JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 10 Number 1 | June 2025: 153-164
DOI: 10.15575/join.v10i1.1441

Multi-Platform Detection of Melon Leaf Abnormalities Using AVGHEQ and YOLOv7

Sahrial Ihsani Ishak[1], Karlisa Priandana[2], Sri Wahjuni[3]
[1,2,3] School of Data Science, Mathematics and Informatics, IPB University, Bogor, Indonesia

Article Info

Article history:

Received August 15, 2024
Revised December 18, 2024
Accepted December 27, 2024
Published May 10, 2025

Keywords:

AVGHEQ
Internet of Things
Melon Disease Detection
Multiplatform System
YOLOv7

ABSTRACT

This research develops a multiplatform system for detecting abnormalities in melon leaves, integrating an Internet of Things (IoT) approach using Jetson Nano, a Streamlit-based website, and a mobile application for real-time monitoring. The system employs preprocessing with Average Histogram Equalization (AVGHEQ) to enhance image quality, followed by modeling with the YOLOv7 algorithm on a dataset of 469 training images and 52 test images, validated through 5-fold cross-validation. The model achieved a mean Average Precision (mAP) of 84% with an inference detection time of 4.5 milliseconds. Implementation on Jetson Nano resulted in a 25% increase in CPU usage (from 25% to 50%) and a 20% increase in RAM usage (from 70% to 90%). By combining these platforms and leveraging robust data preprocessing and modeling techniques, the system provides an accessible, efficient, and scalable solution for agricultural monitoring, enabling farmers to address plant health issues promptly and effectively.

Corresponding Author:

Karlisa Priandana,
Computer Science Study Program, School of Data Science, Mathematics and Informatics, IPB University, Bogor, Indonesia
Jl. Raya Dramaga, Kampus IPB Dramaga, Bogor (16680), West Java, Indonesia
Email: karlisa@apps.ipb.ac.id

1. INTRODUCTION

Agricultural commodities play an important role in national economic growth [1], [2] . According to data from the Central Statistics Agency of the Republic of Indonesia, fruit production in Indonesia increased from 2016 to 2022 [3]. However, there are commodities that experience fluctuations in production, one of which is melon [3]. These fluctuations are caused by poor plant growth during the growing period [4].

Abnormalities in melon plants can occur due to two main factors: lack of nutrients and pest infestation, as well as diseases that attack periodically. This disorder can lead to crop failure if not treated quickly and appropriately [5].

Abnormalities in melon plants are usually treated with a diagnosis from the farmer or plant nutritionist, which requires time for further identification in the laboratory [6]. To overcome this problem, information technology can help in detecting abnormalities in melon plants effectively and efficiently. One solution that can be used is artificial intelligence (AI) [7].

Artificial intelligence (AI) is a field of science that integrates machine and human intelligence [8], [9]. One of the branches of AI that has a good performance is deep learning. Deep learning is a development of machine learning that focuses more on feature extraction, so that its performance is more optimal [6]. One of the problems that can be solved with deep learning is object detection. Object detection methods fall into two categories: one-stage, which is fast in inference, and two-stage, which is more accurate [11].

The detection model to be used is YOLOv7, one of the latest methods of the YOLO series, which is superior in detection speed compared to other algorithms [8]. The model will be applied to computer mini-devices such as the Jetson Nano, which is capable of executing artificial neural networks in parallel for various tasks such as image classification, object detection, segmentation, and sound processing [9]. The Jetson Nano has a more powerful GPU than the Raspberry Pi, making it suitable for AI applications [10].

The implementation of the object detection method on the Jetson Nano has been carried out by several researchers. For example, [15] detected powdery mildew on strawberry leaves using the DAC-YOLOv4 method, which showed better mAP and could be applied in real-time to Jetson Xavier and Jetson Nano. [16] showed that the Pruned-YOLO v5s+Shuffle (PYSS) model had high performance in detecting powdery mildew in melon leaves, with improved mAP and precision, as well as reduced model size and inference time.

This research aims to build a monitoring system for melon plant diseases based on melon leaves that can be monitored periodically. The main focus is to build an IoT system consisting of the Jetson Nano as a mini computer, a USB camera to capture the melon leaf images, a Wi-Fi adapter, and a website to show the abnormality detection results to the user.

2. METHOD

Figure 1 illustrates the flow of this research, beginning with the acquisition of secondary data from this research by [17] which includes both training and test data. The training data is first prepared using augmentation techniques and AVGHEQ. Next, cross-validation with 5 folds is performed. Each model generated from the folds is evaluated, and an ensemble method is used to select the best model. This model is then integrated into the designed multiplatform system, which includes Jetson Nano, a website, and an Android application. Finally, the multiplatform system is evaluated, and the research is concluded.

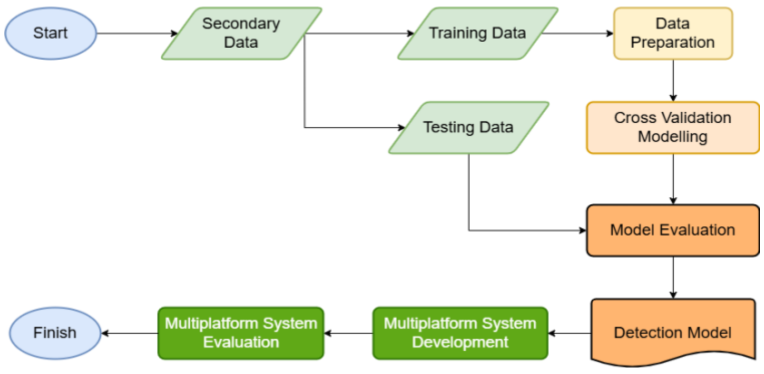


Figure 1. Research flow

2.1. *Data*

This study uses secondary data on melon leaf images from the results of previous studies [17] The entire dataset consists of 521 images, which have been labeled and processed in txt format. Each image has a resolution of 5 megapixels, with a width and height of 2592 and 1944 pixels, respectively. The dataset was divided into 90% training data and 10% test data.

