Available online at website: https://jurnal.iaii.or.id/index.php/RESTI

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 9 No. 5 (2025) 1153 - 1162 e-ISSN: 2580-0760

Hyperparameter Tuning with Optuna to optimize the YOLOv11n Model for Weed Detection

Candhy Fadhila Arsyad[1], Pulung Nurtantio Andono[2], Moch Arief Soeleman[3]
[1,2,3]Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia

[1]p31202302594@mhs.dinus.ac.id, [2]pulung@dsn.dinus.ac.id, [3]arief2802@dsn.dinus.ac.id

Abstract

Accurate weed detection is essential for maintaining the cleanliness and aesthetic appeal of residential yards. This study aimed to optimize YOLOv11n, a lightweight object detection model, to achieve high precision in weed identification under real-world conditions. The novelty of this study lies in the application of Optuna, an automatic hyperparameter optimization framework, to enhance model performance while maintaining computational efficiency for resource-limited devices such as drones and IoT systems. The research involved data augmentation techniques including crop (0–20% zoom), hue (±20°), saturation (±30%), brightness (±20%), exposure (±15%), and mosaic augmentation. These augmented images were used to train four YOLO nano variants (v5n, v8n, v11n, v12n), which were evaluated using standard metrics: Precision, Recall, F1-Score, and mean Average Precision (mAP). Among the models tested, YOLOv11n with Custom Optuna configuration delivered the highest performance, achieving a 94.6% F1-score and 97.8% mAP@0.5. These results demonstrate that the optimized YOLOv11n model can support accurate and efficient real-time weed detection in household environments, particularly on edge devices with limited hardware capabilities. This makes it a viable solution for practical implementation in precision agriculture and smart gardening.

Keywords: image augmentation; Optuna; residential weed detection; YOLO nano; YOLOv11n

How to Cite: Candhy Fadhila Arsyad, Pulung Nurtantio Andono, and Moch Arief Soeleman, "Hyperparameter Tuning with Optuna to optimize the YOLOv11n Model for Weed Detection", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 9, no. 5, pp. 1153 - 1162, Oct. 2025.
Permalink/DOI: https://doi.org/10.29207/resti.v9i5.6682

Received: May 28, 2025
Accepted: September 20, 2025
Available Online: [October 25, 2025

This is an open-access article under the CC BY 4.0 License
Published by Ikatan Ahli Informatika Indonesia

1. Introduction

Weeds are invasive plants that commonly appear in various ecosystems, both in agricultural lands and home gardens. Although the presence of weeds in home gardens does not always threaten crop yields as it does in agriculture, their existence can affect the aesthetics, cleanliness, and environmental health of the surrounding area. Weeds can compete with ornamental plants or food crops in home gardens for water, sunlight, and soil nutrients. Additionally, weeds may serve as breeding grounds for pests and diseases that can damage plants in the vicinity of home gardens [1], [2]. According to data released by the Food and Agriculture Organization (FAO), weed infestations in home gardens can reduce the aesthetics and health of nearby plants and increase the costs of garden maintenance and upkeep [3].

Amid these challenges, computer vision-based detection technology has emerged as a promising modern approach to automatically and efficiently address the problem of weed management. One of the most widely adopted object detection algorithms is You Only Look Once (YOLO), which is renowned for its high inference speed and computational efficiency. YOLO has evolved significantly from its initial versions to lightweight variants such as YOLOv5n, YOLOv8n, and the most recent YOLOv11n, which are specifically designed to operate on devices with limited computational resources while maintaining competitive detection accuracy [4], [5].

The selection of YOLOv11n in this study is based on careful consideration. While Convolutional Neural Network (CNN)-based approaches, such as InceptionV3 and InceptionResNetV2, have demonstrated strong performance in plant disease classification tasks for instance, Firnando et al. (2024) reported an accuracy of 99.53% when using these architectures on rice leaf disease datasets [6] CNNs have inherent limitations. These include their inability to capture global contextual information within images

and their high computational requirements during both training and inference. As such, CNNs are less suitable for lightweight applications, such as in home environments or on resource-constrained IoT devices. An alternative that has also shown promising performance is the Vision Transformer (ViT), as highlighted in the study by Kristiyanto et al. (2024). ViT utilizes self-attention and multi-head attention mechanisms to capture global visual patterns, achieving a recall of 97.16% in rice disease classification [7]. However, ViT suffers from high architectural complexity, substantial memory requirements, and significantly longer training times compared to CNNs, making it less practical for edge computing scenarios or real-time systems with limited hardware capabilities.

YOLOv11n, on the other hand, represents an intriguing balance between high accuracy and computational efficiency. This model is capable of quickly and efficiently detecting small objects and is specifically designed to perform well in environments with varying lighting conditions, complex backgrounds, and high visual dynamics, as commonly found in home gardens. Additionally, YOLOv11n excels in its lightweight architecture, which enables integration with micro devices such as Raspberry Pi or other IoT sensors, making it highly applicable for automated environmental monitoring systems.

The application of YOLO-based object detection has been used in various fields. In a study by Mei and Zhu (2024), the BGF-YOLOv10 algorithm was developed to enhance small object detection in low-resolution UAV images using BoTNet and GhostConv techniques. However, this model still faces limitations in computational power efficiency [8]. Nieradzik et al. (2024) developed WoodYOLO, optimized for microscopic wood fiber analysis with better memory efficiency, but it is less suitable for detecting complex natural objects like weeds [9]. Xi et al. (2021) developed YOLOv3-Multi with Spatial Pyramid Pooling (SPP) to improve small object detection, but this model has yet to be tested in dynamic agricultural environments [10]. Singha and Aydin (2021) used YOLOv4 to detect small drones in various weather conditions but faced difficulties in detecting objects overlapping with complex backgrounds [11]. Sportelli et al. (2023) compared the models YOLOv5, YOLOv6, YOLOv7, and YOLOv8 using the 'Weeds' dataset, with YOLOv8l achieving a precision of 0.9476 and mAP@0.5 of 0.9795. However, this model still faces weaknesses in terms of slower inference speed compared to other models like YOLOv5m [12].

