

Network Reduction Strategy on YOLOv8 Model for Mango Leaf Disease Detection

I Gede Khresna Adi Wedanta Beratha^{*1}, Ni Putu Sutramiani², Ni Kadek Ayu Wirdiani³

Department of Information Technology, Faculty of Engineering, Udayana University

Jl. Raya Kampus Unud, Bukit Jimbaran, Badung 80361, Bali, Indonesia

Email: ¹khresnaadi15@gmail.com, ²sutramiani@unud.ac.id, ³ayuwirdiani@unud.ac.id

Abstrak. *Strategi Reduksi Jaringan pada Model YOLOv8 untuk Deteksi Penyakit Daun Mangga.* Deteksi penyakit pada daun mangga merupakan langkah penting dalam menjaga kesehatan tanaman dan meningkatkan produktivitas pertanian karena daun adalah salah satu bagian vital yang terlibat dalam proses fotosintesis dan pertumbuhan tanaman. Penyakit yang menyerang daun mangga dapat menyebabkan kerusakan yang menghambat pertumbuhan tanaman, sehingga pengembangan sistem deteksi yang akurat dan efisien sangat penting untuk membantu petani dalam mengidentifikasi dan mengatasi masalah ini sejak dini. Tujuan dari penelitian ini adalah untuk mengembangkan model deteksi penyakit pada daun mangga menggunakan model YOLOv8 yang dioptimalkan dengan network reduction strategy. Data yang digunakan terdiri dari gambar daun mangga dengan empat kelas penyakit dan satu kelas daun sehat yang bersumber dari Kaggle. Hasil penelitian menunjukkan bahwa model YOLOv8 yang dioptimalkan dapat menghasilkan model dengan kompleksitas rendah tanpa mengorbankan performa model. Model yang dioptimalkan dengan network reduction mencapai nilai mAP50-95 tertinggi sebesar 0,988, melebihi model dasar sebesar 0,3%.

Kata Kunci: daun mangga, deteksi, pertanian, network reduction, YOLOv8

Abstract. *Detecting diseases on mango leaves is a crucial step in maintaining plant health and enhancing agricultural productivity, considering that leaves are one of the vital parts involved in the photosynthesis process and plant growth. Diseases that affect mango leaves can cause damage that hinders the growth of the plants, making the development of an accurate and efficient detection system essential to assist farmers in identifying and addressing these issues early on. The objective of this research is to develop a disease detection model for mango leaves using the YOLOv8 model optimized with a network reduction. The data used consists of images of mango leaves with four classes of diseases. The results of the study indicate that the optimized YOLOv8 model can produce a model with low complexity without compromising model performance. The model optimized with network reduction achieved the highest mAP50-95 value of 0.988, surpassing the baseline model by 0.3%.*

Keywords: agriculture, detection, mango leaves, network reduction, YOLOv8

1. Introduction

Mango tree (*Mangifera indica*) is one of the tropical plants whose cultivation has been widely carried out in several countries other than Southeast Asia. Mango is a fruit that is rich in antioxidants, vitamin C, and vitamin E [1]. According to Badan Pusat Statistik Indonesia can produce up to 3.3 million tons of mangoes every year [2]. Mango production in Indonesia experienced a decline from 2022, which initially produced 3,308,894.7 tons, to 3,302,619.7 tons in 2023 [2]. One of the factors was the disease that attacked the mango tree. Farmers generally only use their senses and chemical control to detect and prevent disease. This method is quite time-consuming and prone to errors. With the advancement of technology today, detecting diseases in plants can be done faster using computer vision.

The application of computer vision in detecting plant diseases, especially mangoes, has been widely developed. Most studies use machine learning models without employing optimization strategies. One of the previous studies on plant disease detection was conducted using the YOLOv4 model to detect diseased mango leaves with relatively high detection accuracy results at 98% for predicting diseased leaves and reaching 100% for predicting healthy leaves [3].

Krishnamoorthy et al. [4] proposed a deep neural network with transfer learning for early detection of rice leaf diseases. This study combines CNN and transfer learning using several models like VGG-16, ResNet50, and InceptionV3. Tiara Sari et al. [5] apply a Convolutional Neural Network (CNN) to detect dry corn kernels using seven convolutional layers. The results show that the detection accuracy varies between 80% and 100%, depending on the size and position of the image when it was taken. Another study used YOLOv5 to detect tomato plant phenotypes, including fruits, flowers, and plant nodes [6]. This study confirmed that the YOLOv5 model can handle the challenges of object detection in plant images, such as size variation and color similarity between objects. Another study using MobileNet as a Convolutional Neural Network (CNN) architecture for disease classification in potato leaves, including early blight, late blight, and healthy [7]. The best results were achieved with the RMSprop optimizer, a learning rate of 0.0001, and 50 epochs, which resulted in an accuracy of 97.9% and a loss of 0.0390. Another study used a Convolutional Neural Network (CNN) architecture to detect diseases in corn leaves, with three classes, namely healthy, common rust, leaf blight, and leaf spot [8]. This study shows that the CNN model can detect corn leaf diseases well.

