Indonesian Journal of Electrical Engineering and Computer Science
Vol. 34, No. 2, May 2024, pp. 1204~1213
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v34.i2.pp1204-1213

Refining tomato disease recognition: hyperparameter tuning on ResNet-101 architecture for precise leaf-based classification

Tegar Arifin Prasetyo, Tiurma Lumban Gaol, Nico Felix Sipahutar, Tessalonika Siahaan, Trito Exaudi Manik, Rudy Chandra
Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del, Toba, Indonesia

Article Info

Article history:

Received Dec 4, 2023
Revised Feb 7, 2024
Accepted Feb 9, 2024

Keywords:

Convolutional neural network
Hyperparameter tuning
Intelligent classification
Residual network-101
Tomato plant disease

ABSTRACT

Tomatoes plants are widely recognized as versatile vegetables globally. This study aims to develop a high-precision web interface for classifying various leaf diseases in tomatoes. Utilizing a convolutional neural network (CNN) algorithm using the residual network-101 (ResNet-101) architecture, this tool aids farmers in accurately identifying leaf diseases in tomatoes, thereby reducing the risk of crop failure. The dataset comprises 6,800 images, categorized into four classes: early blight, spider mites two spotted, tomato yellow leaf curl virus, and healthy tomatoes, each containing 1,700 images. Hyperparameter tuning was conducted as part of an experiment aimed at enhancing the performance of the model. Experiments involved varying epoch values (10, 25, 30, 50, 60, 75, 100, and 110), a fixed batch size of 4, different learning rates (0.1, 0.01, 0.001, 0.0001), and num workers (4, 8, 16). The results demonstrated an accuracy of 99% with 100 epochs, a batch size of 4, a learning rate of 0.001, and 16 num workers. Consequently, this research contributes to a deeper understanding of disease management in tomato plants, ensuring optimal quality and quantity of the harvest.

Corresponding Author:

Tegar Arifin Prasetyo
Department of Information Technology, Faculty of Vocational Studies, Institut Teknologi Del
Sisingamangaraja Street, Sitoluama, Laguboti 22381, Toba, Indonesia
Email: tegar.prasetyo@del.ac.id

1. INTRODUCTION

Tomatoes are among the most widely used vegetables worldwide. Belonging to the Solanaceae family, tomatoes are one of the most adaptable plant products that can be cultivated year after year [1]. This plant can be widely cultivated, ranging from lowland areas to surrounding regions. According to the Central Statistics Agency report in 2022, tomato production in Indonesia reached a total of 1,116,740 tons. Tomatoes can be grown in almost all regions of the country. Nevertheless, tomatoes are susceptible to various diseases, which can have an impact on both the quality and quantity of the tomato plants [2]. Accurately classifying tomato diseases based on leaf symptoms is a considerable challenge, given the numerous factors at play, such as fungal, bacterial, and viral infections [3], [4]. Although symptoms like brown spots or leaf curling may appear alike in various diseases like early blight, two-spotted spider mites, and tomato yellow leaf curl virus in tomato plants, distinguishing them requires meticulous examination [5]–[7] and expertise to pinpoint specific patterns, lesions, and accompanying signs, essential for precise diagnosis and effective management strategies. Early Blight is a type of disease that results in small, scattered, and irregularly shaped dark brown spots on infected leaves, extending to young leaves. Two-spotted spider mites refer to a leaf disease that causes the appearance of white spots on the upper side of leaves, caused by spider mites, leading to wilting, and tearing of the leaves. Tomato yellow leaf curl virus indicates a disease infected by whiteflies, characterized by the curling, or twisting of young leaves and yellowing, while older leaves do not experience shrinking.

