SINERGI Vol. 29, No. 1, February 2025: 197-206
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2025.1.018

Performance evaluation of hyper-parameter tuning automation in YOLOV8 and YOLO-NAS for corn leaf disease detection

Huzair Saputra[1], Kahlil Muchtar[2]*, Nidya Chitraningrum[3], Agus Andria[4], Alifya Febriana[5]
[1]Master of Artificial Intelligence, Department of Informatics, Universitas Syiah Kuala, Indonesia
[2]Department of Electrical and Computer Engineering, Universitas Syiah Kuala, Indonesia
[3]Research Center for Biomass and Bioproducts, National Research and Innovation Agency, Indonesia
[4]The Central Bureau of Statistics (BPS) Aceh Tamiang, Aceh Tamiang Regency, Indonesia
[5]Department of Electrical Engineering, Yuan Ze University, Taiwan

Abstract
Corn cultivation was crucial in Southeast Asia, significantly contributing to regional food security and economies. However, leaf diseases posed a significant threat, causing substantial losses in production and quality. This research utilized artificial intelligence (AI) technology to address this issue by automating the hyper-parameter tuning process in YOLO (You Only Look Once) object detection models for early corn leaf disease detection. High-resolution images of corn leaves were captured and preprocessed for consistency. The preprocessing stage involved creating new dataset folders for images and labels, resizing images while preserving their aspect ratio, and rotating them if necessary. The images, containing 11,596 labeled instances, were analyzed using YOLOv8 and YOLO-NAS models. Each image's detected disease regions were converted into YOLO-format text files with x, y, width, and height coordinates, describing the presence and severity of infections. The models' performances were evaluated using precision, recall, mAP50, and mAP50-95 metrics. YOLOv8m achieved a mAP50 of 98.5% and mAP50-95 of 67.8%, while YOLO-NAS-L demonstrated superior detection capabilities with a mAP50 of 70.3% and mAP50-95 of 38.9%. This automated system facilitated early disease identification and enabled prompt preventive measures, thereby enhancing crop yields and mitigating losses. The findings highlighted the potential of advanced AI-driven detection systems in revolutionizing crop management and supporting global food security.

Keywords:
Corn leaf disease;
Hyper-parameter tuning;
Object detection;
YOLO-NAS;
YOLOv8;

Article History:
Received: March 29, 2024
Revised: July 26, 2024
Accepted: July 29, 2024
Published: January 4, 2025

Corresponding Author:
Kahlil Muchtar
Electrical and Computer Engineering Department, Universitas Syiah Kuala, Indonesia
Email: kahlil@usk.ac.id

INTRODUCTION

Agriculture is vital for feeding the growing global population, with corn (Zea mays) being a crucial crop. However, plant diseases pose a significant threat to modern agricultural productivity, jeopardizing the yield and quality of essential crops like corn [1]. Traditional detection methods are time-consuming and prone to errors, which increases economic losses. Machine learning, particularly deep neural networks, offers promising advancements in improving detection accuracy and efficiency [2].

Deep Learning, especially Convolutional Neural Networks (CNNs), excels in analyzing visual data, and detecting complex patterns with high precision. CNNs have been successful in numerous applications, including object detection and image segmentation, by learning hierarchical features from images [3][4]. Deep learning has also revolutionized fields like healthcare, where

CNN-based methods are used for analyzing health data, facilitating early diagnosis, and personalized treatment plans [5]. Pivkin et al. explore using neural networks to detect defects in direct metal deposition. They review relevant literature on neural networks and AI, affirming the importance of this research. The study focuses on three neural network models—U-Net, ResUNet, and VGG-16—commonly used in computer vision. After training and evaluating these models, they identify the most effective one for defect detection. The paper concludes by suggesting directions for future research in this area [6].

Hyper-parameter tuning in machine learning is essential for optimizing model performance. Effective tuning can significantly impact model accuracy and efficiency. While various optimization techniques exist, their effectiveness depends on the specific problem [7]. In computer vision, machine learning addresses challenges like large-scale images and complex object detection, with techniques like Maximally Stable Extremal Regions (MSER) used for vehicle detection and human pose estimation being pivotal for advancements in fields such as surveillance and sports analysis [8].

YOLO (You Only Look Once) models, including the recent YOLOv8 and YOLO-NAS, have enhanced real-time object detection with improved accuracy and speed [9][10]. YOLO-NAS, incorporating Neural Architecture Search (NAS), achieves superior performance compared to earlier YOLO versions, offering higher accuracy and faster processing times [11]. Figure 1 demonstrates the comparison of YOLO object detection models. The trend graph above illustrates that the YOLO-NAS architecture introduces the latest state-of-the-art (SOTA) with unparalleled speed and accuracy performance, surpassing other models such as YOLOv5, YOLOv6, YOLOv7, and YOLOv8. YOLO-NAS also provides the best evaluation results with a mean average precision (mAP) ~0.5 points more accurate and 10-20% faster than equivalent YOLOv8 variants, as evidenced in Table 1 [12].

Hyper-parameter tuning frameworks like Tune and Ray Tune are crucial for optimizing deep learning models. These frameworks facilitate efficient experimentation and adjustment of hyper-parameters, supporting various machine learning libraries and algorithms. Figure 2 provides an overview of the components of the Ray Tune framework [13]. This research focuses on comparing YOLOv8 and YOLO-NAS object detection methods for detecting pests and diseases on corn leaves, leveraging optimal hyper-parameters for each model.