The use of the YOLO algorithm has begun to show variations in its use, from classifying fruit types, detecting fruit ripeness, to identifying plant diseases. Agustina and Sukron [9] proposed an Android-based YOLO algorithm to detect the ripeness level of papaya fruit. This study produced a model with an accuracy value of 93%, a precision of 94%, and a recall of 93%. A similar study used YOLOv5 to detect chili plant diseases with a variation in data splitting scenarios [10]. This experiment showed that the 80:10:10 data sharing produced the best performance with a precision of 94.6%, a recall of 93.6%, and an mAP of 95.9%. Mamun et al. [11] compared the performance of YOLOv5 and YOLOv8 in detecting grape leaf diseases through a mobile application. Results of this study showed that YOLOv8 had superior performance with 99.9% precision and 100% recall compared to YOLOv5 in performing the task of detecting grape leaf diseases. Another study proposed YOLOv4 to determine the type of mango plant through leaf images [12]. This study managed to achieve a high precision and a recall of 95%. Another similar study used YOLOv5 to detect fresh and rotten fruit [13]. This study shows that the YOLOv5 model is very effective in classifying fruit objects with a high accuracy of up to 84%. Aras et al. proposed the YOLOv5 model to detect the ripeness of tomatoes [14]. This study managed to achieve an accuracy value of 73%. The lack of variation in the dataset influences the accuracy results. Overall, these studies show that the YOLO algorithm, especially the YOLOv5 and YOLOv8 versions, show excellent results in object detection in various applications.

The YOLO optimization strategy has been proven as one of the solutions to increase efficiency, reduce complexity, and optimize object detection accuracy in various fields. Piarsa et al. [15] applied the network reduction and ensemble learning strategies to YOLOv8 to increase the efficiency of blood cell detection. This study produced a YOLOv8-60 model with a network reduced to 60% which was able to outperform the YOLOv8-baseline performance by 2.3%. A similar study used the YOLOv5 model with network scaling to detect Balinese carving motifs through the development of the Balicarv dataset, which includes 2,372 images with 8,817 annotations in seven classes of carving motifs [16]. This study applies three scenarios with three resolution variations, namely 320, 640, and 1280 pixels, and network depth settings at 30% (YOLOv5-30) and 60% (YOLOv5-60). The result is that the Ensemble-YOLOv5-60-NMS model with 60% Network Scaling can outperform the baseline YOLOv5 model by 1.4%.

In this study, we proposed a network reduction strategy to detect mango leaf disease. The proposed approaches aim to produce a lighter YOLOv8 model by reducing its network depth while maintaining its performance. This innovation not only contributes to the efficiency of disease detection in mango plants but also offers a practical solution that can be directly applied to agricultural practices. By optimizing the model for faster processing and lower computational cost, it can be deployed in resource-limited settings, making it accessible to farmers. The lightweight model can help farmers monitor and detect leaf diseases in real time, thus improving crop management and minimizing the use of pesticides, leading to more sustainable farming

practices. Furthermore, this study contributes to the broader community through the dissemination of knowledge and insights, providing valuable literacy on leveraging machine learning techniques for sustainable agricultural practices.

2. Literature Review

Literature study contains a discussion of several theories that form the foundation for this research. These theories serve as supporting theories in the development of this study.

2.1 YOLOv8

YOLO is a component of the CNN algorithm that can detect objects in images or videos, represented as bounding boxes. YOLO works by separating images into SXS matrices, with each box separator value serving as a bounding box. The bounding box will classify the class and then predict the probability and offset value. The bounding box that exceeds the probability value threshold will be selected to find objects in the image [17].

YOLOv8 consists of three main modules, namely backbone, neck, and head. Backbone is a deep learning architecture that functions to extract features from input images, then the neck module that will combine the features obtained from the results of feature extraction in the backbone module, and the head module that will predict the class and bounding box of the object, and will be the output of the results of the object detection model [18].

2.2 Optimization Strategy

Optimization in machine learning is the process of finding the best parameters so that the model can produce the best performance [19]. Model optimization in machine learning is an important aspect that influences the performance of the machine learning model. The application of optimization strategies in machine learning models involves techniques and strategies for using and tuning parameters to obtain the best results from the machine learning model [19]. Network reduction is one of the optimization methods applied to the YOLOv8 model to generate a lighter model by reducing the depth of the YOLO model [15]. Network reduction is applied to the YOLOv8 model to achieve the same or even improved performance from the baseline model, but with a model complexity that tends to be lighter.

3. Research Method

The research method includes an explanation of the research stages used in the mango leaf detection using YOLOv8 with a network reduction strategy. The research method used in conducting research can be seen in Figure 1.