Applying classification techniques across various facets of human life emerges as a key solution to problem-solving [8]–[10]. The integration of artificial intelligence in computer vision further enriches the presentation of valuable information to humans through visual representations, encompassing diverse subjects such as animals, plants, and more [11]. This research provides invaluable insights into the identification and classification of diverse objects, with a specific focus on diseases that affect tomato plants. Through the utilization of a case study approach, its objective is to accurately categorize the common leaf diseases commonly found in tomatoes. The manual surveillance of these diseases poses significant challenges due to limited farm labor availability and restricted working hours, thereby increasing the likelihood of errors. Addressing this research gap will undoubtedly strengthen disease management strategies in tomato cultivation.

In recent years, there has been a significant shift in research focus towards the application of machine learning and deep learning in the realm of accurately identifying and classifying plant diseases through advanced image analysis techniques [12]–[15]. In the realm of deep learning, notable progress has been made, specifically in convolutional neural networks (CNNs) [16]. This recent advancement has resulted in significant breakthroughs spanning various domains, with one noteworthy application being the accurate classification of plant diseases. The categorization of nine plant species was carried out using a combination of AlexNet and support vector machine, as detailed in the work by [17], resulting in an impressive accuracy of 91.15%. Liu *et al.* [18] constructed a ten-layer CNN model to plant leaf classification, achieving an accuracy of 87.92% when tested with the Flavia dataset with 4,800 leaf images and 32 kinds of leaf. Utilizing AlexNet and visual geometry group (VGG)16 Net, Rangarajan *et al.* [19] successfully classified images of tomato plant diseases into seven types: healthy, late blight, leaf mold, two-spotted spider mite attack, target spot, tomato mosaic virus disease, and tomato yellow leaf curl virus disease with the result of the classification used AlexNet is 97.49% and VGG16 Net is 97.29%. Zhang *et al.* [20] employed the ResNet-50 model for the detection of tomato leaf disease, demonstrating a commendable accuracy level of 97.28%. In evaluating tomato plant disease classification, Aquil and Ishak [21] attained a peak F1-score of 99.92% with ResNet-101, considering both scratch and pre-trained CNNs. In addition, Vengaiah and Priyadharshini [22] demonstrated that ResNet-101 outperformed VGGNet in terms of accuracy across the majority of the evaluated criteria during their implementation and analysis of both models. Incorporating insights from diverse prior studies, the consensus emerges that the pretrained CNN model featuring ResNet-101 architecture displays robust performance in effectively classifying tomato plant diseases.

ResNet-101 showcases a diverse set of hyperparameters, offering the flexibility to achieve distinct classification accuracies when applied to the same tasks with varying hyperparameter setups. Trivedi *et al.* [23] employed a CNN to categorize tomato diseases based on the analysis of leaf morphology. The study included a comprehensive exploration of hyperparameters, with a specific focus on optimizing epoch duration, batch size, and learning rate for the training algorithm with an accuracy of 98.49%. Takase [24] systematically explored the influence of varying batch sizes on the performance of CNNs across diverse datasets in their research study. Xu [25] conducted experiments involving hyperparameter optimization, specifically focusing on the learning rate, batch size, and number of epochs. The experiment found a noteworthy reduction in the error of the trained network model as the learning rate increased, batch size decreased, and epochs accumulated within a defined range, while maintaining constant values for other hyperparameters. In the research conducted by [26], the emphasis was on elucidating the crucial significance of hyperparameter tuning in machine learning. This involves defining hyperparameters, including those of neural networks, before execution, and optimizing these parameters during the training phase is pivotal for achieving peak performance and reducing generalization errors. Accordingly, the optimization of hyperparameters holds the potential to improve image classification accuracy when utilizing the CNN architecture with a pretrained ResNet-101 model.

