed detection results more reliable.
3. Finally, the introduction of the attention mechanism is another highlight of this model. By introducing the attention mechanism, the model becomes more sensitive when detecting targets, especially when dealing with small targets. This attention mechanism can guide the model to focus on the key areas in the image, that is, the areas where weeds are located, thereby improving the detection accuracy of weeds. Specifically, this study integrates the high-performance LSKA-Attention into the C2f module of the original YOLOv8 model, thereby forming the C2f_LSKA_Attention module [45]. This newly formed module has multiple advantages. It can not only effectively reduce the computational complexity of the model and the consumption of computational resources during the model operation but also significantly improve the detection accuracy of the model for weeds, making the model more efficient and accurate when handling complex weed detection tasks.

From the experimental results, the ADL-YOLOv8 model shows excellent performance and surpasses other leading target detection models in multiple aspects. Especially in terms of model optimization, ADL-YOLOv8 reduces the computational requirements and parameter scale while improving the detection accuracy. Specifically, compared with the original YOLOv8n, ADL-YOLOv8 has 29.92% fewer parameters and 11.45% less computational load. These optimized model characteristics make it very suitable for being applied in environments with limited memory and computational resources, such as embedded devices. In embedded devices, the memory and computational capacity are often relatively limited, and the light weight and high efficiency of the ADL-YOLOv8 model enable it to run stably in such an environment and maintain high-performance weed detection capabilities.

These improvements confirm our hypothesis. The ADL-YOLOv8 model helps promote the development of agriculture towards a more intelligent and sustainable direction, which is of great significance for the sustainable development of the entire field of agriculture.

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

Funding: This research was funded by the National Natural Science Foundation of China (52275251) and The Earmarked Fund for CARS-18.

Data Availability Statement: Data are contained within the article.

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

References

1. Xiaoming, C.; Tianzeng, C.; Haomin, M.; Ziqi, Z.; Dehua, W.; Jianchao, S.; Jun, W. An improved algorithm based on YOLOv5 for detecting Ambrosia trifida in UAV images. *Front. Plant Sci.* **2024**, *15*, 1360419. [CrossRef] [PubMed]
2. Gao, L.; Zhao, X.; Yue, X.; Yue, Y.; Wang, X.; Wu, H.; Zhang, X. A Lightweight YOLOv8 Model for Apple Leaf Disease Detection. *Appl. Sci.* **2024**, *14*, 6710. [CrossRef]
3. Parra, L.; Marin, J.; Yousfi, S.; Rincón, G.; Mauri, P.V.; Lloret, J. Edge detection for weed recognition in lawns. *Comput. Electron. Agric.* **2020**, *176*, 105684. [CrossRef]
4. Gée, C.; Denimal, E. RGB image-derived indicators for spatial assessment of the impact of broadleaf weeds on wheat biomass. *Remote Sens.* **2020**, *12*, 2982. [CrossRef]
5. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I 14, pp. 21–37.
6. Zhao, M.; Su, Y.; Wang, J.; Liu, X.; Wang, K.; Liu, Z.; Liu, M.; Guo, Z. MED-YOLOv8s: A new real-time road crack, pothole, and patch detection model. *J. Real-Time Image Process.* **2024**, *21*, 26. [CrossRef]
7. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef] [PubMed]
8. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
9. Gan, D.; Gromiha, P. *Advanced Intelligent Computing Theories and Applications*; Springer: Berlin/Heidelberg, Germany, 2010.
10. Zhang, R.; Wang, C.; Hu, X.; Liu, Y.; Chen, S. Weed location and recognition based on UAV imaging and deep learning. *Int. J. Precis. Agric. Aviat.* **2020**, *3*, 23–29. [CrossRef]
11. Hu, D.; Ma, C.; Tian, Z.; Shen, G.; Li, L. Rice Weed detection method on YOLOv4 convolutional neural network. In Proceedings of the 2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA), Xi'an, China, 28–30 May 2021; pp. 41–45.
12. Wang, A.; Peng, T.; Cao, H.; Xu, Y.; Wei, X.; Cui, B. TIA-YOLOv5: An improved YOLOv5 network for real-time detection of crop and weed in the field. *Front. Plant Sci.* **2022**, *13*, 1091655. [CrossRef]
13. Gallo, I.; Rehman, A.U.; Dehkordi, R.H.; Landro, N.; La Grassa, R.; Boschetti, M. Deep object detection of crop weeds: Performance of YOLOv7 on a real case dataset from UAV images. *Remote Sens.* **2023**, *15*, 539. [CrossRef]
14. Ding, Y.; Jiang, C.; Song, L.; Liu, F.; Tao, Y. RVDR-YOLOv8: A Weed Target Detection Model Based on Improved YOLOv8. *Electronics* **2024**, *13*, 2182. [CrossRef]
15. Wang, A.; Zhang, W.; Wei, X. A review on weed detection using ground-based machine vision and image processing techniques. *Comput. Electron. Agric.* **2019**, *158*, 226–240. [CrossRef]
16. Jing, J.; Zhai, M.; Dou, S.; Wang, L.; Lou, B.; Yan, J.; Yuan, S. Optimizing the YOLOv7-Tiny Model with Multiple Strategies for Citrus Fruit Yield Estimation in Complex Scenarios. *Agriculture* **2024**, *14*, 303. [CrossRef]
17. 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, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
18. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
19. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
20. Ma, N.; Wu, Y.; Bo, Y.; Yan, H. Chili Pepper Object Detection Method Based on Improved YOLOv8n. *Plants* **2024**, *13*, 2402. [CrossRef] [PubMed]
21. Hung, M.-H.; Ku, C.-H.; Chen, K.-Y. Application of Task-Aligned Model Based on Defect Detection. *Automation* **2023**, *4*, 327–344. [CrossRef]
22. Zhang, Y.; Ni, Q. A novel weld-seam defect detection algorithm based on the s-yolo model. *Axioms* **2023**, *12*, 697. [CrossRef]
23. Zhang, T.; Zhang, J.; Pan, P.; Zhang, X. YOLO-RRL: A Lightweight Algorithm for PCB Surface Defect Detection. *Appl. Sci.* **2024**, *14*, 7460. [CrossRef]
24. Zheng, S.; Jia, X.; He, M.; Zheng, Z.; Lin, T.; Weng, W. Tomato Recognition Method Based on the YOLOv8-Tomato Model in Complex Greenhouse Environments. *Agronomy* **2024**, *14*, 1764. [CrossRef]
25. Zhang, X.; Song, Y.; Song, T.; Yang, D.; Ye, Y.; Zhou, J.; Zhang, L. LDConv: Linear deformable convolution for improving convolutional neural networks. *Image Vis. Comput.* **2024**, *149*, 105190. [CrossRef]
26. Tang, S.; Zhang, S.; Fang, Y. HIC-YOLOv5: Improved YOLOv5 for small object detection. In Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 13–17 May 2024; pp. 6614–6619.