2.2. *Data Preparation*

Data preparation is carried out on the training data using image preprocessing AVGHEQ (Averaging Histogram Equalization) and augmentation techniques. The advantage of AVGHEQ is its ability to adaptively enhance image contrast while reducing noise and preserving image details [18]. Image augmentation is an important stage that can affect the results of deep learning models [14]. Image augmentation in deep learning is used to increase the quantity and diversity of data on the dataset. This technique is called augmentation [19]–[21]. Augmentation can also be used to reduce overfitting in models and handle small amounts of data [20]. In this study, the utilized augmentation techniques are those defined in YOLOv7 [12] as listed in Table 1.

Table 1. Augmentation Techniques

No	Technique	Definition	Value
1	HSV (H)	image HSV-Hue augmentation (fraction)	0.015
2	HSV (S)	image HSV-Saturation augmentation (fraction)	0.7
3	HSV (V)	image HSV-Value augmentation (fraction)	0.4
4	Translate	image translation (+/- fraction)	0.2
5	Scale	image scale (+/- gain)	0.9
6	Flip (Left and Right)	image flip left-right (probability)	0.5
7	Mosaic	image mosaic (probability)	1
8	Mixup	image mixup (probability)	0.15
9	Paste-in	Image copy-paste (probability)	0.15

2.3. *Cross-Validation Modeling*

The goal of k-fold cross-validation is to evaluate the performance of the model more accurately by dividing the training data into subsets and ensuring that each piece of data is used for training and validation [22]. This method helps reduce bias and variance in model performance estimation by dividing the data into *k* subsets and using each subset as one-time test data (validation). All data is used for training and validation, so no data is wasted, which is especially useful when available data is limited. By calculating the average performance of *k* different experiments, k-fold cross-validation provides a more stable and reliable estimate of the model's performance compared to the single validation method. In addition, this method helps detect and prevent overfitting by testing the model on parts of the data that are not used during training, ensuring that the model does not over-adapt to the training data.

In this study, the cross-*validation modeling* stage was carried out on the initial training data using 5 folds. In other words, the data is divided into 5 equal parts randomly. Each fold contains 80% of the initial training data as training data and 20% of the initial training data as validation data, and is trained using YOLOv7. The training was done using a *hyperparameter* configuration obtained from the grid search tuning method, as shown in Table 2.

Table 2. Hyperparameters for Training

No	Parameter	Value
1	Batch size	32
2	Epoch	5000
3	Image Size	640 × 640 pixel
4	Pretrained Model	Yolov7-tiny
5	Learning Rate	0.001
6	Kernel Size	3
7	Activation Functions	SiLU
8	Pooling Layer	MaxPool
9	Batch Size	32
10	Momentum	0.93
11	Using Augmentation	Yes
12	Patience	No, 300

2.4. Model Evaluation

The models developed are evaluated using specific metrics to identify the most effective results. The metrics used in this study are mAP, accuracy, and detection time. The mAP formula is obtained by calculating the accuracy, recall, and Average Precision (AP) values, as in (1) – (5) [23].

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

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

$$acc(i) = \frac{TP(i)+TN(i)}{TP(i)+FP(i)+TN(i)+FN(i)} \quad (3)$$

$$AP(i) = \int_{r=0}^{1} p(r)dr \quad (4)$$

$$mAP = \frac{1}{K} \sum_{i=1}^{K} AP(i) \quad (5)$$

where p = precision; r = recall; i = index for class; K = number of class; TP = True Positive; FP = False Positive; FN = False Negative; acc = Accuracy; AP = Average Precision; mAP = mean Average Precision.

2.5. *Multiplatform System Development*

The development of the tool and system was carried out to ensure practical functionality and tangible contributions to the field. The components required to build the tool are listed in Table 3. The overall workflow of the tool and system is shown in Figure 7-9. Users have three device options for detecting anomalies in melon leaves, which can be adapted to suit their specific needs and work environment.

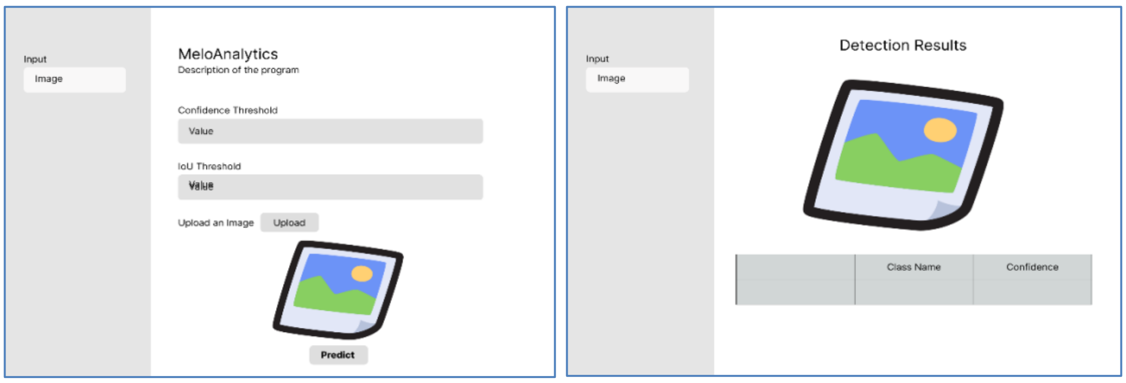
The first option is to use the Jetson Nano, a computing device that is ideal for field or production environments that require real-time data processing. The second option is to use the website, which is suitable for working environments with stable Internet connections. The third option is to use the Android application, which is ideal for users who need high mobility and want to perform live detection via their Android device in the field or at different locations. These three options allow users to choose the most efficient and effective method based on their specific use cases and working conditions.

If the user chooses to use the Jetson Nano device, the results of the melon leaf anomaly detection will be displayed directly on the screen connected to the device, allowing for instant, real-time monitoring in the field.