In the era of the Fourth Industrial Revolution in machine learning, hyper-parameter tuning has become a commonly used technique to find the optimal combination of hyper-parameters in a model or algorithm [14]. This technique aims to improve accuracy, precision, recall, or other relevant evaluation metrics in the model and is highly useful in saving time and effort in manually searching for optimal hyperparameter configurations. Automated hyper-parameter configuration makes it easier to find hyper-parameter values that provide the best performance for automatic object detection system approaches using CNN [15].

Table 1. Comparison of YOLO-NAS and YOLOv8 in terms of mAP and latency

Model	mAP	Latency (milliseconds)
YOLO-NAS S	47.5	3.21
YOLO-NAS M	51.55	5.85
YOLO-NAS L	52.22	7.87
YOLO-NAS S INT- 8	47.03	2.36
YOLO-NAS M INT- 8	51	3.78
YOLO-NAS L INT- 8	52.1	4.78
YOLOv8 S	47	4
YOLOv8 M	50	6.5
YOLOv8 L	52.5	9.5

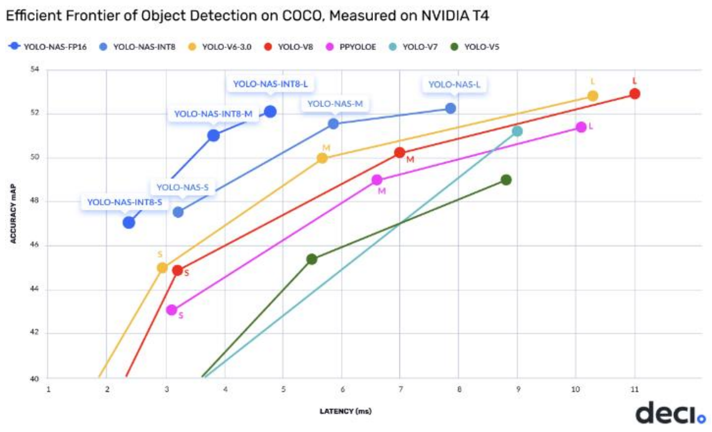


Figure 1. Comparison of YOLO object detection models [11]

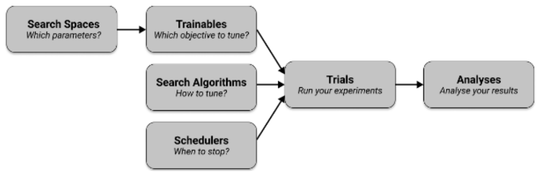


Figure 2. Ray Tune framework [13]

Park et al. developed HyperTendril, a visual analytics framework designed for user-driven hyperparameter optimization of deep neural networks, addressing the challenge of manually tuning hyperparameters. Despite the advances in automated machine learning (AutoML) methods for searching optimal hyperparameters, their effectiveness is hindered by dependency on initial configurations. HyperTendril allows users to iteratively refine search spaces and AutoML configurations through interactive tuning, offering insights into hyperparameter behaviors and variable importance analysis. Evaluation through longitudinal user research highlighted its efficacy in a professional setting [16]. Similarly, Dou et al. introduced HyperTuner, a cross-layer multi-objective hyperparameter auto-tuning framework aimed at optimizing both model hyperparameters and system parameters to balance accuracy, training time, and energy consumption. HyperTuner utilizes the ADUMBO algorithm to find Pareto-optimal configurations by selecting the most promising configuration through an adaptive uncertainty metric. Experimental results on a local distributed TensorFlow cluster demonstrated HyperTuner's superior convergence and diversity compared to other baseline algorithms, showing its adaptability across various data analytic service scenarios [17].

Hyper-parameter optimization plays a pivotal role in enhancing the performance of machine learning models, particularly in object detection frameworks. Gradient descent algorithms, including variations with momentum terms, have been extensively reviewed for their efficiency in optimizing learning processes. Neural networks have also seen innovations in activation functions, such as the Swish activation function, which operates as a self-gated mechanism to improve learning outcomes. Additionally, Gaussian processes have emerged as a versatile framework for a variety of machine learning applications, offering powerful predictive capabilities [18]. Within the realm of object detection, the YOLO (You Only Look Once) framework stands out for its unified, real-time detection capabilities, which streamline the detection process and enhance performance metrics across various datasets.

METHOD

Dataset

In this research, we utilized an open-source dataset from the Kaggle repository, specifically the 'Corn Leaf Infection Dataset' provided by Acharya [19]. This dataset comprises 4,225 images with a high resolution of 3,456 x 4,608 pixels. The images are categorized into two classes: 'healthy' and 'infected,' as outlined in Table 2. The testing data will further validate the model's performance using real-world data from The Central Bureau of Statistics (BPS) Aceh Tamiang, ensuring its applicability beyond the training dataset.