ResNet constitute a convolutional network that has undergone training on a dataset exceeding 1 million images sourced from the ImageNet database [27] and ResNet-101 model boasts a pre-trained neural network capable of classifying images across 1,700 object categories, making it proficient in learning diverse and effective feature representations for various images. Our research brings notable contributions to the field, particularly in devising a novel CNN model tailored for the classification of diseases in tomato plants. Furthermore, we conduct extensive optimization of hyperparameters to bolster the performance of the pretrained ResNet-101 model in accurately categorizing tomato plant diseases based on leaf images. Additionally, our study introduces a user-friendly web interface for the classification of diseases in tomato plants, providing labeled results and detailed accuracy information. In our research, we conducted extensive experiments to fine-tune the hyperparameters and optimize the model's performance. This encompassed exploring a spectrum of epoch values ranging from 10 to 110, maintaining a consistent batch size of 4, experimenting with various learning rates spanning from 0.1 to 0.0001, and adjusting the number of workers from 4 to 16. Our findings revealed a remarkable achievement of 99% accuracy attained with 100 epochs, a batch size of 4, a learning rate of 0.001, and 16 workers. Through this meticulous inquiry, we deepen our understanding of disease management in tomato plants, ultimately contributing to the enhancement of crop quality and yield optimization.

2. METHOD

The research methodology's design, as depicted in Figure 1. It provides a clear understanding of its systematic approach. This concise elucidation improves transparency regarding the study's methodological framework, facilitating comprehension.

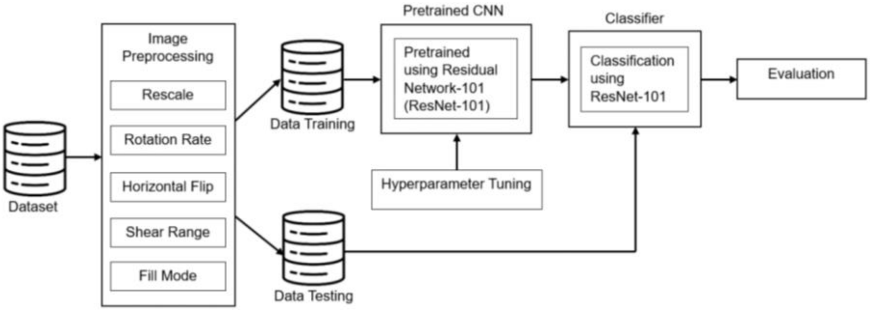


Figure 1. The schematic representation of the research methodology design

Based on Figure 1, the research methodology for classifying disease types on tomato leaves employs a pre-trained CNN, specifically ResNet-101. It commences with dataset collection, encompassing images of healthy and variously infected tomato leaves. These images undergo preprocessing stages to augment data variability and prevent overfitting, involving rescaling, rotation, horizontal flipping, shearing, and fill mode adjustments to handle new pixel formations. Post-preprocessing, the images are divided into training and testing sets. The training phase includes hyperparameter tuning to optimize ResNet-101 model performance. Subsequently, the trained model is applied to classify images in the testing set. Finally, model efficacy is evaluated using metrics like accuracy, precision, and recall gauging its ability to identify and differentiate between different tomato leaf diseases.

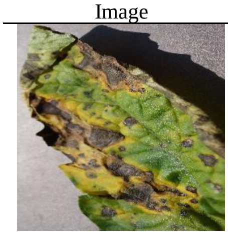
2.1. Tomato plant disease based on leaf image dataset

The data to be used is sourced from a dataset provider that offers image datasets. For this study, the dataset source will be Kaggle, a platform known for its diverse datasets. The choice of Kaggle is based on its alignment with the specific requirements of this research. The data extraction from the Kaggle dataset is chosen because the selected dataset provides information relevant to the research needs. In this study, the data is categorized into three sets: training data, validation data, and testing data, with a distribution ratio of 70:20:10, respectively. The total dataset comprises 6,800 images, divided into training data (70%) with 4,760 images, including 1,190 images for each of the four disease classes; validation data (20%) with 1,360 images, including 340 images for each disease class; and testing data (10%) with 680 images, including 170 images for each disease class. During dataset collection, specific criteria are applied for image selection in both training and testing phases. The characteristics of the training data consist of four classes: early blight, spider mites two spotted, tomato yellow leaf curl virus, and healthy. The dataset is collected by gathering images from each leaf object, and once collected, the images are organized into folders to facilitate leaf object categorization. Examples of images for dataset collection can be observed in Table 1.