Table 3. Device Hardware Components

No	Component	Function
1	Jetson Nano 2GB	Mini computer to house the built model
2	WiFi Adapter	Connecting a mini computer to the internet
3	USB Camera	Capture melon leaves image

If the user selects the website, the analysis results will be displayed directly on the website, accessible via a browser, providing flexibility for remote monitoring. If the Android option is selected, the detection results will be displayed on the Android app, making it easy to access anywhere. In addition, whether via the website or the Android app, the detection results can be stored in the cloud, ensuring that the data is securely recorded and accessible at any time. However, when using the Jetson Nano device, the results cannot be stored in the cloud, but instead are centrally processed and stored on the local device.



(a) (b)
Figure 2. Wireframe design for website, (a) main page, (b) detection results page

The website is designed to predict the melon plant abnormality using 3 inputs: images, videos, and webcams. The website features two pages: the main page and the prediction results page. Figure 2(a) shows the wireframe of the main page of the website, and Figure 2(b) shows the wireframe of the prediction results page.

The application we designed consists of five pages: the homepage, the main page, the analysis page, the information page, the detection page, and the detection results page. The designed application can be seen in Figure 3.

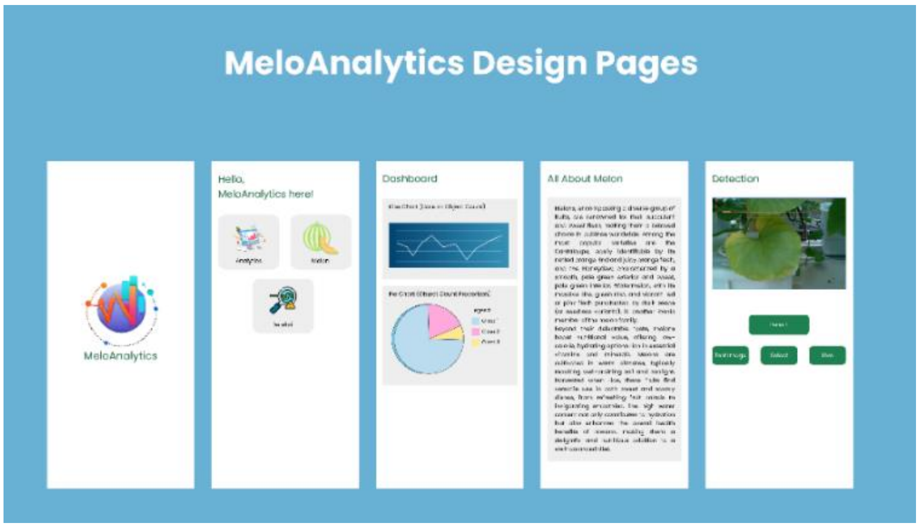


Figure 3. Wireframe design for mobile application

2.6. Multiplatform System Evaluation

System testing is carried out to ensure that the system that has been built can be used properly. Testing is carried out by trying it directly in the field by doing a predetermined scenario. As shown in Figure 4, the evaluation process starts from evaluating the model using mAP, accuracy, and detection time. Then, Jetson Nano resource usage was also evaluated by comparing the CPU and RAM usage before and after detection. Finally, the developed website and Android application were also evaluated using a black box framework to test the functionality of the website based on the user's perspective.

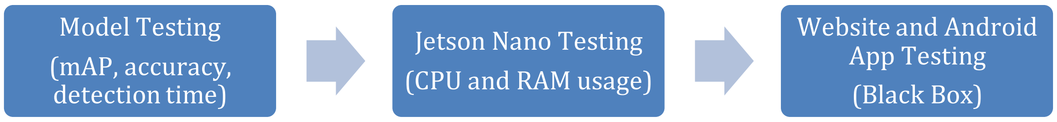

Model Testing
(mAP, accuracy, detection time)
Jetson Nano Testing
(CPU and RAM usage)
Website and Android App Testing
(Black Box)

Figure 4. IoT System Evaluation Flow

3. RESULT AND DISCUSSION

3.1. *Data Preparation Results*

Data preparation aims to prepare the data for training by the YOLOv7 object detection model. Figure 5 illustrates the results of the augmentation performed using Python. The HSV (Hue, Saturation, Value) settings were 0.015, 0.7, and 0.4, respectively, which introduced significant color variations. The Translate technique, with a value of 0.2, introduced positional variations of objects within the images, while Scale, with a value of 0.9, altered object sizes. Flip (Left and Right) with a value of 0.5 provided orientation variations through horizontal flipping. The Mosaic technique, with a value of 1, combined multiple images into one, creating diverse contextual variations and scale comparisons. Mixup and Paste-in, each with a value of 0.15, enhanced model generalization by blending images with different weights and inserting additional images into the main image.