Based on previous studies, the application of YOLO in small object detection, UAV surveillance, and microscopic analysis has shown progress, but still faces significant limitations, such as suboptimal accuracy under varying lighting conditions, challenges in detecting overlapping objects, and high computational power requirements. In the context of home gardens, the complexity of weed shapes and colors, as well as dynamic lighting, pose barriers to reliable detection. Although there are YOLO models optimized for specific conditions, such as WoodYOLO, most have not been tested for non-uniform natural objects, and there has been no exploration of the performance of YOLOv11n or other nano models in weed detection. Unlike the study by Sportelli et al. (2023), the research conducted by the author focuses more on efficiency, accuracy, and lightweight design using nano YOLO variants such as YOLOv11n for efficient weed detection on devices with limited resources. However, conventional YOLO models generally do not meet the computational efficiency requirements for IoT devices in agriculture, and there has been inadequate evaluation under extreme conditions and low lighting, thus necessitating the development and comparison of YOLOv11 nano models in weed detection in home gardens.

Based on the analysis, there is a gap between ideal conditions and real-world challenges in weed detection using YOLO. The ideal condition expects object detection to be accurate under various lighting conditions, environments, and devices with limited power. However, real-world challenges show that existing models are still limited in detecting small objects, handling dynamic lighting, and working efficiently on low-power devices. This study aims to address this gap by optimizing YOLOv11n and comparing it with previous nano YOLO versions for weed detection in home gardens, with higher efficiency and better accuracy. This model is expected to work optimally in complex field conditions through specific data augmentation and hyperparameter tuning, while maintaining a balance between detection speed and accuracy for application in IoT-based agricultural systems and devices with limited resources [13].

To ensure that the developed model is truly relevant in real-world conditions, performance evaluation is not conducted using a single metric, but rather several key metrics such as Precision, Recall, F1-Score, and Mean Average Precision (mAP) under the 0.5 IoU and 0.5:0.95 scenarios. Among these metrics, the primary focus is placed on mAP@0.5:0.95 and F1-Score as they provide a comprehensive overview of the model's spatial accuracy and classification quality. mAP@0.5:0.95 measures detection precision across varying bounding box overlap thresholds, while F1-Score captures the balance between the model's ability to correctly identify weeds (recall) and avoid prediction errors (precision). The emphasis on these two metrics is deemed crucial as weeds exhibit significant variation in shape and size, and often appear under inconsistent lighting conditions in outdoor environments such as home gardens.

To address the challenges of weed detection in home gardens, the proposed solution involves using specific data augmentation techniques such as crop, hue, saturation, brightness, exposure, and mosaic to increase data diversity during the training dataset. YOLOv11n

was selected due to its low computational requirements and the balance it offers between speed and accuracy, with a lighter and more efficient architecture. Additionally, hyperparameter tuning will be conducted to ensure the model performs optimally in detecting weeds. This tuning process involves adjusting key parameters such as learning rate, batch size, and other optimizers that directly affect model training performance. This study will focus on developing and applying solutions to address the challenges of weed detection in home gardens, thereby enhancing the efficiency and effectiveness of household environmental management. The dataset is sourced from the Roboflow website at https://app.roboflow.com/gulma/weeds-nxe1w-c9dwl/4.

2. Methods

2.1 Dataset

The image dataset employed in this research was sourced from the Roboflow platform, comprising 4,203 annotated weed images collected from household garden environments [14]. Prior to model training, each image was standardized to a resolution of 640×640 pixels, adhering to the input specifications of the YOLO architecture [15], [16]. To improve the model's capacity for variation and generalization, various image augmentation strategies were implemented, as illustrated in Figure 1 and Table 1.

Table 1. Types of augmentation applied

Augmentation Type	Details
Crop	0% Minimum Zoom, 20% Maximum Zoom
Hue	Between -20° and +20°
Saturation	Between -30% and +30%
Brightness	Between -20% and +20%
Exposure	Between -15% and +15%
Mosaic	Applied

These include cropping with a minimum zoom of 0% and a maximum of 20%, hue adjustment ranging from -20° to +20°, saturation variation between -30% and +30%, brightness adjustment from -20% to +20%, and exposure modification from -15% to +15%. In addition, mosaic augmentation was implemented by combining four different images into a single frame. These augmentation techniques were systematically applied to increase visual diversity and strengthen the model's ability to generalize across various lighting conditions and background complexities.

Object annotation was performed in the form of bounding boxes using the Roboflow interface, and the results were converted into YOLO format. After the augmentation process, the total number of images increased to 6,457. This dataset was later divided into three partitions, consisting of 70% for training purposes, 20% for validation, and the remaining 10% allocated for testing.



(crop) (hue)

(saturation) (brightness)

(exposure) (mosaic)

Figure 1. Sample image of the augmented dataset

2.2. Experimental design

YOLOv11n represents a compact version of the You Only Look Once (YOLO) algorithm, tailored for high-performance real-time object detection while maintaining lightweight characteristics. Its architecture emphasizes both speed and precision, making it suitable for implementation on devices with constrained computational capacity, such as IoT systems or drones [17]. As illustrated in Figure 2, the YOLOv11 structure is composed of three primary modules: the Backbone, the Neck, and the Head, each contributing respectively to feature extraction, information integration, and prediction tasks from input imagery.

The Backbone module leverages the Cross-Stage Partial Network (CSPNet), which effectively extracts spatial and semantic features from images. This operation is performed incrementally through multiple convolutional layers and C3K2 units, supported by shortcut connections to preserve critical structural details like edges, patterns, and textures from the weed samples [18].

Following this, the Neck module fuses multi-scale feature maps by integrating the Feature Pyramid Network (FPN) and Path Aggregation Network (PANet). This integration equips the model with the capability to detect objects across varying scales.

Enhancements in this stage are achieved through the inclusion of SPPF (Spatial Pyramid Pooling-Fast) and C2PSA (Cross-Channel Partial Self-Attention), which enrich contextual understanding and improve focus on crucial regions within the image.