2.2. Image preprocessing

In the image preprocessing implementation phase, all datasets that have previously undergone labeling and splitting processes will be processed. In this study, the data splitting process is performed manually by categorizing all data into 3 sets and 4 disease classes. The first step in the data preprocessing stage involves creating a path for each dataset, which will be useful for storing data that has undergone the preprocessing process [28]. During the dataset preprocessing stage, we employ the image data generator to perform data augmentation, thereby enhancing the variety of images. The image data generator comprises several parameters that preprocess input data for model construction. The parameters utilized in this stage include: (i) rescale with a value of 1. /255: this parameter normalizes the pixel values of the images. In this study, the images have RGB coefficients ranging from 0 to 255. However, to ensure compatibility with the model, which expects values between 0 and 1, rescaling is performed by dividing each pixel value by 255. (ii) Rotation range with a value of 20: this parameter is employed to randomly rotate images by up to 20 degrees. (iii) Horizontal flip with a value of ‘True’: this parameter facilitates horizontal flipping or mirroring of images. (iv) Shear range with a value of 0.2: this parameter induces a shear transformation by shifting the angle by 0.2. (v) Fill mode with a value of ‘Nearest’: this parameter is utilized to fill newly created pixels that may appear during image transformations with the nearest pixel value from the original image.

Table 1. An illustration of a tomato plant disease derived from leaf images within the dataset

Image	Characteristics type	Image	Characteristics type
	Early blight		Spider mites two spotted
	Tomato yellow leaf curl virus		Healthy

2.3. Pretrained model

Various researchers have employed ResNet-101 for image classification, yielding commendable results in the classification outcomes [29]. In this research, the first part of the ResNet-101 architecture is conv1, consisting of a conv2D layer with 64 filters, a kernel size of 7×7, and a stride of 2×2. It utilizes the same padding or same layer. Subsequently, batch normalization is applied, followed by rectified linear units (ReLU) activation. A max pooling layer is then added to the model with a pool size of 3×3. This process continues until reaching the conv5 stage. However, in the final steps, two additional layers are introduced: GlobalAveragePooling2D (to obtain the average of all features with a 4D tensor) and flatten (to convert the 4D tensor to 1D). Additionally, a SoftMax activation is employed to generate output in the form of probabilities for each class. The SoftMax activation is utilized when there are more than two classes in the output of the dense layer. However, if only two classes are present, a sigmoid activation is used. The sigmoid activation is beneficial for calculating the probabilities of the output. The outcome of this process is a model that will be employed for classifying leaf diseases in tomato plants. This architecture can be visualized in Figure 2.

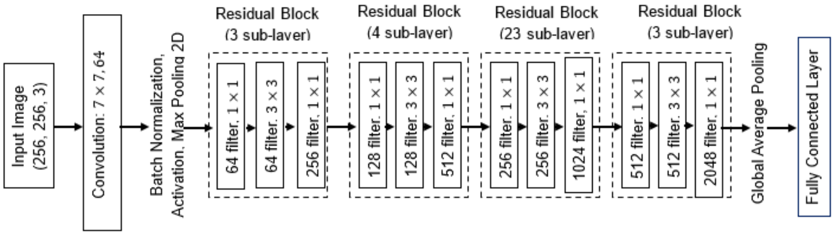


Figure 2. The architecture of ResNet-101

2.4. Hyperparameter tuning

In this phase, hyperparameter optimization is conducted through a manual tuning approach, wherein the hyperparameter values are determined manually by combining various hyperparameters [30]. The hyperparameters considered in this study encompass the learning rate, batch size, and epoch. The research methodology involves an experimental approach for investigation. In the experimentation phase of this study, diverse sets of experiments were conducted by altering epoch values (10, 25, 30, 50, 60, 75, 100, and 110), maintaining a constant batch size of 4, exploring various learning rates (0.1, 0.01, 0.001, 0.0001), and adjusting the number of workers (4, 8, 16).