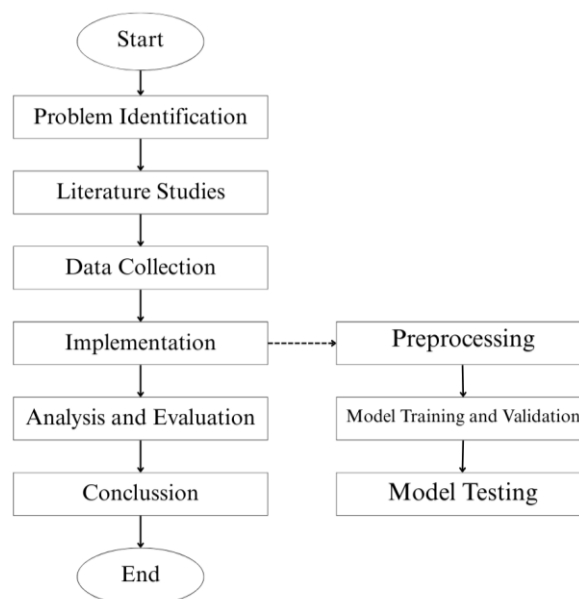


Figure 1. Research Methods

Figure 1 illustrates the research flow employed in mango leaf detection using YOLOv8 with a network reduction strategy. The research flow consists of 6 stages, which include problem identification, literature study, data collection, implementation, analysis and evaluation, and conclusion.

3.1 Dataset

The dataset used is a public dataset available on Mendeley Data [20]. There are 2,479 images across five classes. Figure 2 shows the dataset sample. In this study, the datasets are split into three parts, i.e., training, validation, and testing, in a 70:20:10 ratio.

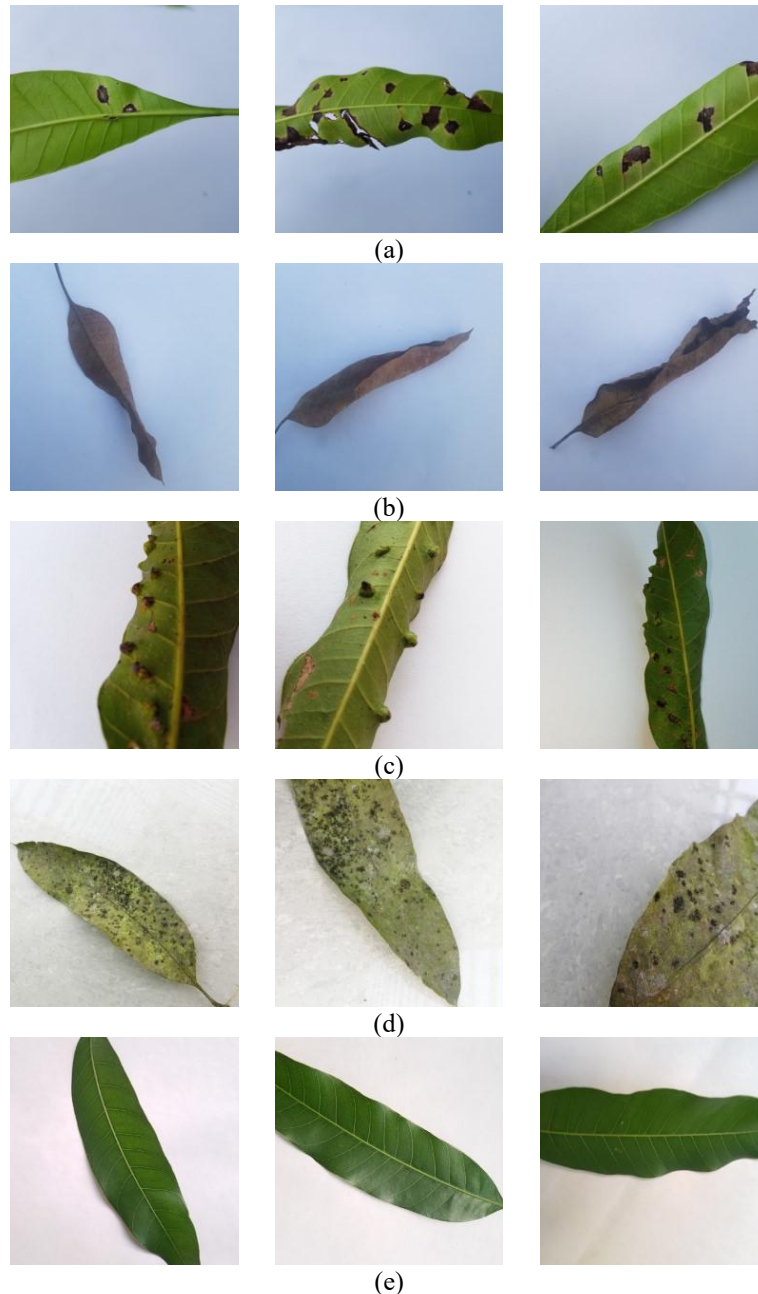


Figure 2. Sample of Mango Leaf Dataset: (a) Anthracnose (b) Die Back (c) Gall Midge (d) Sooty Mould (e) Healthy