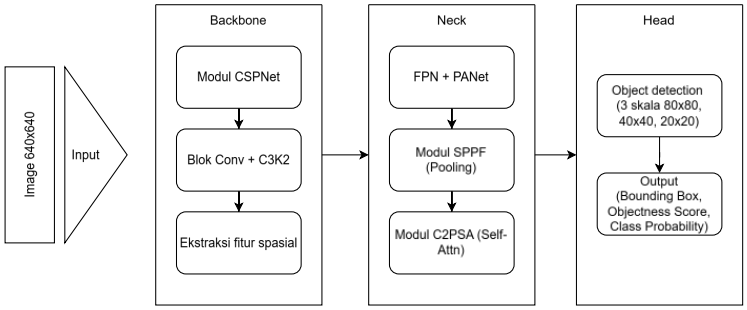


Figure 2. YOLOv11 architecture

At the final stage, the Head component produces three essential outputs: the bounding box coordinates to localize detected objects, an objectness confidence score indicating the likelihood of an object's presence, and classification probabilities to determine the object category. These predictions are conducted over three resolution scales 80×80, 40×40, and 20×20 allowing efficient detection of weeds with diverse dimensions [19].

Although the YOLOv11n architecture follows the modular structure of YOLOv11 overall, this nano version is designed with a lighter parameter capacity to optimize computational efficiency. Based on the model summary architecture shown in Figure 3, YOLOv11n has only about 2.59 million parameters and 6.4 GFLOPs, which is much lower compared to the larger versions of YOLOv11. Some blocks, such as C3K2, consist of only 1 to 3 convolution units, and the number of channels in each layer is reduced to 64, 128, or 256. This allows the model to remain fast during inference while maintaining competitive detection performance, especially for small objects like weeds.

	from	n	params	module	arguments
0	-1	1	464	ultralytics.nn.modules.conv.Conv	[3, 16, 3, 2]
1	-1	1	4672	ultralytics.nn.modules.conv.Conv	[16, 32, 3, 2]
2	-1	1	6640	ultralytics.nn.modules.block.C3k2	[32, 64, 1, False, 0.25]
3	-1	1	36992	ultralytics.nn.modules.conv.Conv	[64, 64, 3, 2]
4	-1	1	26080	ultralytics.nn.modules.block.C3k2	[64, 128, 1, False, 0.25]
5	-1	1	147712	ultralytics.nn.modules.conv.Conv	[128, 128, 3, 2]
6	-1	1	87040	ultralytics.nn.modules.block.C3k2	[128, 128, 1, True]
7	-1	1	295424	ultralytics.nn.modules.conv.Conv	[128, 256, 3, 2]
8	-1	1	346112	ultralytics.nn.modules.block.C3k2	[256, 256, 1, True]
9	-1	1	164608	ultralytics.nn.modules.block.SPPF	[256, 256, 5]
10	-1	1	249728	ultralytics.nn.modules.block.C2PSA	[256, 256, 1]
11	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
12	[-1, 6]	1	0	ultralytics.nn.modules.conv.Concat	[1]
13	-1	1	111296	ultralytics.nn.modules.block.C3k2	[384, 128, 1, False]
14	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
15	[-1, 4]	1	0	ultralytics.nn.modules.conv.Concat	[1]
16	-1	1	32096	ultralytics.nn.modules.block.C3k2	[256, 64, 1, False]
17	-1	1	36992	ultralytics.nn.modules.conv.Conv	[64, 64, 3, 2]
18	[-1, 13]	1	0	ultralytics.nn.modules.conv.Concat	[1]
19	-1	1	86720	ultralytics.nn.modules.block.C3k2	[192, 128, 1, False]
20	-1	1	147712	ultralytics.nn.modules.conv.Conv	[128, 128, 3, 2]
21	[-1, 10]	1	0	ultralytics.nn.modules.conv.Concat	[1]
22	-1	1	378880	ultralytics.nn.modules.block.C3k2	[384, 256, 1, True]
23	[16, 19, 22]	1	430867	ultralytics.nn.modules.head.Detect	[1, [64, 128, 256]]

YOLOv11n summary: 181 layers, 2,590,035 parameters, 2,590,019 gradients, 6.4 GFLOPs

Figure 3. YOLOv11n model summary

The experimental flow in this study is designed to optimize the performance of weed object detection by utilizing the lightweight and computationally efficient YOLOv11n architecture. As shown in Figure 3, the process begins with data processing through the Roboflow platform, which supports a structured annotation pipeline and dataset preparation.

The preprocessing stage in Figure 4, starts with the input of weed images taken from the garden. Manual annotation is then performed by marking bounding boxes around the weed objects using the Roboflow interface. After annotation is completed, automatic preprocessing is carried out, including resizing the images to 640×640 pixels to match the standard YOLO input format. The next step is the application of data augmentation to enrich visual variation, including crop, hue, saturation, brightness, exposure, and mosaic augmentation.

The next stage is hyperparameter tuning, which is done automatically using the Optuna algorithm. This process involves deeper exploration into combinations of hyperparameter values such as initial and final learning rates (lr0, lrf), momentum, warmup epoch count, warmup momentum, batch size, and optimizer (AdamW). The goal is to achieve a better training configuration that provides a balance between precision and model stability [20].

After determining the optimal configuration, the training process is carried out using the YOLOv11n

framework. The training utilizes the preprocessed dataset, which has been proportionally split into 70% for training, 20% for validation, and 10% for testing. The model's performance is assessed at every epoch using evaluation metrics including precision, recall, and the F1-score.

In the inference phase, the outputs generated by the model namely bounding box coordinates, objectness confidence, and class predictions are refined through Non-Maximum Suppression (NMS) to discard overlapping or redundant detections. A confidence threshold is also applied to exclude results with low reliability. The final output of the detection system comprises the weed's position in the image (bounding box), its identified class, and the corresponding confidence level.