3. RESULTS AND DISCUSSION

The results and discussions sections provide a detailed analysis of the outcomes resulting from the implemented system. This segment specifically elaborates on the implementation of the designed model, encompassing details such as the implementation environment, encountered constraints, the libraries utilized, the approach to hyperparameter tuning, the execution of the CNN model using ResNet-101 architecture, and a comprehensive evaluation of the model. Additionally, it includes an in-depth examination of the results derived from each classification process conducted in this research.

3.1. Result of building model

ResNet-101 exhibits structural complexity, comprising layers involving convolution, normalization, activation (ReLU), and addition (skip connection). In this study, the implementation of ResNet-101 begins with an input layer of dimensions (None, 256, 256, 3). The model undergoes its initial transformation in the first convolutional layer (conv1), yielding an output of (None, 128, 128, 64) with 9,472 parameters. Subsequently, batch normalization (bn_conv1) and an activation layer (activation_100) follow. Max pooling (max_pooling2d_1) ensues, producing an output of (None, 63, 63, 64), before entering a series of convolutional blocks (conv2_x, conv3_x, conv4_x). Each block consists of convolution, batch normalization, activation, and skip connection. After passing through these blocks, the ResNet-101 model generates a final output with dimensions (None, 16, 16, 1024). This process enables the model to process input data in the form of 3×256×256 RGB images and produce output consistent with its structure. The total number of parameters in the model is calculated based on the contribution of each layer during the transformation process. The ResNet-101 model summary is presented in Table 2.

Table 2. The ResNet-101 model summary

Layer (type)	Output shape	Params	Layer (type)	Output shape	Params
Input layer	(None, 256, 256, 3)]	0	Conv2d	(None, 16, 16, 256)	590,080
Conv1	(None, 128, 128, 64)	9472	Bn_155	(None, 16, 16, 256)	1,024
Bn_conv1	(None, 128, 128, 64)	256	Activation_150	(None, 16, 16, 256)	0
Activation_100	(None, 128, 128, 64)	0	Conv2d_106	(None, 16, 16, 1024)	263,168
Max_pooling2d_1	(None, 63, 63, 64)	0	Bn_156	(None, 16, 16, 1024)	4,096
Conv2_1	(None, 63, 63, 64)	4160	Add_49	(None, 16, 16, 1024)	0
Bn_103	(None, 63, 63, 64)	256	Activation_151	(None, 16, 16, 1024)	0
Activation_101	(None, 63, 63, 64)	0	Conv4_11	(None, 16, 16, 256)	262,400
Conv2d_70	(None, 63, 63, 64)	36928	Bn_157	(None, 16, 16, 256)	1,024
…	…	…	…	…	…
Bn_151	(None, 16, 16, 256)	1024	Conv2d_138	(None, 8, 8, 512)	2,359,808
Activation_146	(None, 16, 16, 256)	0	Bn_204	(None, 8, 8, 512)	2,048
Conv2d_103	(None, 16, 16, 256)	590080	Activation_198	(None, 8, 8, 512)	0
Bn_152	(None, 16, 16, 256)	1024	Conv2d_139	(None, 8, 8, 2048)	1,050,624
Activation_147	(None, 16, 16, 256)	0	Bn_205	(None, 8, 8, 2048)	8,192
Conv2d_104	(None, 16, 16, 1024)	263168	Add_65	(None, 8, 8, 2048)	0
Bn_153	(None, 16, 16, 1024)	4096	Activation_199	(None, 8, 8, 2048)	0
Add_48	(None, 16, 16, 1024)	0	Avg_pool	(None, 2048)	0
Activation_148	(None, 16, 16, 1024)	0	Flatten_1	(None, 2048)	0
Conv4_10	(None, 16, 16, 256)	262400	Dense_1	(None, 4)	8,196
Bn_154	(None, 16, 16, 256)	1024	Total params: 42,666,372		
Activation_149	(None, 16, 16, 256)	0	Trainable params: 42,561,028		
			Non-trainable params: 105,344		