The dataset consists of 2,225 images of infected corn leaf samples stored in a dedicated folder for training and validation purposes. Each image is accompanied by a corresponding label indicating whether the leaf is infected or not. Additionally, the dataset contains 11,596 bounding box annotations, meticulously outlining the location of infected areas within the images, which aids in precise object detection. The dataset exhibits an imbalance, with 2,000 images labeled as 'healthy' and 2,225 images labeled as 'infected'. An explanation file accompanies the dataset, containing bounding box coordinates for the infected images. Overall, this dataset is a valuable resource for training models to detect and accurately classify infected corn leaves.

Methods

The methodological approach employed in this research is illustrated in Figure 3, providing an overview of the research stages. The dataset of infected corn leaf samples is organized within a dedicated folder, potentially containing images of corn leaves along with corresponding labels indicating infection status. The dataset comprises a total of 2,225 images and 2,225 labels.

The preprocessing stage involves preparing the dataset for training by conducting various operations on the data. Specifically focusing on the infected category, preprocessing steps are implemented to address images of infected corn leaves. During the data preprocessing pipeline, a series of essential operations are applied to ensure proper formatting of input images for subsequent tasks, particularly in computer vision or deep learning. This mechanism begins by creating the necessary directories for storing images and their corresponding labels. It first sets up the `images` and `labels` subfolders within the new dataset directory. Each image from the original dataset is then read and converted to RGB format. If needed, images are rotated 90 degrees clockwise to ensure consistent orientation.

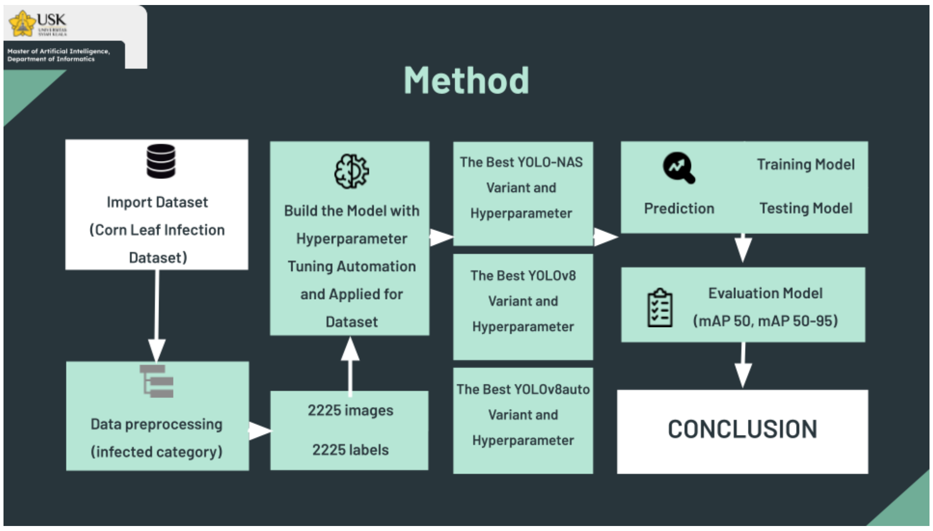

USK
Master of Artificial Intelligence, Department of Informatics
Method
Import Dataset (Corn Leaf Infection Dataset)
Build the Model with Hyperparameter Tuning Automation and Applied for Dataset
The Best YOLO-NAS Variant and Hyperparameter
The Best YOLOv8 Variant and Hyperparameter
The Best YOLOv8auto Variant and Hyperparameter
Prediction
Training Model
Testing Model
Evaluation Model (mAP 50, mAP 50-95)
Data preprocessing (infected category)
2225 images
2225 labels
CONCLUSION

Figure 3. Proposed Method

The images are resized to a maximum dimension of 854 pixels while maintaining their aspect ratio. After resizing, the images are saved in the newly created `images` folder. Simultaneously, annotations are processed by scaling bounding box coordinates relative to the new image dimensions. Each bounding box is adjusted to fit within the normalized range of 0 to 1, with coordinates (x, y) and dimensions (width, height) converted accordingly. Invalid bounding boxes are flagged if they fall outside the valid range, with any issues highlighted visually. The processed labels are saved in YOLO format, and the bounding boxes of the infected class are updated. This pipeline homogenizes input data, addressing potential image orientation and size variations from diverse sources [20].

The next step involves determining static hyper-parameters to be set in developing each model that will undergo tuning. The key hyper-parameters are 'Epoch,' 'Image Size,' 'Batch Size,' and 'Workers,' as outlined in Table 3.

Hyper-parameter tuning is crucial in optimizing model performance and preventing over-fitting or under-fitting. It entails experimenting with various hyper-parameter values and evaluating their outcomes to discover combinations that yield the best validation or test data results. Ray Tune is a framework that will be utilized as a supportive tool for conducting hyper-parameter tuning. The hyper-parameters used in several deep-learning algorithms are outlined in Table 4.

The variables and values in Table 4 refer to the sampling functions provided by the Tune library, which will be used for hyperparameter optimization. These functions generate random numbers within the specified minimum and maximum values range. The hyper-parameter values listed above will be randomly sampled and then utilized within the respective functions of the YOLOv8 and YOLO-NAS models. By executing the training scripts using these six parameter values, it is anticipated that the models can be trained to achieve the best accuracy.