27. Huynh-The, T.; Hua, C.-H.; Pham, Q.-V.; Kim, D.-S. MCNet: An efficient CNN architecture for robust automatic modulation classification. *IEEE Commun. Lett.* **2020**, *24*, 811–815. [CrossRef]
28. Jiang, T.; Zhou, J.; Xie, B.; Liu, L.; Ji, C.; Liu, Y.; Liu, B.; Zhang, B. Improved YOLOv8 Model for Lightweight Pigeon Egg Detection. *Animals* **2024**, *14*, 1226. [CrossRef]
29. Liu, W.; Lu, H.; Fu, H.; Cao, Z. Learning to upsample by learning to sample. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 6027–6037.
30. Yang, W.; Qiu, X. A lightweight and efficient model for grape bunch detection and biophysical anomaly assessment in complex environments based on YOLOv8s. *Front. Plant Sci.* **2024**, *15*, 1395796. [CrossRef] [PubMed]
31. Khaniki, M.A.L.; Mirzaeibonehkhater, M.; Manthouri, M. Enhancing Pneumonia Detection using Vision Transformer with Dynamic Mapping Re-Attention Mechanism. In Proceedings of the 2023 13th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 1–2 November 2023; pp. 144–149.
32. Zheng, Z.; Ge, Z.; Tian, Z.; Yang, X.; Zhou, Y. WoodGLNet: A multi-scale network integrating global and local information for real-time classification of wood images. *J. Real-Time Image Process.* **2024**, *21*, 147. [CrossRef]
33. Lau, K.W.; Po, L.-M.; Rehman, Y.A.U. Large separable kernel attention: Rethinking the large kernel attention design in cnn. *Expert Syst. Appl.* **2024**, *236*, 121352. [CrossRef]
34. Cong, S.; Zhou, Y. A review of convolutional neural network architectures and their optimizations. *Artif. Intell. Rev.* **2023**, *56*, 1905–1969. [CrossRef]
35. Wang, J.; Wang, Y.; Sun, A.; Zhang, Y. A Lightweight Network FLA-Detect for Steel Surface Defect Detection. *Res. Sq.* 2024, *preprint*. [CrossRef]
36. Chen, X.; Lv, J.; Fang, Y.; Du, S. Online detection of surface defects based on improved YOLOV3. *Sensors* **2022**, *22*, 817. [CrossRef]
37. Yang, J.; Hu, Q.; Cheng, M.-M.; Wang, L.; Liu, Q.; Bai, X.; Meng, D. *Computer Vision: Second CCF Chinese Conference, CCCV 2017, Tianjin, China, October 11–14, 2017, Proceedings, Part III*; Springer: Berlin/Heidelberg, Germany, 2017; Volume 773.
38. Du, W.; Jiang, G.; Xu, W.; Ma, J. Sequential patent trading recommendation using knowledge-aware attentional bidirectional long short-term memory network (KBiLSTM). *J. Inf. Sci.* **2023**, *49*, 814–830. [CrossRef]
39. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722.
40. Gao, H.; Sun, Y.; Shi, W. The Internet of Things Drives Smart City Management: Enhancing Urban Infrastructure Efficiency and Sustainability. *J. Organ. End User Comput. (JOEUC)* **2024**, *36*, 1–17. [CrossRef]
41. Deng, X.; Liu, Q.; Deng, Y.; Mahadevan, S. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Inf. Sci.* **2016**, *340*, 250–261. [CrossRef]
42. Wang, C.-Y.; Yeh, I.-H.; Liao, H.-Y.M. Yolov9: Learning what you want to learn using programmable gradient information. *arXiv* **2024**, arXiv:2402.13616.
43. Wang, A.; Chen, H.; Liu, L.; Chen, K.; Lin, Z.; Han, J.; Ding, G. Yolov10: Real-time end-to-end object detection. *arXiv* **2024**, arXiv:2405.14458.
44. Uygun, T.; Ozguven, M.M. Real-Time Detection of Shot-Hole Disease in Cherry Fruit Using Deep Learning Techniques via Smartphone. *Appl. Fruit Sci.* **2024**, *66*, 875–885. [CrossRef]
45. Tang, Z.; Zhang, W.; Li, J.; Liu, R.; Xu, Y.; Chen, S.; Fang, Z.; Zhao, F. LTSCD-YOLO: A Lightweight Algorithm for Detecting Typical Satellite Components Based on Improved YOLOv8. *Remote Sens.* **2024**, *16*, 3101. [CrossRef]