Table 2 reveals that the total number of parameters is 42,666,372, representing the parameters trained by the machine. Parameters are categorized into two types: trainable parameters, whose values are calculated and can be updated during the training process, and non-trainable parameters, whose values remain unchanged throughout training. The total number of parameters that do not undergo updates during training is 42,561,028, with 105,344 parameters that remain unchanged. The number of parameters in a model can vary based on architecture and complexity, and different architectures have different parameter counts.

3.2. Result of hyperparameter tuning

The careful selection of hyperparameters tuning significantly influences the effectiveness of CNN training, playing a crucial role in determining the model's performance. During the training phase, we utilized a dataset split, allocating 70% with 1190 data points per class. For validation, 20% of the dataset was used, containing 340 data points per class, leaving the remaining 10% for testing purposes. Our experimentation involved varying epoch values (10, 25, 30, 50, 60, 75, 100, and 110), maintaining a fixed batch size of 4, exploring different learning rates (0.1, 0.01, 0.001, 0.0001), and adjusting the number of workers (4, 8, 16).

Table 3 shows that in the 31st experiment, featuring 100 epochs, a batch size of 4, a learning rate of 0.0001, and num workers is 16, the model achieved optimal performance, showcasing superior accuracy compared to other experiments. Experiment 31 stands out as the best-performing trial, exhibiting excellent results in terms of loss, accuracy, validation loss, and validation accuracy, with a minimal loss of 0.00 and an accuracy of 0.99. These outcomes signify that the model from 31 experiment makes predictions with high accuracy and adeptly adapts to the training data.

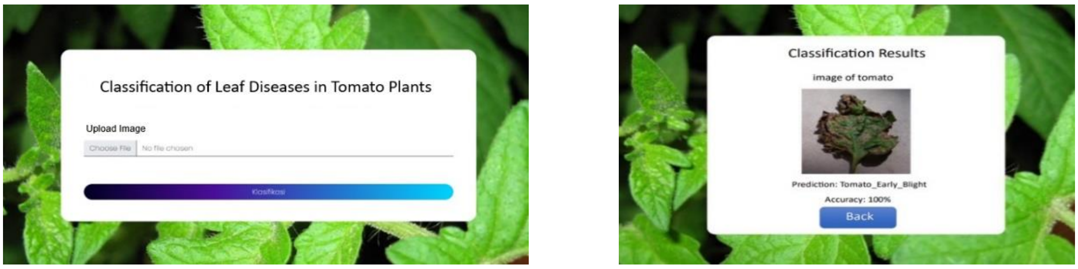
Based on Table 3, a lower training loss value indicates the model's proficiency in understanding patterns within the training data, thus enhancing its ability to predict test data accurately. Consequently, experiment 31 emerges as the most successful model in learning patterns from the training data. It provides precise predictions for the test data, reflecting its robust performance.