Figure 2 shows a sample of a dataset consisting of mango leaves affected by different conditions. There are five distinct categories shown, with a total of 2,479 images across these classes. The first row (a) shows leaves affected by Anthracnose, characterized by dark brownish

lesions on the leaves. The second row (b) displays leaves suffering from Die Back, where the edges are curled and dried. The third row (c) represents the Gall Midge condition, marked by multiple small bulging spots on the leaf surface. The fourth row (d) illustrates the Sooty Mould, where the leaves exhibit a blackish fungal growth. Finally, the fifth row (e) shows healthy leaves, with no visible damage, serving as the control class for the study. This image helps visualize the variety of conditions that mango leaves can undergo, assisting in the development of image classification models to distinguish between these different states for agricultural monitoring and disease management.

3.2 Network Reduction Strategy

This study implemented a network reduction strategy on the YOLOv8 network. Network reduction strategy is fine-tuning the YOLOv8 network to produce a lighter model by reducing the network depth in the YOLO model. Table 1 shows the network reduction strategy in the YOLOv8 model

Table 1. Network Reduction on YOLOv8

Model	Network Depth
YOLOv8-85	85%
YOLOv8-90	90%
YOLOv8-95	95%
YOLOv8-Baseline	100%

A strategy was applied to reduce the network size by testing four different depth scenarios. The goal was to create a lighter model while maintaining its performance. To achieve this, adjustments were made to the YOLOv8 configuration file, reducing the network depth, width, and maximum channels. For YOLOv8-85, the depth, width, and max channels were set to 0.85, 0.85, and 512, respectively. For YOLOv8-90, the settings were 0.90, 0.90, and 512, respectively. Similarly, for YOLOv8-95, the values were set to 0.95, 0.95, and 512. YOLOv8-Baseline serves as the reference model, using the full network depth of 100%.

Parameters nc: 80 # number of classes # [depth, width, max_channels] t: [1.0, 1.0, 512]
YOLOv8.0n backbone backbone: # [[from, repeats, module, args] - [-1, 1, Conv, [64, 3, 2]] # 0-P1/2 - [-1, 1, Conv, [128, 3, 2]] # 1-P2/4 - [-1, 3, C2f, [128, True]] - [-1, 1, Conv, [256, 3, 2]] # 3-P3/8 - [-1, 6, C2f, [256, True]] - [-1, 1, Conv, [512, 3, 2]] # 5-P4/16 - [-1, 6, C2f, [512, True]] - [-1, 1, Conv, [1024, 3, 2]] # 7-P5/32 - [-1, 3, C2f, [1024, True]] - [-1, 1, SPPF, [1024, 5]] # 9
YOLOv8.0n head head: - [-1, 1, nn.Upsample, [None, 2, "nearest"]] - [[-1, 6], 1, Concat, [1]] # cat backbone P4 - [-1, 3, C2f, [512]] # 12 - [-1, 1, nn.Upsample, [None, 2, "nearest"]] - [[-1, 4], 1, Concat, [1]] # cat backbone P3 - [-1, 3, C2f, [256]] # 15 (P3/8-small) - [-1, 1, Conv, [256, 3, 2]] - [[-1, 12], 1, Concat, [1]] # cat head P4 - [-1, 3, C2f, [512]] # 18 (P4/16-medium) - [-1, 1, Conv, [512, 3, 2]] - [[-1, 9], 1, Concat, [1]] # cat head P5 - [-1, 3, C2f, [1024]] # 21 (P5/32-large) - [[15, 18, 21], 1, Detect, [nc]] # Detect(P3, P4, P5)

Figure 3. YOLOv8 Network

Figure 3 presents the detailed structure of the YOLOv8 network, outlining both its backbone and head configurations. The backbone is composed of a series of convolutional layers, each with varying depths and channels, designed to progressively extract more complex and abstract features from the input image. These convolutional layers start with relatively straightforward operations and gradually increase in depth and complexity, enabling the model to capture fine-grained spatial information and higher-level patterns necessary for object recognition. The backbone also integrates specialized modules, such as the C2f blocks, which enhance feature representation by leveraging advanced techniques to improve the model's ability to detect intricate patterns in the data.

The head configuration is responsible for utilizing the features extracted by the backbone to generate the final predictions, including bounding boxes and class labels. This part of the network is composed of detection layers that operate at different stages, allowing the model to make accurate predictions at various levels of feature abstraction. The head also includes components like upsampling layers, which adjust the size of feature maps to match the required input dimensions for detection, and concatenation layers that merge features from different stages of the network to improve prediction accuracy.