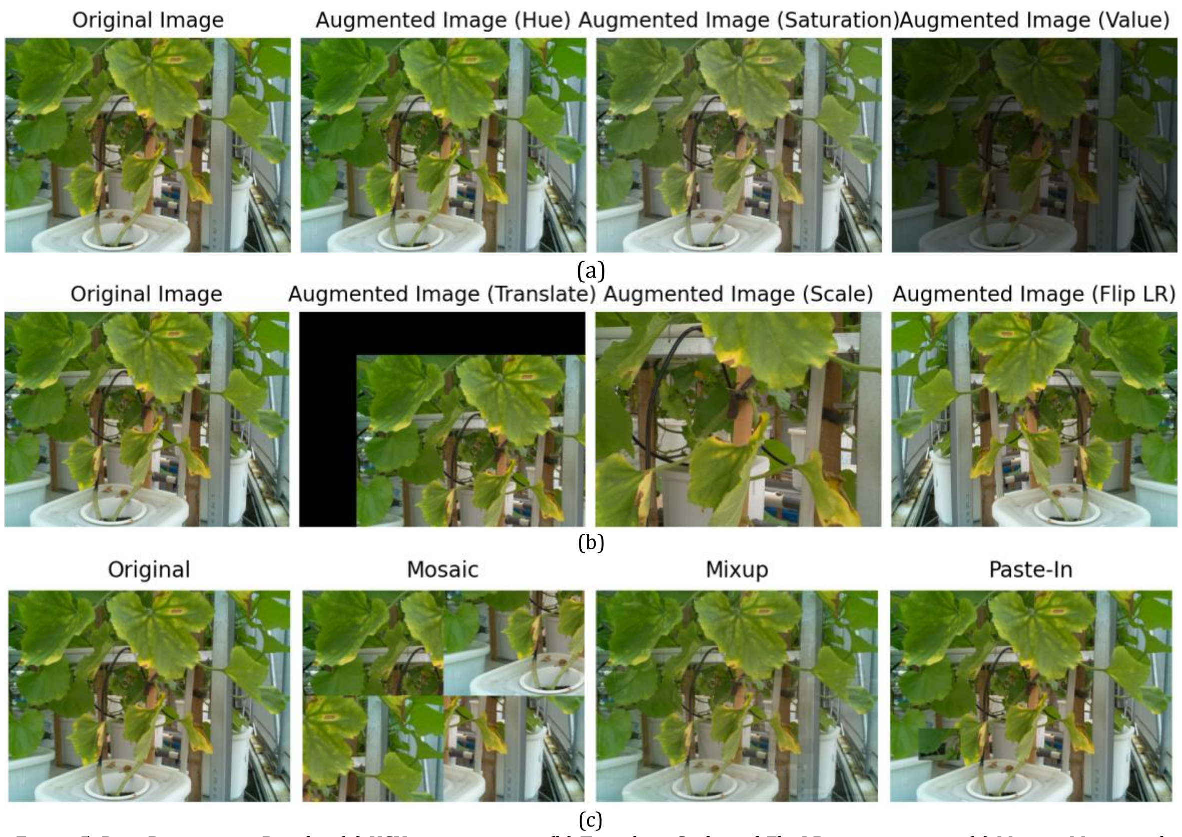


Figure 5. Data Preparation Results, (a) HSV augmentation, (b) Translate, Scale and FlipLR augmentation, (c) Mosaic, Mixup and Paste-in augmentation

3.2. *Cross-Validation Modeling Results*

Each dataset resulting from the cross-validation process is divided into training and validation data. The amount of data, abnormal objects, and normal objects in the original and preprocessed data is, respectively, (375 training data & 94 validation data), (2,362 training data & 590 validation data), and (1,882 training data & 471 validation data). The amount of data, abnormal objects, and normal objects in the augmented data are respectively (941 training data & 235 validation data), (5,554 training data & 1,389 validation data), and (5,160 training data & 1,290 validation data). The average training time of the model is 20 hours, the model size is 12.3 MB. The average of mAP on the training stage is 54.44%.

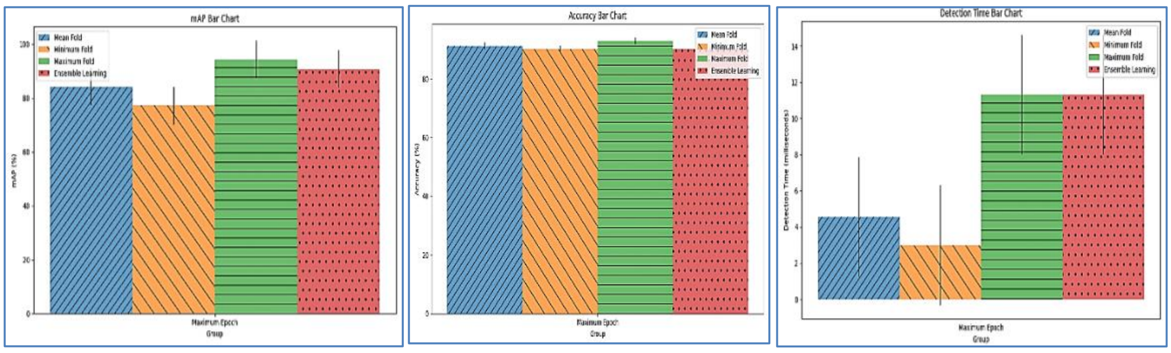
3.3. *Model Evaluation Results*

In object detection model evaluation, metrics such as Mean of all folds (average across all folds), Minimum of all folds (lowest value across all folds), and Maximum of all folds (highest value across all folds) are crucial for measuring the consistency and variation in model performance under various testing conditions. Mean of all folds provides an overview of the model's average performance across all folded datasets, while Minimum of all folds and Maximum of all folds indicate the lowest and highest points of the model's performance spectrum. Standard deviation (std) is used to measure the spread of data around the mean, which is essential for understanding how consistent the model is in its results. Ensemble learning configuration adds value by combining results from all folds to enhance overall performance. This analysis not only provides a deep understanding of the model's strengths in object detection performance but also clarifies areas where the model can be improved to achieve more consistent and reliable results in practical applications.

Figure 6(a) displays the mAP (Mean Average Precision) metric, which illustrates the average precision of the model in object detection. The average mAP achieved is 84.12%, with a minimum value of 77.1% and a maximum of 94.3%. A standard deviation of 7.0 indicates significant variation in object detection precision across different folded datasets, highlighting the model's capability and potential in handling varied testing conditions. The ensemble learning value set at 90.7 shows that the use of ensemble techniques contributes positively to overall mAP performance enhancement.

Figure 6(b) visualizes the model's Accuracy metric, reflecting the percentage of correct predictions compared to total predictions made. The model demonstrates good consistency with an average accuracy of 91.19%. The accuracy range across folds spans from 90.08% to 92.82%, with a standard deviation of 1.17, indicating relatively minor variations among the tests conducted. This standard deviation underscores the consistency in the model's accuracy performance across various testing conditions.

Figure 6(c) illustrates the model's Detection Time, which is the average time required for the model to detect objects. In this evaluation, the average detection time is 4.55 milliseconds, with a minimum time of 3.0 milliseconds and a maximum of 11.3 milliseconds. A standard deviation of 3.31 indicates significant variation in the model's detection time performance across different testing conditions, emphasizing the importance of efficiency in practical object detection applications. The ensemble learning value set at 11.3 indicates notable variability in detection time between testing conditions, which can impact the model's practical performance in real-world scenarios. This analysis provides a comprehensive overview of the strengths and challenges faced by the model across various evaluation aspects.