In the proposed method, YOLOv8, YOLOv8auto, and YOLO-NAS are employed to optimize the detection of infected corn leaves through different approaches. YOLOv8 is used as a base model for object detection, aimed at detecting and locating infected corn leaves in images. This model undergoes extensive hyper-parameter tuning to optimize its performance. YOLOv8 uses a range of values for key hyper-parameters such as learning rate (lr0), momentum, and weight decay. The tuning process involves sampling these values within specified ranges to find the optimal combination for high detection accuracy.

Table 2. Corn leaf image data set

Label	Total	
Healthy	2,000 images	0 annotations
Infected	2,225 images	11,596 annotations

Table 3. Static hyper-parameters are used to train all models

Tuning Model		Best Model	
Hyper-parameter	**Value**	**Hyper-parameter**	**Value**
Epoch	10	Epoch	100
Image Size	640	Image Size	640
Batch Size	8	Batch Size	8
Workers	8	Workers	8

Table 4. Search space for automatic hyper-parameter tuning of each model.

Model	Hyper-parameter Tuning
YOLOv8	"optimizer": "AdamW", "lrf": (0.01, 1.0), "lr0": (1e-5, 1e-1), "momentum": (0.6, 0.98), "cls": (0.2, 4.0), "weight_decay": (0.0, 0.001)
YOLOv8auto	"optimizer": "AdamW", "lrf": (0.01, 1.0), "lr0": 0.01, "momentum": 0.937, "cls": (0.2, 4.0), "weight_decay": (0.0, 0.001)
YOLO-NAS	"optimizer": ["AdamW", "SGD"], "cosine_final_lr_ratio": (0.01, 1.0), "initial_lr": (1e-5, 1e-1), "momentum": (0.6, 0.98), "classification_loss_weight": (0.2, 4.0), "weight_decay": (0.0, 0.001)

YOLOv8auto, on the other hand, is designed to streamline the hyper-parameter tuning process by incorporating fixed values for certain hyper-parameters while still allowing for tuning within specified ranges for others. This variant aims to simplify and speed up the optimization process. Unlike YOLOv8, YOLOv8auto has fixed values for lr0 (set to 0.01) and momentum (set to 0.937). This means that these hyper-parameters do not undergo a range-based tuning process. Instead, the focus is on tuning other parameters within specified ranges. This approach reduces the complexity of tuning and leverages predefined values that are expected to perform well.

YOLO-NAS (Neural Architecture Search) is utilized to automatically explore different neural network architectures along with hyper-parameter tuning. This variant aims to find the best model architecture that yields optimal performance in detecting infected corn leaves. YOLO-NAS expands the search space to include various model architectures in addition to hyper-parameters. It uses different optimizers (e.g., AdamW, SGD) and has a broader range for parameters like learning rate and momentum. This comprehensive approach allows YOLO-NAS to potentially discover more effective model configurations compared to traditional tuning methods.

The final stage involves evaluating the performance of the trained models using various evaluation metrics to measure their effectiveness. These metrics include precision, recall, F1-score, IoU, and mAP. mAP (mean Average Precision) is a commonly used evaluation metric in object detection tasks. "mAP 50" refers to the average precision at an IoU (Intersection over Union) threshold of 0.5, while "mAP 50-95" represents the average precision across IoU thresholds ranging from 0.5 to 0.95. These scores provide a measure of how well the model performs in detecting infected corn leaves at various levels of precision.

The formulas for calculating precision, recall, F1-score, IoU, and mAP are presented in (1)-(5) [21, 22, 23].

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

$$IoU = \frac{TP}{(TP + FP + FN)} \tag{4}$$

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{|TP_c|}{|FP_c| + |TP_c|} \tag{5}$$

RESULTS AND DISCUSSION

Experiment Result

All variants of the YOLOv8 model were executed using the integrated Ray Tune framework within the Tune library. Ultralytics YOLOv8 leveraged Ray Tune for hyper-parameter tuning, simplifying the optimization process for the YOLOv8 model. Ray Tune facilitated advanced search strategies, parallelism, and early stopping to accelerate the tuning process. Both the YOLOv8 and YOLOv8auto models were separately run based on the predefined hyper-parameters outlined in Table 3 and Table 4.

During the automated hyper-parameter tuning phase, the YOLOv8auto variants were labeled '1', while the YOLOv8 variants were labeled '2'. Consequently, these models were identified as YOLOv8n1, YOLOv8l1, YOLOv8m1, YOLOv8s1, and YOLOv8x1 for the YOLOv8auto variants, and YOLOv8n2, YOLOv8l2, YOLOv8m2, YOLOv8s2, and YOLOv8x2 for the YOLOv8 variants. The results of the automated hyper-parameter tuning process are reflected in Figure 4, illustrating the relative performance among the model variants.

The determination of these hyper-parameter values was automated by Ray Tune using a method with 10 iterations and 10 epochs per iteration for each variant. The primary objective was to achieve the best mAP50-95 values for each model variant. The experimental results show that the YOLOv8auto model variants, on average, outperformed the YOLOv8 models. This phenomenon occurred because the tuning process focused on two hyper-parameter values, namely "lr0" and "momentum," as listed in Table

4. The experiments prove that hyper-parameter configurations significantly impact the model, influencing the learning process and enhancing training effectiveness [3].