The YOLOv8-Baseline model serves as the reference model in this setup, utilizing the full network depth and configuration, and is used as the benchmark for evaluating the performance of different depth scenarios. By adjusting the depth, width, and maximum number of channels, the model can be optimized for size and computational efficiency without sacrificing detection performance. This flexibility makes YOLOv8 adaptable to various use cases, balancing model size and accuracy to achieve optimal results in object detection tasks.

3.3 Model Performance Evaluation

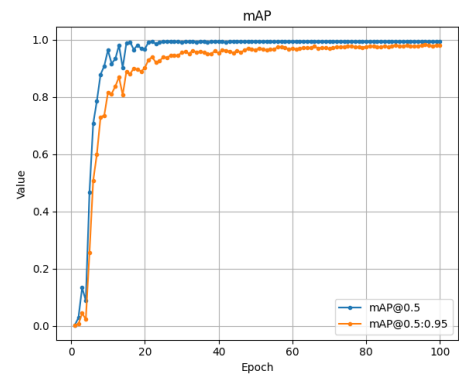
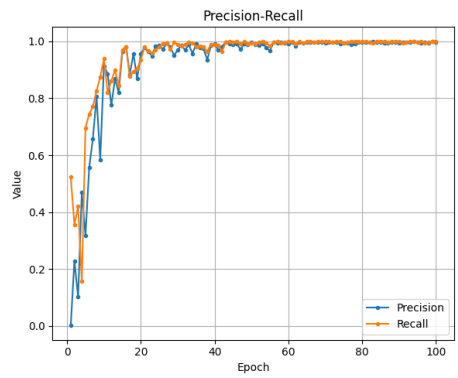
Performance evaluation is the process of measuring the performance of a model in performing a specific task based on test data. Performance evaluation of a model can use various evaluation metrics depending on the type of model and the purpose of creating the model, such as classification, segmentation, or object detection. The method used in this study involves examining the values of precision, recall, mAP50, mAP50-95, and the confusion matrix.

4. Result and Discussion

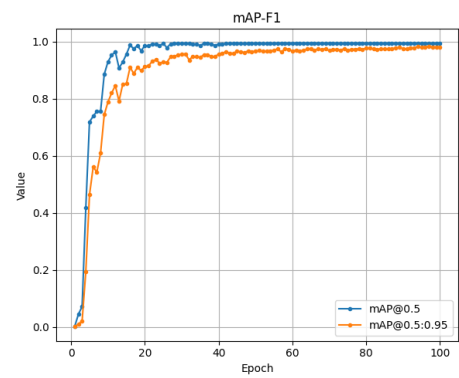
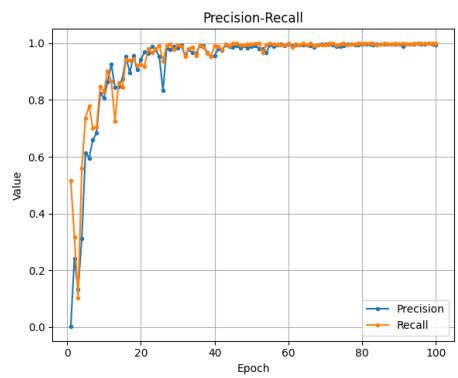
This section discusses the results of the research, including the performance comparison of each scenario from the application of network reduction to the YOLOv8 model. Figure 4 shows the matrices for four models based on precision-recall and mean average precision.

4.1 Performance Result

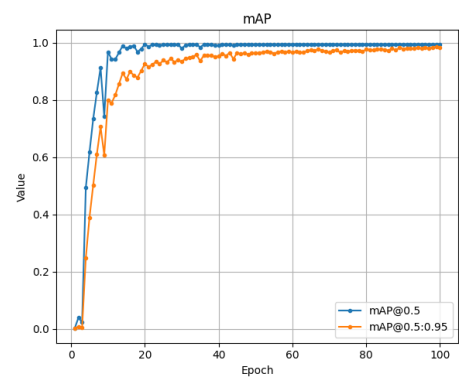
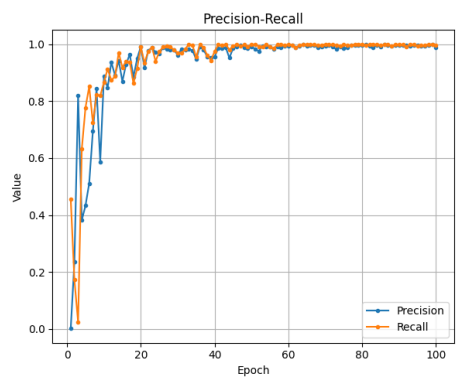
The training process was carried out on the YOLOv8 model by applying four experimental scenarios, as outlined in Chapter 3. Scenario (a) involves training the YOLOv8-baseline model with a network depth of 100% without implementing the network reduction optimization strategy. Scenario (b) entails training the YOLOv8-95 model with a network depth of 95% and applying the network reduction optimization strategy by 5%. Scenario (c) involves training the YOLOv8-90 model with a network depth of 90% and applying the network reduction optimization strategy by 10%. Scenario (d) consists of training the YOLOv8-85 model with a network depth of 85% and applying the network reduction optimization strategy by 15%. The models from each scenario will be tested and evaluated. The following are the results of the modeling for each of the scenarios described.