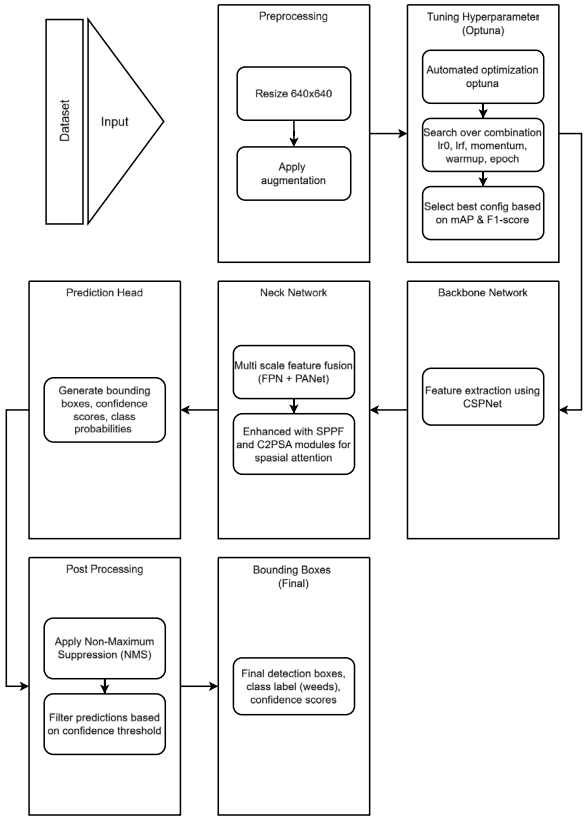


Figure 4. Experiment flow model

The model training process is conducted using the Ultralytics YOLO framework based on PyTorch, with support from an NVIDIA A100-SXM4 40GB GPU provided through the Google Colab Pro platform. The use of this high-end GPU allows the training process to run faster and more stably, especially for hyperparameter tuning and iterative training of lightweight YOLO models. The entire model is trained for 100 epochs using images sized 640×640 pixels and a batch size of 128. The dataset is divided into three parts: 70% for training, 20% for validation, and 10% for testing. All experiments are conducted with uniform configurations to ensure fair comparison results across models.

For model weight optimization, the AdamW algorithm was employed as the optimizer due to its ability to maintain training stability and prevent overfitting. Learning rate scheduling was performed using the Cosine Annealing method, wherein the learning rate gradually decreases from the initial value to the final value following a cosine function [21], [22]. The initial learning rate (lr0), final learning rate (lrf), and additional parameters such as momentum, warmup epochs, and warmup momentum were adjusted based on the hyperparameter tuning process [23].

The hyperparameter tuning approach was conducted using the Optuna algorithm, a trial-based optimization framework that automatically searches for the best combination of parameters including lr0, lrf, momentum, warmup epochs, warmup momentum, batch size, and optimizer type [24]. Table 2 presents a comparison of hyperparameter configurations between the untuned condition, default Optuna tuning, and custom Optuna tuning, highlighting parameter adjustments that impact the performance of the YOLOv11n model. This tuning process enables improvements in accuracy and training efficiency of the YOLO nano model by accelerating adaptive exploration of the parameter space, reducing manual experimentation time, and maintaining model stability and performance, particularly when deployed on devices with limited computational resources.

Table 2. Hyperparameter types and values

Parameter	Without Tune	Default Optuna	Custom Optuna
epoch	100	100	100
batch	16	128	128
imgsz	640	640	640
optimizer	AdamW	AdamW	AdamW
lr0	0.01	0.00923	0.0075
lrf	0.01	0.01165	0.01
momentum	0.937	0.8412	0.8412
weight_decay	0.0005	0.00069	0.00069
warmup_epochs	3.0	4.63251	4.63251
warmup_momentum	0.8	0.70459	0.70459
box	7.5	7.88407	7.88407
cls	0.5	0.64288	0.64288
dfl	1.5	0.97741	0.97741
hsv_h	0.015	0.01018	0.01018
hsv_s	0.7	0.59085	0.59085
hsv_v	0.4	0.19751	0.19751
degrees	0.0	0.0	0.0
translate	0.1	0.07771	0.07771
scale	0.5	0.44419	0.44419
shear	0.0	0.0	0.0
perspective	0.0	0.0	0.0
flipud	0.0	0.0	0.0
fliplr	0.5	0.5559	0.5559
bgr	0.0	0.0	0.0
mosaic	1.0	0.33633	0.33633
mixup	0.0	0.0	0.0
copy_paste	0.0	0.0	0.0

This study involves four lightweight YOLO model variants: YOLOv5n, YOLOv8n, YOLOv11n, and YOLOv12n, all of which are nano variants designed specifically for computational efficiency. Each model is trained with identical configurations and datasets to

allow for objective and measurable performance evaluation.

2.3. Performance metrics

To assess the performance of the object detection model developed in this study, three key evaluation metrics are employed: Precision, Recall, and mean Average Precision (mAP). These metrics were selected due to their ability to offer a holistic assessment of the model's effectiveness in detecting and categorizing weed instances with high accuracy [25].

Precision reflects the ratio of correctly identified positive predictions to the total number of predicted positives. It is computed using Formula 1

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

Here, TP refers to the count of true positive detections or weed instances that the model correctly identified, and FP denotes false positives or cases where non-weed objects were mistakenly classified as weeds. High precision indicates that the model has a low prediction error rate.

Recall, on the other hand, evaluates the model's effectiveness in retrieving all relevant weed objects within an image. The corresponding recall formula is expressed as Formula 2.

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

In this context, FN stands for false negatives, representing weed objects that the model failed to detect. A high recall score signifies the model's robustness in identifying nearly all weed instances, including those that are small in size or appear in visually complex regions.

To assess overall performance, this study also employs mean Average Precision (mAP), a standard metric in object detection that quantifies the area beneath the precision-recall curve. Specifically, two variations of mAP are utilized: (1.) mAP@0.5: the mean precision computed at an Intersection over Union (IoU) threshold of 0.5. (2). mAP@0.5:0.95: the mean precision calculated across IoU thresholds ranging from 0.5 to 0.95, with a 0.05 step interval.

The average precision (AP) is computed as the integral of the precision-recall curve, given by Formula 3.

$$\text{AP} = \int_0^1 P(R)\, dR \quad (3)$$

In this context, $P(R)$ denotes the precision as a function of recall. The mean Average Precision (mAP) is then calculated as the mean of the Average Precision (AP) values across all object classes. Since this study focuses on a single class (weeds), the mAP value directly corresponds to the AP for that class [26].

To complement this, the F1-score is also employed as an evaluation metric. It represents the harmonic mean between precision and recall and is calculated using Formula 4.

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

By using this combination of metrics Precision, Recall, mAP, and F1-score, the performance of the YOLOv11n model can be comprehensively evaluated and compared against other YOLO-based architectures in a fair and consistent manner [27].

3. Results and Discussions

This research involved a comparative analysis of four YOLO model variants YOLOv5n, YOLOv8n, YOLOv11n, and YOLOv12n for the task of weed detection in residential garden environments. Each model was trained for 100 epochs, and the performance outcomes are summarized in Table 3. The evaluation employed several performance indicators, including Precision, Recall, F1-Score, mAP@0.5, mAP@0.5:0.95, and the total training time.