Using the Ray Tune method, the algorithm for determining the best results from various YOLOv8 models involves selecting outcomes based on the mAP50-95 metric with the maximum mode. Subsequently, the best YOLOv8 model is run, adjusting the 'epoch' value according to Table 3 before training. Post-training, the model undergoes evaluation using validation data, extracting metrics like precision, recall, mAP50, and mAP50-95 from the evaluation results. The evaluation metrics obtained after training the infected leaf dataset are presented in Table 5. A higher mAP score signifies better accuracy in detecting infected leaves [24]. The performance evaluation results of the five best YOLOv8 model variants were assessed based on several metrics, including precision, recall, mAP50, and mAP50-95, thus providing a comprehensive understanding of the relative performance of each model.

YOLOv8x stands out with a precision of 96.3%, recall of 94.8%, mAP50 of 98.3%, and mAP50-95 of 66.4%. However, YOLOv8m is the superior variant, achieving the highest performance with the precision of 96.3%, recall of 95.2%, mAP50 of 98.5%, and mAP50-95 of 67.8%. This variant can be considered the optimal choice based on the results of hyper-parameter tuning, demonstrating significant improvement across various aspects of model performance.

In a comprehensive review, YOLOv8m emerges as the most superior model, attaining the highest accuracy across all evaluative metrics. With balanced precision, recall, and mAP50, along with a high mAP50-95 score, YOLOv8m becomes a robust choice for object detection tasks in this context. These results indicate that the hyper-parameter settings applied to YOLOv8m r significantly enhance the model's capabilities, making it the optimal choice for this object detection task.

Unlike before, the YOLO-NAS model is executed without utilizing a framework, but rather, the hyper-parameter values are randomly selected using a script, following the values listed in Table 3 and Table 4. The hyper-parameter tuning process commences by iterating through each predetermined variant of the YOLO-NAS model. Upon completing training, the hyper-parameter tuning process for the next variant is repeated until all variants have been processed. This stage aims to find the hyper-parameter combinations that yield the best results for the YOLO-NAS model. The outcomes of this hyper-parameter tuning process are observed in Figure 5.

The determination of hyper-parameter values is carried out using a method involving 10 iterations and 10 epoch repetitions per iteration for each variant. The best 'yolo_nas_l' variant is found in the 1st iteration, the best 'yolo_nas_m' variant is found in the 6th iteration, and the best 'yolo_nas_s' variant is found in the 10th iteration.

After evaluating the model performance using evaluation metrics, the next step involves comparing the pest detection results obtained by predicting on the raw infected corn leaf images, totaling 500 images. These results serve as a supporting factor in determining the best model. Next, the average detection values obtained from predictions on raw images are visualized in the comparison of mAP and average detection values between YOLOv8 and YOLO-NAS in Figure 6. The average detection results for each variant are calculated based on the predictions made on the 500 images.

After obtaining the best variants for each iteration, the value of 'EPOCHS' is increased to 100 for each best variant. This process aims to enhance the accuracy and performance of the model by providing more time for the learning process and parameter adjustment. By increasing the number of epochs, the model is expected to have more opportunities to adjust weights and improve prediction quality. After rerunning the script with the updated epoch values, evaluation metrics are generated for each best variant of the YOLO-NAS model, as presented in Table 6. The presented results show a significant comparison between the YOLOv8 and YOLO-NAS models after the hyperparameter tuning process. Particularly, YOLOv8m from the YOLOv8 model group stands out with excellent performance, exhibiting higher precision, recall, mAP50, and mAP50-95 values than other variants.

Table 5. Performance comparison of YOLOv8 model variants after hyper-parameter tuning

Model	**P (%)**	**R (%)**	**F1 (%)**	**mAP50 (%)**	**mAP50-95 (%)**
YOLOv8x	96,3	94,8	95.6	98,3	66,4
YOLOv8l	95,7	93,9	94.8	97,9	65,2
YOLOv8m	**96,3**	**95,2**	**95.7**	**98,5**	**67,8**
YOLOv8s	93,3	91,5	92.5	96,6	58,9
YOLOv8n	87,3	82,8	84.9	91,0	52,1

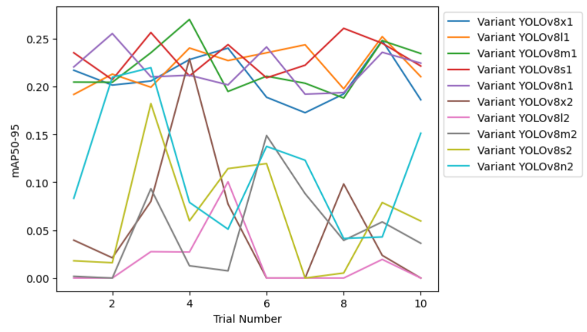


Figure 4. Graph of hyper-parameter tuning automation results on the YOLOv8 model variant

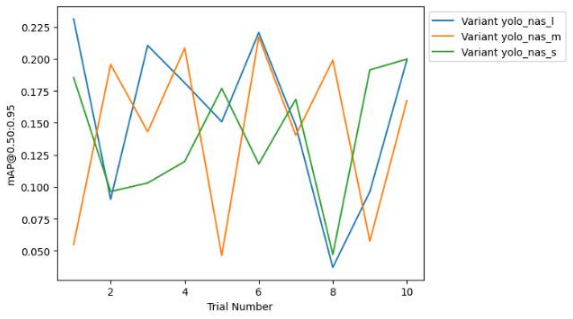


Figure 5. Graph of hyper-parameter tuning results on the YOLO-NAS model variant

Table 6. Performance comparison of YOLO-NAS model variants after hyper-parameter tuning