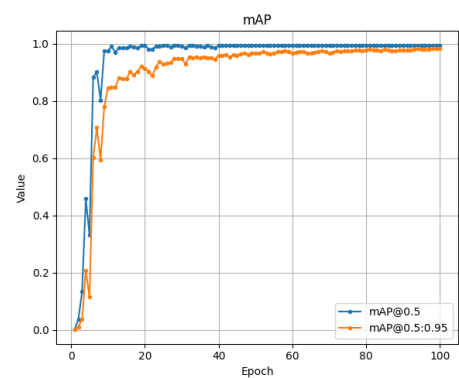
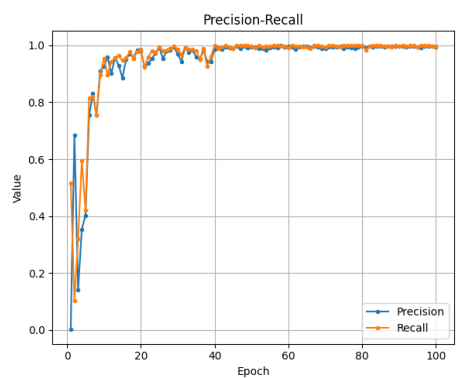
(a)



(b)



(c)



(d)

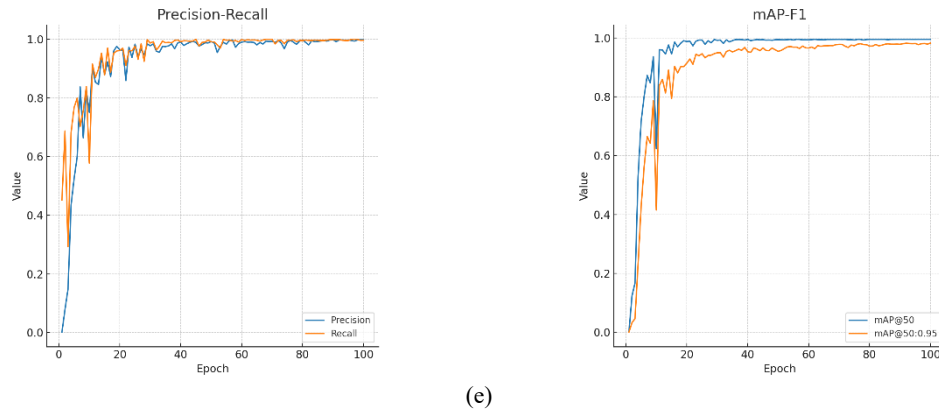


Figure 4. Precision Recall and Mean Average Matrices of Network Reduction Strategy: (a) YOLOv8-Baseline (b) YOLOv8-95 (c) YOLOv8-90 (d) YOLOv8-85 (e) YOLOv8-80

Figure 4 above illustrates the evolution of precision, recall, mAP50, and mAP50-95 across training epochs for four YOLOv8 model variants. YOLOv8-Baseline model shows a rapid increase in precision and recall during the initial epochs, with both metrics exceeding 0.90 early in the training. However, the mAP50-95 curve rises more gradually and plateaus at a lower value, indicating that the model's ability to generalize across different IoU thresholds is relatively limited. Subfigure (b), representing the YOLOv8-95 model, demonstrates improved stability in both precision-based metrics and mAP values, suggesting more consistent learning across epochs. In subfigure (c), the YOLOv8-90 model exhibits even stronger performance, where mAP50 and mAP50-95 curves are smoother and more closely aligned, reflecting better feature learning and localization accuracy. The best performance is observed in subfigure (d) with the YOLOv8-85 model, which not only achieves the highest mAP50-95 value but also maintains highly stable and convergent precision, recall, and mAP curves. This model reaches optimal accuracy early in the training and sustains it consistently, highlighting its robust generalization capability. Among all tested models, the YOLOv8-85 model that applies the network reduction strategy of 15% emerges as the most effective configuration, achieving the best trade-off between detection performance and training stability because it shows faster and more stable convergence on both main metrics used.

4.2 Performance Comparison

This section presents a comparative analysis of the performance of each YOLOv8 model variant based on key evaluation metrics, including precision, recall, mAP50, and mAP50-95, as shown in Table 2. The purpose of this comparison is to determine which model configuration offers the best trade-off between detection accuracy and generalization capability. By analyzing the results quantitatively, we can identify the most optimal model variant for the intended detection task and evaluate the impact of network reduction on overall performance.