Table 3. YOLO performance model in various conditions

Model	Condition	P	R	F1-Score	mAP50	Training Time(min)
YOLOv5n	Without Tune	0.945	0.918	0.931	0.964	54.18
	Default Optuna	0.945	0.915	0.930	0.965	31.38
	Custom Optuna	0.941	0.922	0.931	0.965	31.14
YOLOv8n	Without Tune	0.943	0.928	0.935	0.970	50.16
	Default Optuna	0.930	0.917	0.924	0.963	30.24
	Custom Optuna	0.937	0.917	0.927	0.962	30.00
YOLOv11n	Without Tune	0.933	0.929	0.931	0.968	59.58
	Default Optuna	0.949	0.934	0.941	0.974	31.44
	Custom Optuna	0.949	0.944	0.946	0.978	31.38
YOLOv12n	Without Tune	0.926	0.908	0.917	0.960	73.50
	Default Optuna	0.943	0.914	0.928	0.966	40.02
	Custom Optuna	0.933	0.900	0.916	0.958	41.28

YOLOv5n showed excellent performance in the 'Without Tune' condition, which is the default condition of YOLO without any parameter changes. In this condition, Precision reached 0.945 and F1-Score 0.931. However, the training time was longer (54.18 minutes). Nevertheless, these results demonstrate the strength of YOLOv5n even without tuning.

YOLOv8n also performed best in the 'Without Tune' condition, with Precision 0.943 and F1-Score 0.935. However, performance decreased in the 'Default

Optuna' and 'Custom Optuna' conditions, with a slight drop in Precision and F1-Score. Default Optuna is the condition where the initial parameters provided by the Optuna algorithm are automatically adjusted to achieve the best combination based on prior training (similar to following a recipe with predefined measurements by Optuna).

On the other hand, YOLOv11n with 'Custom Optuna' provided the most optimal performance, achieving Precision 0.949 and F1-Score 0.946, as well as the highest mAP@0.5 (0.978). Custom Optuna refers to a condition where the default parameters from Optuna have been further adjusted such as tuning `lr0`, `lrf`, momentum, `warmup_epochs`, and `warmup_momentum` to maximize the model's performance. While the 'Default Optuna' configuration already yielded strong results, further manual refinement resulted in significant improvements in Recall and F1-Score, demonstrating the impact of fine-tuned optimization.

The superior performance of YOLOv11n under the Custom Optuna configuration can be attributed to several architectural and optimization advantages. First, the compact yet powerful structure of YOLOv11n especially the integration of CSPNet and C2PSA modules enables efficient feature extraction and enhanced focus on critical visual regions. This architecture is particularly effective in handling small and overlapping weed objects, which are common at home garden environments.

Second, the custom hyperparameter tuning refined key training parameters such as the initial learning rate (`lr0`), final learning rate factor (`lrf`), momentum, and warmup epochs. These adjustments ensured smoother convergence during training and better generalization during inference. Notably, the learning rate was slightly reduced to prevent overshooting, and momentum was optimized to maintain training stability. This combination yielded an ideal balance between training efficiency and model accuracy.

In practical terms, YOLOv11n's performance 94.6% F1-score and 97.8% mAP@0.5 indicates high reliability for real-time weed detection using low-power devices. This is crucial for IoT applications in precision agriculture where computational resources are limited. The low inference time and lightweight model size (~2.6 million parameters) also make YOLOv11n suitable for deployment on edge devices such as Raspberry Pi, enabling autonomous monitoring of household gardens with minimal latency.

Compared to the baseline models YOLOv5n and YOLOv8n, YOLOv11n demonstrates not only accuracy but also robustness under varied lighting conditions and complex visual backgrounds, which are often present in outdoor environments. The consistent detection shown across different training conditions validates the effectiveness of combining architectural improvements with data-driven hyperparameter tuning. These findings underline the importance of both model selection and optimization strategy. While previous studies focused primarily on default settings or large architectures, this research confirms that targeted tuning of compact models can achieve state-of-the-art performance, offering a more scalable and resource-efficient solution for real-world use in smart agriculture systems.

The confusion matrix displayed in Figure 5 shows that the model has excellent performance in detecting weeds (Weeds) in agricultural images. With a high number of True Positives (TP), reaching 3,296, the model successfully classified most weed objects correctly. On the other hand, False Positives (FP), which were recorded at 342, represent the number of detection errors where the model misclassified background as weeds, while False Negatives (FN), recorded at only 111, indicate that most weed objects were detected well. These results signify high levels of Precision and Recall, highlighting the model's capability to accurately distinguish relevant targets while reducing misclassification of irrelevant objects.

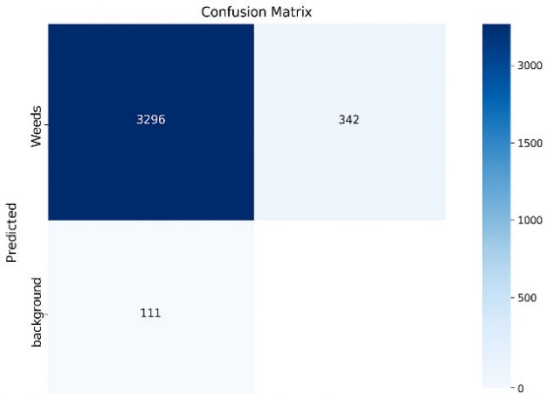


Figure 5. Confusison matrix on YOLOv11n 'Custom Optuna' condition

Based on the Result Curve graph in Figure 6, it can be seen that during the training process, the loss values for both training and validation data showed a consistent decline, indicating that the model was successfully minimizing prediction errors, both for bounding boxes and object classification. The metrics Precision, Recall, mAP50, and mAP50-95 also showed significant improvements, indicating that the model is not only becoming more accurate in detecting weeds but also more stable in its performance, even on previously unseen validation data. The model showed clear convergence around the 70th epoch, where the loss values began to stabilize, and performance metrics reached optimal values. Thus, the model demonstrates optimal performance in detecting weed objects with high efficiency, making it suitable for application in IoT-based precision agriculture systems.