(a) (b) (c)

Figure 6. Values of (a) mAP, (b) accuracy, and (c) detection time with standard deviation

Modeling using patience 300 produces better accuracy compared with other models. However, the differences between the two are not much different. Both modelling results in better mAP, accurate detection and time than the previous modeling stage. Both also have good models, as evidenced by the fact that there is no overfitting because the mAP in the evaluation results is greater than the training results (Table 4).

Table 4. Comparison of using Patience and not using Patience

Model	mAP (%)		Accuracy (%)		Detection Time (ms)	
	Average	Rank	Average	Rank	Average	Rank
Maximum epoch	**84.12 ± 7.0**	**1**	**91.19 ± 1.2**	**2**	**4.55 ± 3.3**	**2**
Patience *300*	81.57 ± 4.1	2	92.23 ± 2.4	1	5.03 ± 3.9	1

3.4. *Multiplatform System Development Results*

The designed tool can be seen in Figure 7. Jetson Nano is a powerful edge computing platform, designed for machine learning and image processing tasks on IoT (Internet of Things) devices. The cooling fan serves to keep the Jetson Nano's temperature optimal during operation, preventing overheating that can affect performance. The power supply provides stable and sufficient electrical power to support the operation of the Jetson Nano. An HDMI cable connects the Jetson Nano to the monitor screen for visual display. The WiFi adapter allows the Jetson Nano to connect to the internet network. The USB camera is used as a visual input in the case of leaf abnormality detection, allowing the

Jetson Nano to process image data from the camera to detect and analyze anomalies in leaves by leveraging machine learning capabilities. The device is connected to a monitor in order to access the Jetson Nano OS. All of these tools work together to form a comprehensive and effective system in supporting leaf abnormality detection applications using the Jetson Nano.

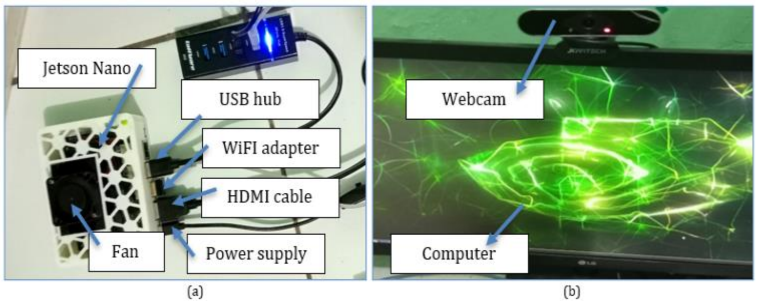
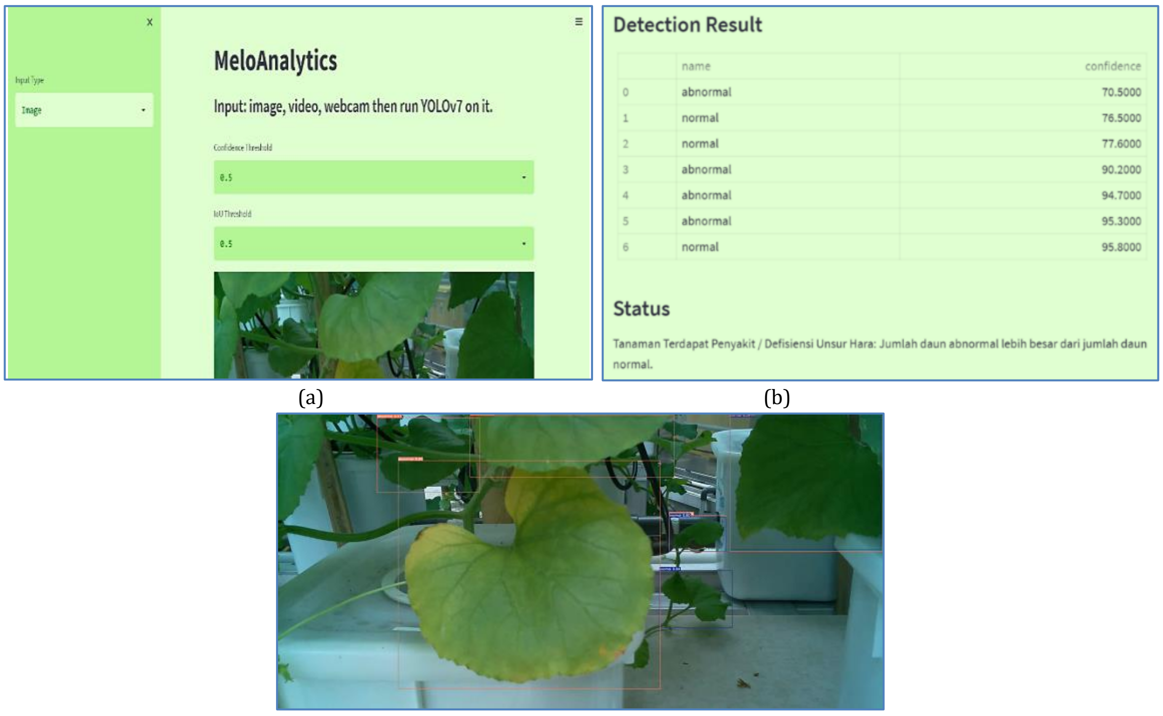


Figure 7. System IoT, (a) Jetson Nano, (b) computer and USB Camera

The system is a website and android based application designed to detect images or videos and then send the results to a Google Firestore server, as shown in Figure 8 and 9. Figure 8(a) displays the main page of the website, while Figure 8(b) shows the detection results of melon leaf images, along with the conclusions drawn from those results. This website-based system is built using Python and is hosted on Streamlit Cloud. It can detect objects in images, videos, and camera feeds. The detection results include bounding boxes, class names, and confidence scores of the detected objects Figure 8(c).