Model	P (%)	R (%)	F1 (%)	mAP50 (%)	mAP50-95 (%)
yolo-nas-l	**62,8**	**71,6**	**66,9**	**70,3**	**38,9**
yolo-nas-m	57,4	67,7	62,1	63,5	34,1
yolo-nas-s	58,9	59,3	59,0	59,9	31,3

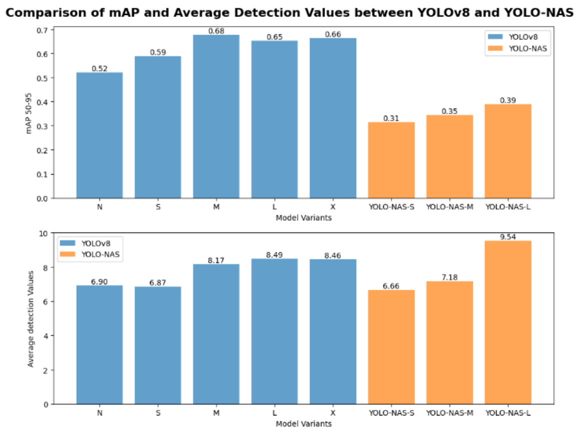


Figure 6. Comparison of mAP and average detection values between YOLOv8 and YOLO-NAS

On the other hand, in terms of YOLO-NAS, the YOLO-NAS-L variant demonstrates the best results, with higher precision, recall, mAP50, and mAP50-95 values than other variants after undergoing the hyper-parameter tuning process.

YOLOv8m showed superior performance in mAP50-95 with a score of 0.68, while YOLO-NAS-L has a higher average detection value, with a score of 9.54. YOLO-NAS variant L outperforms all other models in terms of average detection value. This suggested that YOLO-NAS L might be the most effective model for tasks that require high detection rates. Through this evaluation, it could be concluded that YOLOv8m and YOLO-NAS-L were the best choices for detecting corn leaf disease. Both models have their respective advantages and can be selected based on certain priorities in object detection tasks.