The detection outputs presented in Figure 7 confirm that the model achieves high accuracy in identifying weed objects, as indicated by the bounding boxes assigned to each detected instance. Each bounding box includes a

confidence score that represents the model's level of certainty in its prediction. In the first sample image, several weeds were detected with notable confidence values 0.90, 0.85, and 0.82 demonstrating the model's effectiveness even when dealing with smaller or overlapping objects. Similarly, the second images exhibit strong detection performance, with respective confidence scores of 0.91, 0.87, and 0.83, underscoring the model's ability to maintain reliable detection across varying backgrounds and object dimensions. Overall, these outcomes highlight the model's robustness and efficiency in handling diverse visual scenarios with a high degree of precision.

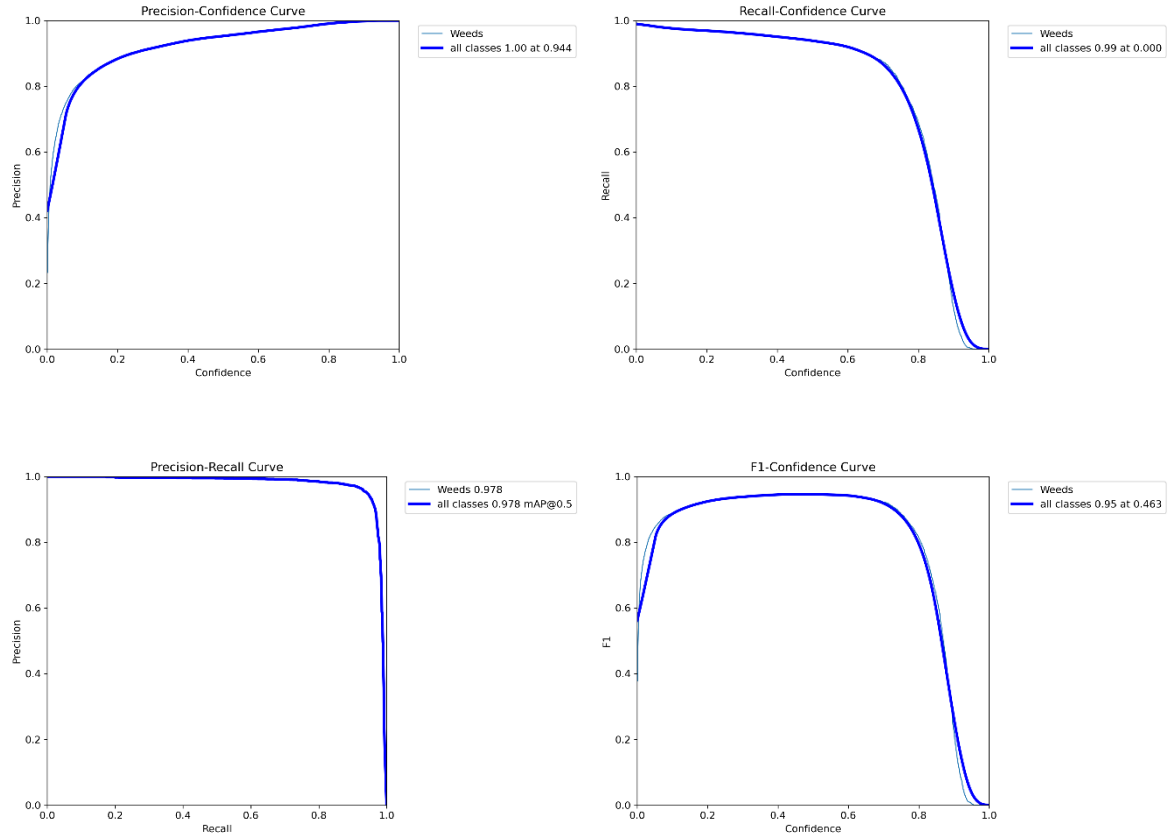


Figure 6. Result Curve on YOLOv11n 'Custom Optuna' condition

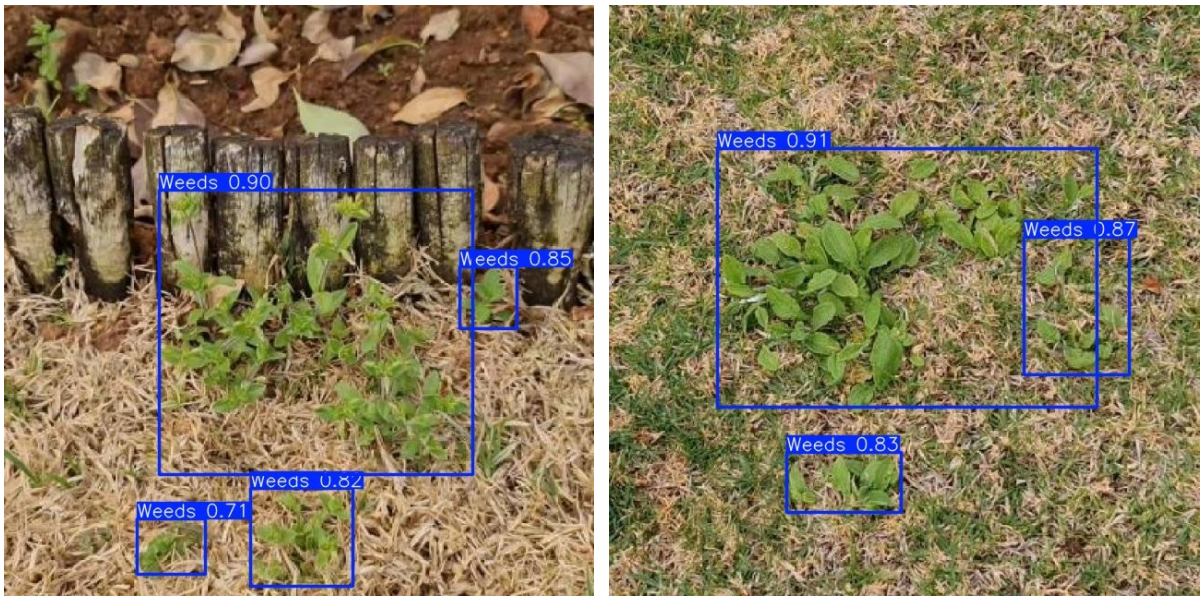


Figure 7. Image with object detection for garden weeds

To assess the benefits of the approach proposed in this research, a comparative performance analysis was conducted, focusing specifically on YOLO nano variants. The evaluation concentrated on key metrics: Precision (P), Recall (R), and mAP@0.5. For benchmarking purposes, the study by Sportelli et al [12], which evaluated YOLOv5n and YOLOv8n on the public 'Weeds' dataset under default training settings, served as a reference. Their experimental setup included an image resolution of 640×640 pixels, batch size of 8, 100 training epochs, a learning rate of 0.01, and the use of the Stochastic Gradient Descent (SGD) optimizer with a momentum value of 0.937. The applied augmentation methods encompassed hue, saturation, brightness, translation, scaling, flipping, and mosaic techniques.