Input Type: Image

Confidence Threshold: 0.5

IoU Threshold: 0.5

Detection Result

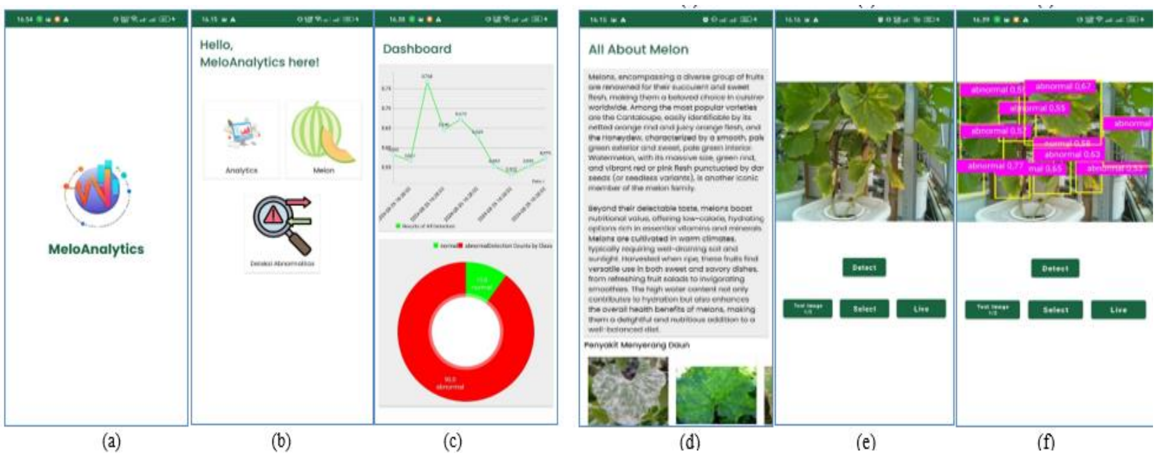
	name	confidence
0	abnormal	70.5000
1	normal	76.5000
2	normal	77.6000
3	abnormal	90.2000
4	abnormal	94.7000
5	abnormal	95.3000
6	normal	95.8000

Status

Tanaman Terdapat Penyakit / Defisiensi Unsur Hara: Jumlah daun abnormal lebih besar dari jumlah daun normal.

Figure 8. Home page (a), Detection page (b), (c) Detection results sample

Figure 9 shows the MeloAnalytics application interface consisting of six main pages. (a) Displays the initial page (splash screen) with the application logo. (b) The main page provides three options: Analytics, information about melons, and abnormality detection. (c) The analysis page (dashboard) contains a line graph and a pie chart displaying the results of abnormality detection analysis on melons. (d) The information page provides explanations about melons, including descriptions, health benefits, and leaf diseases affecting melons. (e) The detection page allows users to upload or capture images of melon leaves for analysis. (f) The detection results page displays images labeled as abnormal in areas detected with issues, along with confidence scores. This application is designed to facilitate the analysis and monitoring of melon plant health.



Melons, encompassing a diverse group of fruits are renowned for their succulent and sweet flesh, making them a beloved choice in cuisine worldwide. Among the most popular varieties are the Cantaloupe, easily identifiable by its netted orange rind and juicy orange flesh, and the Honeydew, characterized by a smooth, pale green exterior and sweet, pale green interior. Watermelon, with its massive size, green rind, and vibrant red or pink flesh punctuated by dar seeds (or seedless variants), is another iconic member of the melon family.

Beyond their delectable taste, melons boast nutritional value, offering low-calorie, hydrating options rich in essential vitamins and minerals. Melons are cultivated in warm climates, typically requiring well-draining soil and sunlight. Harvested when ripe, these fruits find versatile use in both sweet and savory dishes, from refreshing fruit salads to invigorating smoothies. The high water content not only contributes to hydration but also enhances the overall health benefits of melons, making them a delightful and nutritious addition to a well-balanced diet.

Figure 9. Pages on the app. (a) Home page, (b) Main page, (c) Analysis page, (d) About melon page, (e) Detection page, (f) Detection results page

3.5. *Multiplatform System Evaluation Results*

1. Jetson Nano Testing

The values set in the code when running the detection program are the Confidence Threshold and the IoU Threshold of 0.5 and 0.75. This value is used to obtain fairly accurate results for detecting leaf abnormalities in melon plants. The Jetson Nano's performance before and after detecting objects using the best models can be seen in Figures 10 and 11. Figure 10 illustrates the CPU usage on the tool before and after detection. Figure 11 illustrates the RAM usage on the tool before and after detection.

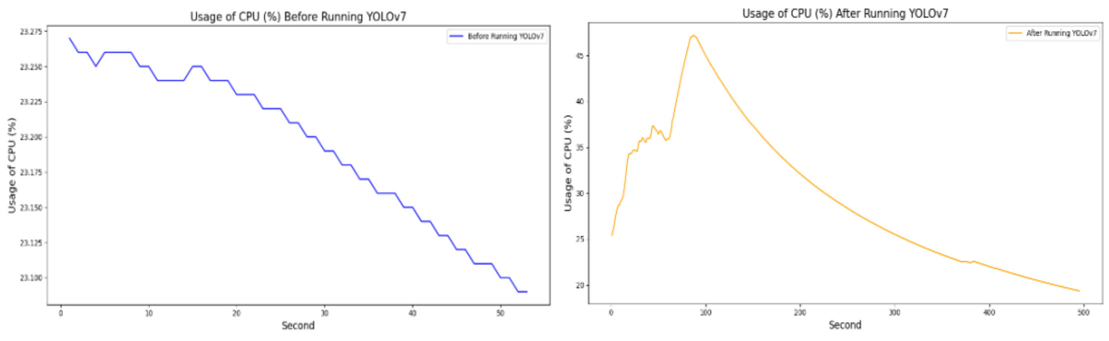


Figure 10. CPU usage, before detection (a), after detection (b)