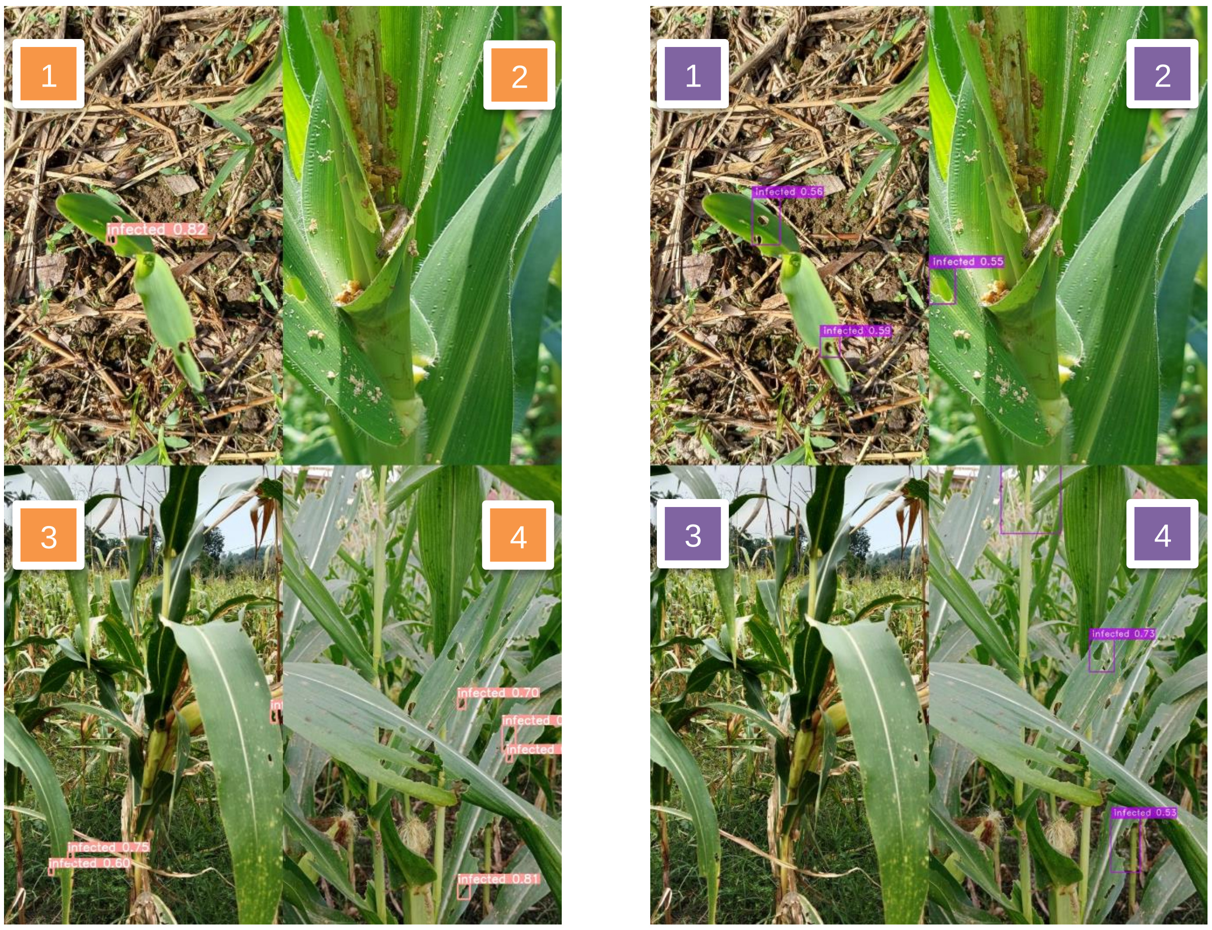
Discussion

The researchers evaluated the performance of the YOLOv8 model without conducting the hyper-parameter tuning process, which was then compared with the YOLOv8 model undergoing hyper-parameter tuning. This evaluation result provided insights into how well these models can perform with default hyper-parameter values [25], as presented in Table 7.

Furthermore, we conducted experiments on a new dataset using four images of infected corn leaves collected from BPS Aceh Tamiang regency. The predicted results of the YOLOv8 and YOLO-NAS models as mentioned in Figure 7. The results showed a comparison between the real differences in their detection capabilities.

While YOLOv8 showed superior accuracy in annotating images with minimal false positives, YOLO-NAS showed a unique ability to detect images in the top right corner (image number 2). However, it failed to detect the image in the bottom left corner (image number 3), a task that YOLOv8 completed.

Additionally, YOLO-NAS produces one false positive annotation (background) in the bottom right corner (image number 4). Minimizing false positives is critical to improving the precision and effectiveness of detection models, as these detection errors could lead to inaccuracies in model predictions and reduced overall model reliability. But if the task requires detecting specific or unique objects, YOLO-NAS might be more suitable despite the risk of false positives.



(a) (b)

Figure 7. YOLOv8 predictions (a) and YOLO-NAS predictions (b) on the new dataset from BPS Aceh Tamiang

Table 7. Comparison of YOLOv8 and YOLOv8-tuned models

Model	Non-Tuned			Tuned		
	Image Size	mAP50 (%)	mAP50-95 (%)	Image Size	mAP50 (%)	mAP50-95 (%)
YOLOv8x	416	96.5	**72.7**	640	983	664
YOLOv8l	416	95.3	69.4	640	97.9	65.2
YOLOv8m	416	94.7	66.6	640	**98.5**	67.8
YOLOv8s	416	90.9	60.2	640	96.6	58.9
YOLOv8n	416	79.2	47.0	640	91.0	52.1

CONCLUSION

Based on the results of the performance evaluation and comparison between the YOLOv8 and YOLO-NAS models that have undergone the hyper-parameter tuning process, it can be concluded that YOLOv8m from the YOLOv8 model group and YOLO-NAS-L from the YOLO-NAS model group stand out as the best choices for detecting corn leaf diseases. YOLOv8m performed better superior performance in mAP50 and mAP50-95, while YOLO-NAS-L has higher detection results. Both models have their respective advantages and can be selected based on specific priorities in the object detection task. This research significantly contributes to increase the efficiency and accuracy of early pest detection in corn plantations by integrating artificial intelligence (AI) in agriculture. By using advanced and optimized object detection models, farmers can identify corn foliar diseases more quickly and accurately, enabling them to take appropriate preventive or control measures promptly. It helped to increase crop yields, reduced losses due to crop diseases, and increased agricultural productivity overall.

Future researchers' endeavors could concentrate on refining methods and techniques to enhance further model performance in detecting corn leaf diseases. Exploring multi-source data integration or incorporating IoT sensors to bolster real-time plant disease detection systems presents another avenue for investigation. Developing object detection models adept at handling variations in light conditions, plant textures, and environmental backgrounds could also be an intriguing research focus. By persistently advancing and refining plant disease detection technology, it is envisaged to make a more substantial contribution to enhancing food security and farmer welfare on a global scale.

ACKNOWLEDGMENT

This research was supported by National Research and Innovation Agency (BRIN) Indonesia. In addition, we thank our colleagues from The Central Bureau of Statistics (BPS) Aceh Tamiang who provided insight and expertise that greatly assisted this research. This research is supported under master thesis research grant USK, with Grant No. 408/UN11.2.1/PG.01.03/SPK/PTNBH/2024.

REFERENCES

[1] A. Tripathi, V. Gohokar, and R. Kute, "Comparative Analysis of YOLOv8 and YOLOv9 Models for Real-Time Plant Disease Detection in Hydroponics," *Engineering, Technology & Applied Science Research*, vol. 14, no. 5, pp. 17269–17275, Oct. 2024, doi: 10.48084/etasr.8301.

[2] M. Haque and J. Adolphs, "Corn Leaf Disease Classification and Detection using Deep Convolutional Neural Network," *Research Project Final Report*, Aug. 2021. doi: 10.13140/RG.2.2.20819.50722.

[3] L. Chau and H. Nguyen-Xuan, "Deep learning for computational structural optimization," *ISA Transactions*, vol. 103, pp. 177–191, Mar. 2020, doi: 10.1016/j.isatra.2020.03.033.