In this study, to establish a fair baseline comparison under the 'without tune' condition, a similar configuration was adopted. YOLOv5n and YOLOv8n models were trained with an input size of 640×640 pixels, for 100 epochs, and using a slightly larger batch size of 16 to better utilize modern GPU capabilities. The AdamW optimizer with its default learning rate was employed, and no additional parameter tuning was performed, preserving the models' default state for evaluation.

The results showed in Table 4 that under the 'without tune' condition, the YOLOv5n model in this study achieved a precision of 0.945, recall of 0.918, and mAP@0.5 of 0.964, values that were significantly higher than those reported by Sportelli et al., which were precision 0.925, recall 0.893, and mAP@0.5 of 0.934 [12]. YOLOv8n showed more prominent results, with precision 0.943, recall 0.928, and mAP@0.5 of 0.970 compared to Sportelli et al.'s precision of 0.926, recall of 0.926, and mAP@0.5 of 0.941 [12]. This indicates that even with the baseline configuration, the models in this study showed higher detection efficiency, likely influenced by differences in the optimizer and slightly larger batch size.

However, peak performance was achieved by the YOLOv11n model with the 'Custom Optuna' configuration, which resulted in precision 0.949, recall 0.944, and mAP@0.5 of 0.978. This model not only surpassed the performance of the other models in this study but also outperformed all results reported by Sportelli et al. [12]. This improvement is attributed to the successful hyperparameter tuning via the Optuna algorithm applied systematically. In the custom configuration, further tuning was focused on adjusting two main parameters from the default Optuna settings: the lr0 (initial learning rate), which was initially 0.00923, was adjusted to 0.0075, and the lrf (final learning rate factor), which was 0.01165, was adjusted to 0.01. These adjustments were made to control the model's convergence speed more stably, allowing for higher accuracy without sacrificing training efficiency.

Table 4. Comparison with SOTA on the 'Weeds' dataset

Model	Condition	P	R	F1-Score	mAP50
YOLOv5n	Sportelli et al [12]	0.925	0.893	0.934	0.925
	Without Tune	0.945	0.918	0.964	0.945
	Default Optuna	0.945	0.915	0.965	0.945
	Custom Optuna	0.941	0.922	0.965	0.941
YOLOv8n	Sportelli et al [12]	0.926	0.926	0.941	0.926
	Without Tune	0.943	0.928	0.970	0.943
	Default Optuna	0.930	0.917	0.963	0.930
	Custom Optuna	0.937	0.917	0.962	0.937
YOLOv11n	Without Tune	0.933	0.929	0.968	0.933
	Default Optuna	0.949	0.934	0.974	0.949
	Custom Optuna	0.949	0.944	0.978	0.949
YOLOv12n	Without Tune	0.926	0.908	0.960	0.926
	Default Optuna	0.943	0.914	0.966	0.943
	Custom Optuna	0.933	0.900	0.958	0.933

4. Conclusions

This study demonstrates that the YOLO nano variant, particularly YOLOv11n, has high potential in detecting weeds in turfgrass images using the public 'Weeds' dataset. Through the Optuna-based hyperparameter tuning approach, particularly with further modifications to the lr0 and lrf parameters, the YOLOv11n model achieved the best performance with an mAP@0.5 of 0.978, surpassing the results of previous studies. This confirms that the combination of a lightweight architecture and efficient tuning strategies can produce accurate and stable object detection. For future development, it is recommended to test the model's generalization on more diverse weed datasets, explore instance segmentation approaches, and integrate the model into edge computing systems such as drones or IoT devices to support real-world implementation in precision agriculture systems.

Acknowledgements

The author declares no conflict of interest in this research. Gratitude is extended to Universitas Dian Nuswantoro for providing institutional support and funding, which made the completion of this study possible.

Author Contributions

Candhy Fadhila Arsyad was responsible for the conception and design of the study, dataset preparation, model training, and manuscript drafting. Pulung Nurtantio Andono provided methodological supervision, validated the experimental design, and contributed to the interpretation of results. Moch Arief Soeleman contributed to the literature review, analysis refinement, and final review of the manuscript for submission readiness. All authors have read and agreed to the published version of the manuscript.

References

[1] J. Lisek, L. Sas-Paszt, A. Mika, and A. Lisek, "The Response of Weeds and Apple Trees to Beneficial Soil Microorganisms and Mineral Fertilizers Applied in Orchards," *Agronomy*, vol. 12, no. 11, 2022, doi: 10.3390/agronomy12112882.

[2] J. Lisek, "Diversity of Summer Weed Communities in

Response to Different Plum Orchard Floor Management in-Row," *Agronomy*, vol. 13, no. 5, 2023, doi: 10.3390/agronomy13051421.

[3] A. Arato, G. Mengesha, Z. Fentahun, and B. Borano, "Association of apple powdery mildew (Podosphaera leucotricha) and anthracnose (Neofabraea malicorticis) epidemics with environmental factors and agronomic practices in Chencha highlands, southern Ethiopia," *Phytopathol. Res.*, vol. 7, 2025, doi: 10.1186/s42483-024-00303-3.

[4] C. M. Badgujar, A. Poulose, and H. Gan, "Agricultural object detection with You Only Look Once (YOLO) Algorithm: A bibliometric and systematic literature review," *Comput. Electron. Agric.*, vol. 223, pp. 1–30, 2024, doi: 10.1016/j.compag.2024.109090.

[5] L. Yang, T. Zhang, S. Zhou, and J. Guo, "AAB-YOLO: An Improved YOLOv11 Network for Apple Detection in Natural Environments," *Agriculture*, vol. 15, no. 8, 2025, doi: 10.3390/agriculture15080836.