Table 2. Model Performance Comparison

Model	Precision	Recall	mAP50	mAP50-95
YOLOv8-Baseline	99,7%	100%	99,5%	98,5%
YOLOv8-95	99,6%	99,9%	99,5%	98,4%
YOLOv8-90	99,8%	99,9%	99,5%	98,5%
YOLOv8-85	99,7%	100%	99,5%	98,8%
YOLOv8-80	99,6%	99,9%	99,5%	98,4%

Table 2 presents a comparison of different YOLOv8 models in terms of performance. YOLOv8-85 model outperforms the other variants across key performance metrics. It achieves the highest precision and recall, both at 100%, and shows a solid performance in mAP50 (99.5%) and mAP50-95 (98.8%). These results highlight YOLOv8-85's ability to quickly converge to optimal performance, demonstrating faster and more stable results compared to the other models. The different models, such as YOLOv8-Baseline, YOLOv8-95, and YOLOv8-90, have slightly

lower performance in terms of mAP50-95 and recall, which positions YOLOv8-85 as the most reliable model for accurate and consistent results.

Table 3. Model Complexity Comparison

Model	Parameter	Training Time	GFLOPs	Model Size (MB)
YOLOv8-Baseline	43.633.695	3.311	165.4	87.7
YOLOv8-95	39.979.807	3.345	154.4	80.4
YOLOv8-90	34.611.207	3.399	130.8	69.6
YOLOv8-85	31.370.175	3.145	118.9	63.1
YOLOv8-80	24.152.823	2.584	92.2	48.6

Table 3 presents a comparison of different YOLOv8 models in terms of model complexity, comparing the number of parameters, training time, GFLOPs, and model size. YOLOv8-85 is the most efficient in terms of model size, with only 63.1 MB, and it also requires the least computational power (118.9 GFLOPs). Despite this, it achieves high performance, making it a highly efficient model. Additionally, YOLOv8-85 has the fewest parameters (31,370,175) and requires 3.145 hours of training, which is slightly less than the other models but still offers optimal results in terms of both accuracy and efficiency.

5. Conclusion and Recommendation

In conclusion, YOLOv8-85 demonstrates the best balance between high performance and low complexity. It excels in key metrics such as precision (99.7%), recall (100%), and mAP50-95 (98.8%), outperforming the other variants. Additionally, it maintains a relatively lower computational cost, which makes it efficient for real-world applications. However, the YOLOv8-80 model, although slightly behind in mAP50-95 (98.4%), offers a highly competitive alternative with fewer parameters (24,152,823) and a smaller model size (48.6 MB). This makes YOLOv8-80 an excellent choice for applications where model size and training time are more critical factors, while still maintaining exceptional performance metrics. In summary, this research demonstrates that applying a network reduction strategy to the YOLOv8 model effectively optimizes performance for mango leaf disease detection. Among the configurations tested, YOLOv8-85 offers the best trade-off between accuracy, efficiency, and model complexity, making it the most reliable model for deployment in resource-constrained environments.

References

- [1] D. Hidayat, "Klasifikasi Jenis Mangga Berdasarkan Bentuk Dan Tekstur Daun Menggunakan Metode Convolutional Neural Network (CNN)," *Journal of Information Technology and Computer Science (INTECOMS)*, vol. 5, no. 1, pp. 98, 2022, doi: 10.31539/intecom.v5i1.3401.
- [2] Badan Pusat Statistik, "Produksi Buah-buahan dan Sayuran Menurut Jenis Tanaman Menurut Provinsi," 2023. [Online]. Available: <https://www.bps.go.id/id/statistics-table/3/U0dKc1owczVSaIJ5VFdOMWVETnIVRVJ6YIRJMFp6MDkjMw==/produksi-buah-buahan-menurut-jenis-tanaman-menurut-provinsi.html?year=2023> (accessed Jan. 12, 2025).
- [3] P. A. Widjaja and J. R. Leonesta, "Differentiate a Health and Sick of Mango Leaves Using YOLOv4," *Formosa Journal of Science and Technology*, vol. 2, no. 7, pp. 1749–1758, Jul. 2023, doi: 10.55927/fjst.v2i7.4792.
- [4] Krishnamoorthy, L. V. N. Prasad, C. S. P. Kumar, B. Subedi, H. B. Abraha, and V. E. Sathishkumar, "Rice leaf diseases prediction using deep neural networks with transfer learning," *Environ Res*, vol. 198, Jul. 2021, doi: 10.1016/j.envres.2021.111275.
- [5] A. TiaraSari and E. Haryatmi, "Penerapan Convolutional Neural Network Deep Learning dalam Pendeteksian Citra Biji Jagung Kering," *Jurnal Rekayasa Sistem dan Teknologi Informasi (RESTI)*, vol. 5, no. 2, pp. 265–271, Apr. 2021, doi: 10.29207/resti.v5i2.3040.