[4] H. Amin, A. Darwish, A. E. Hassanien, and M. Soliman, "End-to-End Deep Learning Model for Corn Leaf Disease Classification," *IEEE Access*, vol. 10, pp. 31103–31115, 2022, doi: 10.1109/ACCESS.2022.3159678.

[5] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Computer Science*, vol. 2, no. 6, p. 420, Nov. 2021, doi: 10.1007/s42979-021-00815-1.

[6] P. M. Pivkin, N. Khodanovich, S. N. Grigoriev, and P. Peretyagin, "Method for detecting defects in direct metal deposition using a neural network," in *Laser + Photonics for Advanced Manufacturing*, S. Lecler, W. Pfleging, and F. Courvoisier, Eds., SPIE, Jun. 2024, p. 80. doi: 10.1117/12.3022863.

[7] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/j.neucom.2020.07.061.

[8] A. Khan, A. Laghari, and S. Awan, "Machine Learning in Computer Vision: A Review," *ICST Transactions on Scalable Information Systems*, p. 169418, Apr. 2021, doi: 10.4108/eai.21-4-2021.169418.

[9] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Jan. 2023. Accessed: Jun. 01, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[10] Ultralytics, "Ultralytics Documentation," 2023. Accessed: Jun. 01, 2023. [Online]. Available: https://docs.ultralytics.com

[11] S. Aharon *et al.*, "Super-Gradients," 2021, *GitHub*. doi: 10.5281/zenodo.7789328.

[12] Augmented Startups, "YOLO NAS vs YOLOv8: A Comprehensive Comparison," 2023. Accessed: Jul. 15, 2023. [Online]. Available: https://www.augmentedstartups.com/blog/yolo-nas-vs-yolov8-a-comprehensive-comparison

[13] The Ray Team, "Key Concepts of Ray Tune," 2023. Accessed: Jan. 14, 2024. [Online]. Available: https://docs.ray.io/en/latest/tune/key-concepts.html

[14] S. Shekhar, A. Bansode, and A. Salim, "A Comparative study of Hyper-Parameter Optimization Tools," in *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, IEEE, Dec. 2021, pp. 1–6. doi: 10.1109/CSDE53843.2021.9718485.

[15] F. Johnson, A. Valderrama, C. Valle, B. Crawford, R. Soto, and R. Nanculef, "Automating Configuration of Convolutional Neural Network Hyperparameters Using Genetic Algorithm," *IEEE Access*, vol. 8, pp. 156139–156152, 2020, doi: 10.1109/ACCESS.2020.3019245.

[16] H. Park, Y. Nam, J.-H. Kim, and J. Choo, "HyperTendril: Visual Analytics for User-Driven Hyperparameter Optimization of Deep Neural Networks," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1407–1416, Feb. 2021, doi: 10.1109/TVCG.2020.3030380.

[17] H. Dou, S. Zhu, Y. Zhang, P. Chen, and Z. Zheng, "HyperTuner: a cross-layer multi-

objective hyperparameter auto-tuning framework for data analytic services," *The Journal of Supercomputing*, vol. 80, no. 12, pp. 17460–17491, Aug. 2024, doi: 10.1007/s11227-024-06123-8.

[18] T. Yu and H. Zhu, "Hyper-Parameter Optimization: A Review of Algorithms and Applications," Mar. 2020, doi: 10.48550/arXiv.2003.05689.

[19] R. Acharya, "Corn Leaf Infection Dataset, Version 1," Oct. 2020. Accessed: Jun. 01, 2023. [Online]. Available: https://www.kaggle.com/qramkrishna/corn-leaf-infection-dataset

[20] L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "Normalization Techniques in Training DNNs: Methodology, Analysis and Application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 10173–10196, Aug. 2023, doi: 10.1109/TPAMI.2023.3250241.

[21] Moh. Khairudin *et al.*, "Early detection of diabetes potential using cataract image processing approach," *SINERGI*, vol. 28, no. 1, p. 55, Dec. 2023, doi: 10.22441/sinergi.2024.1.006.

[22] A. Maulana *et al.*, "Performance Analysis and Feature Extraction for Classifying the Severity of Atopic Dermatitis Diseases," in *2023 2nd International Conference on Computer System, Information Technology, and Electrical Engineering (COSITE)*, IEEE, Aug. 2023, pp. 226–231. doi: 10.1109/COSITE60233.2023.10249760.

[23] S. Susanto, D. Stiawan, M. A. S. Arifin, Mohd. Y. Idris, and R. Budiarto, "Effective and efficient approach in IoT Botnet detection," *SINERGI*, vol. 28, no. 1, p. 31, Dec. 2023, doi: 10.22441/sinergi.2024.1.004.

[24] S.-J. Hong *et al.*, "Moth Detection from Pheromone Trap Images Using Deep Learning Object Detectors," *Agriculture*, vol. 10, no. 5, 2020, doi: 10.3390/agriculture10050170.

[25] N. Chitraningrum *et al.*, "Comparison Study of Corn Leaf Disease Detection based on Deep Learning YOLO-v5 and YOLO-v8," *Journal of Engineering and Technological Sciences*, vol. 56, no. 1, pp. 61–70, Feb. 2024, doi: 10.5614/j.eng.technol.sci.2024.56.1.5.