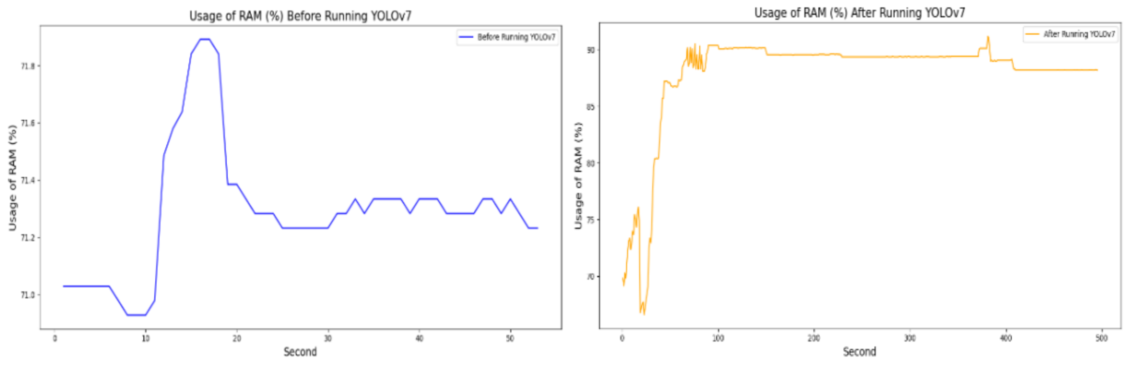


Figure 11. RAM usage, before detection (a), after detection (b)

Based on the graph above, it can be seen that there is a difference in the use of CPU resources on the Jetson Nano before and after the detection process, although it is slight. The Jetson Nano processes models with CPU usage ranging from 25-50%. The *real-time* detection process consumes the most CPU compared to detection using images and videos. The Jetson Nano consumes RAM resources when processing deep *learning* models by 20% from 70 to 90%. The use of the GPU when detecting the model cannot be done on the Jetson Nano 2GB.

2. System Testing

System testing is conducted using *the Black-Box testing framework*. The technical test can be seen in Table 5 and Table 6. Based on the test results, it can be concluded that all scenarios that have been created are successfully executed. The BlackBox testing framework enables comprehensive testing of the functionality of these object detection website and android application, as well as ensuring that the website and android application performs well according to the specified specifications.

3.6. *Multiplatform System Evaluation Results*

The framework testing was conducted utilizing the Blackbox testing system. The testing procedures can be seen in Table 5 and 6. Based on the test comes about, it can be concluded that all the made scenarios were effectively executed. The Blackbox testing system empowers comprehensive testing of the application's protest location usefulness and guarantees that the application performs well concurring to the desired prerequisites.

Also, the testing not as it were centered on the discovery comes about created by the framework but too on the system's responsiveness and solidness when working beneath different conditions. Testing was carried out by considering varieties in protest sorts and lighting conditions to guarantee the system's capacity to reliably identify leaf variations from the norm. Hence, Blackbox testing makes a difference recognize potential errors or inadequacies without the got to assess the source code, subsequently quickening the approval handle of the application's center capacities. The test comes about appear that the framework meets the anticipated execution and precision criteria and is solid for utilize in real-world situations.

Table 5. Black-Box Testing on Website

Action	Scenario	Status
Select the image input type	The user selects the image input type on the available select bar, so it will display the file upload field and the image prediction button	Worked
Select the video input type	The user selects the image input type on the available select bar, so it will display the file upload field and the video prediction button	Worked
Upload an image file	Users upload image files to make predictions	Worked
Upload a video file	Users upload files in the form of videos to make predictions	Worked
Press the image prediction button	The user presses the image prediction button. The detection page will display the successfully detected images. There is class, time, and accuracy information from the detected results.	Worked
Press the video prediction button	The user presses the video prediction button. The detection page will display the successfully detected images. There is class, time, and accuracy information from the detected results. If the image is more than 1, a loop will be created	Worked
Press the realtime camera button	The user presses the realtime camera button. The webpage page will open the camera and immediately detect objects in real-time	Worked
Upload the detection data to the google firestore server	In the detection process, at the end, the system will automatically upload the detected data to the google firestore server	Worked

Table 6. Black-Box Testing on Android Application

Page	Action	Scenario	Status
Initial Page	Display the initial page	The application displays the initial page containing the logo and app name text.	Worked
Main Page	Press the analysis button	The user presses the analysis button. The user is then directed to the analysis page, which contains graphs of detection results.	Worked

Analysis Page	Press the melon information button	The user presses the melon information button. The user is then directed to the information page about melons and abnormalities in melon leaves.	Worked
	Line graph displays data correctly	The application displays the line graph correctly, including x and y axis labels and accuracy values.	Worked
	Pie chart displays data correctly	The application displays the pie chart correctly, including class legends and proportions shown as percentage ratios.	Worked
Detection Page	Options for image detection, file upload, and real-time detection	The application displays results in the form of bounding boxes, object classes, and confidence scores for the detected objects.	Worked

3.7. *Discussion*

This research is a continuation of a previous study [17]. The YOLO-based method was chosen for its speed and low computational resource usage. The AVGHEQ method was selected because it can adaptively enhance image contrast while reducing noise and preserving image details. The implementation of patience and various augmentation techniques can improve the performance of the detection model. The highest results were obtained when applying augmentation during model training and disabling patience, achieving a mAP of 84.12%, accuracy of 91.19%, and a detection time of 4.55 ms. In object detection, increasing the number of epochs makes the model more robust, resulting in better detection outcomes. The model using the maximum *epoch* managed to outperform the model performance in previous studies which reached 48.85% in the Faster R-CNN model, 33.16% in the SSD model, and 16.56% in the YOLOv3 model [17].

The designed multiplatform melon leaf abnormality detection system can be used in real-time and is useful when applied in real-world environments such as greenhouses. This system is expected to detect abnormalities in melon leaves at an early stage, enabling users to make faster decisions regarding field conditions. The website can be accessed at the following link: https://melon-abnormality-detection.streamlit.app. Meanwhile, the application can be viewed and downloaded on the Play Store: https://ipb.link/meloanalyticsapp.