[6] F. M. Firnando, D. R. I. M. Setiadi, A. R. Muslikh, and S. W. Irianandа, "Analyzing InceptionV3 and InceptionResNetV2 with Data Augmentation for Rice Leaf Disease Classification," *J. Futur. Artif. Intell. Technol.*, vol. 1, no. 1, pp. 1–11, 2024, doi: 10.62411/faith.2024-4.

[7] R. K. Rachman, D. R. I. M. Setiadi, A. Susanto, K. Nugroho, and H. M. M. Islam, "Enhanced Vision Transformer and Transfer Learning Approach to Improve Rice Disease Recognition," *J. Comput. Theor. Appl.*, vol. 1, no. 4, pp. 446–460, 2024, doi: 10.62411/jcta.10459.

[8] J. Mei and W. Zhu, "BGF-YOLOv10: Small Object Detection Algorithm from Unmanned Aerial Vehicle Perspective Based on Improved YOLOv10," *Sensors*, vol. 24, no. 21, p. 6911, Oct. 2024, doi: 10.3390/s24216911.

[9] L. Nieradzik *et al.*, "WoodYOLO: A Novel Object Detector for Wood Species Detection in Microscopic Images," *Forests*, vol. 15, no. 11, p. 1910, Oct. 2024, doi: 10.3390/f15111910.

[10] X. Xi, Z. Huang, L. Ning, and Y. Zhang, "Real-time Pedestrian Detection Algorithm Based on Improved YOLOv3," *J. Phys. Conf. Ser.*, pp. 180–183, 2021, doi: 10.1088/1742-6596/2002/1/012075.

[11] S. Singha and B. Aydin, "Automated Drone Detection Using YOLOv4," *Drones*, vol. 5, no. 3, p. 95, Sep. 2021, doi: 10.3390/drones5030095.

[12] M. Sportelli *et al.*, "Evaluation of YOLO Object Detectors for Weed Detection in Different Turfgrass Scenarios," *Appl. Sci.*, vol. 13, no. 14, 2023, doi: 10.3390/app13148502.

[13] J. Luo, Z. Liu, Y. Wang, A. Tang, H. Zuo, and P. Han, "Efficient Small Object Detection You Only Look Once: A Small Object Detection Algorithm for Aerial Images," *Sensors*, vol. 24, no. 21, 2024, doi: 10.3390/s24217067.

[14] A. Startups, "Weeds Dataset," Roboflow Universe. [Online]. Available: https://universe.roboflow.com/augmented-startups/weeds-nxe1w

[15] J. A. O. S. Silva *et al.*, "Deep Learning for Weed Detection and Segmentation in Agricultural Crops Using Images Captured by an Unmanned Aerial Vehicle," *Remote Sens.*, vol. 16, no. 23, 2024, doi: 10.3390/rs16234394.

[16] M. Butt, N. Glas, J. Monsuur, R. Stoop, and A. de Keijzer, "Application of YOLOv8 and Detectron2 for Bullet Hole Detection and Score Calculation from Shooting Cards," *AI*, vol. 5, no. 1, pp. 72–90, 2024, doi: 10.3390/ai5010005.

[17] Q. Chen, Q. Xiong, H. Huang, and S. Tang, "An Efficient and Lightweight Surface Defect Detection Method for Micro-Motor Commutators in Complex Industrial Scenarios Based on the CLS-YOLO Network," *Electron.*, vol. 14, no. 3, pp. 1–26, 2025, doi: 10.3390/electronics14030505.

[18] J. Li, M. Zheng, D. Dong, and X. Xie, "PC-CS-YOLO: High-Precision Obstacle Detection for Visually Impaired Safety," *Sensors*, vol. 25, no. 2, pp. 1–15, 2025, doi: 10.3390/s25020534.

[19] R. Khanam and M. Hussain, "YOLOv11: An Overview of the Key Architectural Enhancements," vol. 2024, pp. 1–9, 2024, [Online]. Available: http://arxiv.org/abs/2410.17725

[20] S. Kaya, B. Kiraz, and A. Y. Çamurcu, "Hyperparameter Optimization in Deep Learning-Based Object Detection of Branching and Endpoints on 2D Brain Vessel Images," in *2024 Innovations in Intelligent Systems and Applications Conference (ASYU)*, Oct. 2024, pp. 1–6. doi: 10.1109/ASYU62119.2024.10757007.

[21] O. V. Johnson, C. Xinying, K. W. Khaw, and M. H. Lee, "ps-CALR: Periodic-Shift Cosine Annealing Learning Rate for Deep Neural Networks," *IEEE Access*, vol. 11, pp. 139171–139186, 2023, doi: 10.1109/ACCESS.2023.3340719.

[22] T. Zhang, B. Zhu, Y. Ma, and X. Zhou, "A Novel Image Encryption Algorithm Based on Multiple Random DNA Coding and Annealing," *Electron.*, vol. 12, no. 3, 2023, doi: 10.3390/electronics12030501.

[23] X. Guo *et al.*, "Citrus Disease Detection Based on Dilated Reparam Feature Enhancement and Shared Parameter Head," *Sensors*, vol. 25, no. 7, 2025, doi: 10.3390/s25071971.

[24] L. H. Lai *et al.*, "The Use of Machine Learning Models with Optuna in Disease Prediction," *Electron.*, vol. 13, no. 23, pp. 1–20, 2024, doi: 10.3390/electronics13234775.

[25] D. Şahin, O. Torkul, M. Şişci, D. D. Diren, R. Yılmaz, and A. Kibar, "Real-Time Classification of Chicken Parts in the Packaging Process Using Object Detection Models Based on Deep Learning," *Processes*, vol. 13, no. 4, 2025, doi: 10.3390/pr13041005.

[26] L. Song, H. Deng, J. Han, and X. Gao, "Improved Aerial Surface Floating Object Detection and Classification Recognition Algorithm Based on YOLOv8n," *Sensors*, vol. 25, no. 6, 2025, doi: 10.3390/s25061938.

[27] J. Liu, X. Liu, H. Chen, and S. Luo, "MDD-YOLOv8: A Multi-Scale Object Detection Model Based on YOLOv8 for Synthetic Aperture Radar Images," *Appl. Sci.*, vol. 15, no. 4, 2025, doi: 10.3390/app15042239.