- [6] A. Cardellicchio *et al.*, “Detection of tomato plant phenotyping traits using YOLOv5-based single stage detectors,” *Computers and Electronics in Agriculture*, vol. 207, Apr. 2023, doi: 10.1016/j.compag.2023.107757.
- [7] P. T. Ompusunggu, “Klasifikasi Penyakit Tanaman Pada Daun Kentang Dengan Metode Convolutional Neural Network Arsitektur Mobilenet,” *Jurnal Syntax Fusion*, vol. 2, no. 09, pp. 740–751, Sep. 2022, doi: 10.54543/fusion.v2i09.217.
- [8] Mutia Fadhillah, D. Suryani, A. Labellapansa, and H. Gunawan, “Corn Leaf Diseases Recognition Based on Convolutional Neural Network,” *IT Journal Research and Development*, vol. 8, no. 1, pp. 14–21, Aug. 2023, doi: 10.25299/itjrd.2023.13904.
- [9] F. Agustina and M. Sukron, “Deteksi Kematangan Buah Pepaya Menggunakan Algoritma YOLO Berbasis Android,” *Journal Informasi Komputer Akuntansi dan Manajemen*, vol. 18, no. 2, Sep. 2022.
- [10] L. Setiana Riva and Jayanta, “Deteksi Penyakit Tanaman Cabai Menggunakan Algoritma YOLOv5 Dengan Variasi Pembagian Data,” *Jurnal Informatika: Jurnal pengembangan IT (JPIT)*, vol. 8, no. 3, Sep. 2023.
- [11] S. B. Mamun, I. J. Payel, M. T. Ahad, A. S. Atkins, B. Song, and Y. Li, “Grape Guard: A YOLO-based mobile application for detecting grape leaf diseases,” *Journal of Electronic Science and Technology*, vol. 23, no. 1, Mar. 2025, doi: 10.1016/j.jnlest.2025.100300.
- [12] P. A. Widjaja and J. R. Leonesta, “Determining Mango Plant Types Using YOLOv4,” *Formosa Journal of Science and Technology*, vol. 1, no. 8, pp. 1143–1150, Dec. 2022, doi: 10.55927/fjst.v1i8.2155.
- [13] A. Wibowo, L. Lusiana, and T. K. Dewi, “Implementasi Algoritma Deep Learning You Only Look Once (YOLOv5) Untuk Deteksi Buah Segar Dan Busuk,” *Paspalum: Jurnal Ilmiah Pertanian*, vol. 11, no. 1, p. 123, Mar. 2023, doi: 10.35138/paspalum.v11i1.489.
- [14] S. Aras, P. Tanra, and M. Bazhar, “Deteksi Tingkat Kematangan Buah Tomat Menggunakan YOLOv5,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 2, pp. 623–628, Mar. 2024, doi: 10.57152/malcom.v4i2.1270.
- [15] I. N. Piarsa, N. P. Sutramiani, and I. W. A. Surya Darma, “Network Reduction Strategy and Deep Ensemble Learning for Blood Cell Detection,” *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, vol. 14, no. 3, p. 161, Dec. 2023, doi: 10.24843/lkjiti.2023.v14.i03.p04.
- [16] I. W. A. S. Darma, N. Suciati, and D. Siahaan, “CARVING-DETC: A network scaling and NMS ensemble for Balinese carving motif detection method,” *Visual Informatics*, vol. 7, no. 3, pp. 1–10, Sep. 2023, doi: 10.1016/j.visinf.2023.05.004.
- [17] A. A. Rasjid, B. Rahmat, and A. N. Sihananto, “Implementasi YOLOv8 Pada Robot Deteksi Objek,” *Journal of Technology and System Information*, vol. 1, no. 3, p. 9, Jul. 2024, doi: 10.47134/jtsi.v1i3.2969.
- [18] Abin Timilsina, “YOLOv8 Architecture Explained,” Mar. 17, 2024. [Online]. Available: <https://abintimilsina.medium.com/yolov8-architecture-explained-a5e90a560ce5> (accessed Aug. 17, 2025).
- [19] A. Y. Nugroho and F. Kristanto, “Mengejar Kinerja Maksimal: Teknik Pengoptimalan Terkini dalam Pembelajaran Mesin,” *Jurnal Penelitian Teknologi Informasi dan Sains*, vol. 2, no. 3, pp. 59–69, Aug. 2024, doi: 10.54066/jptis.v2i3.2358.
- [20] S. Ali *et al.*, “MangoLeafBD Dataset.” [Online]. Available: <https://www.kaggle.com/datasets/aryashah2k/mango-leaf-disease-dataset> (accessed Oct. 24, 2024).