4. CONCLUSION

This research develops a multiplatform IoT-based system for detecting abnormalities in melon leaves, integrating Jetson Nano, a Streamlit-based website, and a mobile application for real-time agricultural monitoring. The system addresses the challenge of early abnormality detection to improve plant health management. Image preprocessing using Average Histogram Equalization (AVGHEQ) enhances image contrast while minimizing noise, and the YOLOv7 algorithm is employed for efficient detection. A dataset of 469 training images and 52 test images was validated through 5-fold cross-validation, achieving a mean Average Precision (mAP) of 84.12%, an accuracy of 91.19%, and a detection time of 4.55 milliseconds. System implementation on Jetson Nano resulted in 25-50% CPU usage and an increase in RAM usage from 70% to 90%, with real-time detection consuming the most resources. Functionality tests using blackbox testing confirmed the system's reliability. By combining preprocessing, robust modeling, and multiplatform deployment, the system offers an accessible and scalable solution for early detection of plant abnormalities, enabling timely intervention and improved crop health management.

ACKNOWLEDGEMENTS

The computations in this work were performed using the HPC facilities of the Computer Science Study Program, School of Data Science, Mathematics and Informatics, IPB University. The dataset used in this study was obtained from [17], sourced from the iSurf Lab (IoT for Smart Urban Farming Laboratory), Computer Science Study Program, School of Data Science, Mathematics and Informatics, IPB University, and Agribusiness and Technology Park IPB University.

REFERENCES

[1] S. I. Kusumaningrum, "Pemanfaatan Sektor Pertanian Sebagai Penunjang Pertumbuhan Perekonomian Indonesia," *J. Transaksi*, vol. 11, no. 1, pp. 80–89, 2019, [Online]. Available: http://ejournal.atmajaya.ac.id/index.php/transaksi/article/view/477

[2] G. Afriyanti, Ana Mariya, Charita Natalia, Sirat Nispuana, M. Farhan Wijaya, and M. Yoga Phalepi, "the Role of the Agricultural Sector on Economic Growth in Indonesia," *Indones. J. Multidiscip. Sci.*, vol. 2, no. 1, pp. 167–179, 2023, doi: 10.59066/ijoms.v2i1.325.

[3] BPS Provinsi Bengkulu, "Provinsi Bengkulu dalam Angka 2022," 2022.

[4] O. S. University, "Environmental factors affecting plant growth," 2024.

[5] D. A. Fitria and M. I. Riyadi, "Strategi Coping Stres Pada Petani Melon Pasca Gagal Panen di Desa Maguwan, Kecamatan Sambit, Kabupaten Ponorogo," *Rosyada Islam. Guid. Couns.*, vol. 3, no. 1, p. 51, 2022.

[6] E. W. Andrianto, N. Hidayat, and Suprapto, "Sistem Pakar Diagnosis Penyakit Pada Tanaman Jagung Menggunakan Metode Naive Bayes Berbasis Android," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 7, pp. 2738–2744, 2018, [Online]. Available: http://j-ptiik.ub.ac.id

[7] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Comput. Electron. Agric.*, vol. 147, no. July 2017, pp. 70–90, 2018, doi: 10.1016/j.compag.2018.02.016.

[8] C. S. R. Silva and J. M. Fonseca, *Artificial Intelligence and Algorithms in Intelligent Systems*, vol. 2. 2019. doi: 10.1007/978-3-319-91189-2_30.

[9] I. Goodfellow, Y. Bengio, and · Aaron Courville, "Deep Learning," *Foreign Aff.*, vol. 91, no. 5, pp. 1689–1699, 2016.

[10] F. Chollet, *Deep Learning with Python*. New York: Manning Publications Co., 2018. doi: 10.23919/ICIF.2018.8455530.

[11] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," pp. 1–39, 2019, [Online]. Available: http://arxiv.org/abs/1905.05055

[12] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," pp. 1–15, 2022, [Online]. Available: http://arxiv.org/abs/2207.02696

[13] J. Nano, "Jetson Nano Developer Kit," 2023.

[14] C. Tan, "Jetson Nano vs Raspberry Pi 4: The Differences," *Https://All3Dp.Com/2/Raspberry-Pi-Vs-Jetson-Nano-Differences/*, 2021.

[15] Y. Li, J. Wang, H. Wu, Y. Yu, H. Sun, and H. Zhang, "Detection of powdery mildew on strawberry leaves based on DAC-YOLOv4 model," *Comput. Electron. Agric.*, vol. 202, no. October, p. 107418, 2022, doi: 10.1016/j.compag.2022.107418.

[16] Y. Xu *et al.*, "Real-time object detection method of melon leaf diseases under complex background in greenhouse," *J. Real-Time Image Process.*, vol. 19, no. 5, pp. 985–995, 2022, doi: 10.1007/s11554-022-01239-7.

[17] H. Rahmat, S. Wahjuni, and H. Rahmawan, "Performance Analysis of Deep Learning-based Object Detectors on Raspberry Pi for Detecting Melon Leaf Abnormality," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 12, no. 2, pp. 572–579, 2022, doi: 10.18517/ijaseit.12.2.13801.

[18] S. C. F. Lin *et al.*, "Image enhancement using the averaging histogram equalization (AVHEQ) approach for contrast improvement and brightness preservation," *Comput. Electr. Eng.*, vol. 46, pp. 356–370, 2015, doi: 10.1016/j.compeleceng.2015.06.001.

[19] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image Data Augmentation for Deep Learning: A Survey," 2022, [Online]. Available: http://arxiv.org/abs/2204.08610

[20] T. Kumar, M. Turab, K. Raj, A. Mileo, R. Brennan, and M. Bendechache, "Advanced Data Augmentation Approaches: A Comprehensive Survey and Future directions," vol. 6223, 2023, [Online]. Available: http://arxiv.org/abs/2301.02830

[21] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Glob. Transitions Proc.*, vol. 3, no. 1, pp. 91–99, 2022, doi: 10.1016/j.gltp.2022.04.020.

[22] D. Berrar, "Cross-validation," *Encycl. Bioinforma. Comput. Biol. ABC Bioinforma.*, vol. 1–3, no. January 2018, pp. 542–545, 2018, doi: 10.1016/B978-0-12-809633-8.20349-X.

[23] A. Salazar-Gomez, M. Darbyshire, J. Gao, E. I. Sklar, and S. Parsons, "Beyond mAP: Towards practical object detection for weed spraying in precision agriculture," *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2022-Octob, pp. 9232–9238, 2022, doi: 10.1109/IROS47612.2022.9982139.