Table 3. Hyperparameter tuning experiment

Number	Epoch	Batch size	Learning rate	Num worker	Loss	Validation loss	Accuracy	Validation accuracy
1	10	4	0.1	4	0.80	0.80	0.71	0.71
2	25	4	0.1	4	0.13	0.13	0.96	0.96
3	30	4	0.1	8	0.29	0.29	0.93	0.93
4	50	4	0.1	16	0.00	0.00	0.98	0.98
5	60	4	0.1	16	0.09	0.09	0.96	0.96
6	75	4	0.1	16	0.19	0.19	0.93	0.93
7	100	4	0.1	16	0.42	0.42	0.93	0.93
8	110	4	0.1	16	0.08	0.08	0.97	0.97
9	10	4	0.01	8	1.26	1.26	0.62	0.62
10	25	4	0.01	4	0.03	0.03	0.98	0.98
12	50	4	0.01	16	0.02	0.02	0.98	0.98
13	60	4	0.01	16	0.26	0.26	0.92	0.92
14	75	4	0.01	16	0.04	0.04	0.98	0.98
15	100	4	0.01	16	0.14	0.14	0.97	0.97
16	110	4	0.01	16	0.01	0.01	0.98	0.98
17	10	4	0.001	8	0.52	0.52	0.79	0.79
18	25	4	0.001	8	1.28	1.28	0.73	0.73
19	30	4	0.001	8	0.01	0.01	0.96	0.96
20	50	4	0.001	8	0.03	0.03	0.97	0.97
21	60	4	0.001	16	0.02	0.02	0.97	0.97
22	75	4	0.001	16	0.12	0.12	0.97	0.97
23	100	4	0.001	16	0.02	0.02	0.92	0.92
24	110	4	0.001	16	0.01	0.01	0.98	0.98
25	10	4	0.0001	8	0.30	0.30	0.89	0.89
26	25	4	0.0001	8	0.14	0.14	0.95	0.95
27	30	4	0.0001	8	0.07	0.07	0.97	0.97
28	50	4	0.0001	16	0.29	0.29	0.90	0.90
29	60	4	0.0001	16	1.05	1.05	0.77	0.77
30	75	4	0.0001	16	0.01	0.01	0.98	0.98
31	100	4	0.0001	16	0.00	0.00	0.99	0.99
32	110	4	0.0001	16	0.02	0.02	0.98	0.98

3.3. Result of website interface

This section presents the outcomes of the custom-built web interface, designed to optimize user interaction with the CNN model leveraging the ResNet-101 architecture for precise disease classification in tomato plants using leaf images. Figure 3 shows that website interface for tomato plant disease based on leaf classification. As depicted in Figure 3(a), the user-friendly interface for image input includes a convenient image drop box, simplifying the image upload process. Upon uploading, the advanced image prediction unfolds, displaying both the submitted images and their corresponding "Processing" status. Figure 3(b) showcases the output web interface, providing a detailed view of accuracy results and labels for previously uploaded or input images. This carefully crafted system ensures seamless and efficient utilization of the CNN model, setting a new standard for accessibility and effectiveness in tomato plant disease classification.



Prediction: Tomato_Early_Blight
Accuracy: 100%

(a) (b)

Figure 3. Web interface for tomato plant disease based on leaf classification (a) input or upload image and (b) result image classification

Additionally, the data from several tests for the healthy leaf class and leaves with diseases, such as early blight, tomato yellow leaf curl virus, and two-spotted tomato spider mites on tomato leaves, encompass various aspects. These include image outcomes, detection results, and accuracy results. All of these are meticulously compiled and presented in Table 4 for comprehensive analysis.

Table 4. Test result testing of early blight, tomato yellow leaf curl virus, two-spotted tomato spider mites, and healthy

Image	Detected disease types	Accuracy	Image	Detected disease types	Accuracy
	Early blight	99.9%		Tomato yellow leaf curl virus	99.9%
	Early blight	100%		Tomato yellow leaf curl virus	100%
	Early blight	99.7%		Tomato yellow leaf curl virus	100%
	Tomato spider mites two-spotted	99.9%		Healthy	100%
	Tomato spider mites two-spotted	99.9%		Healthy	100%
	Tomato spider mites two-spotted	99.9%		Healthy	99.9%

3.4. Evaluation

The confusion matrix serves as a crucial tool for assessing the effectiveness of a classification model by visually representing the predicted outcomes against the actual data. This evaluation, conducted on the testing dataset consisting of 170 data points per class, offers insights into the model’s predictive capabilities. Figure 4 offers a detailed breakdown, illustrating the number of accurately classified and misclassified data points for each class, providing a comprehensive overview of the model's performance across different categories.

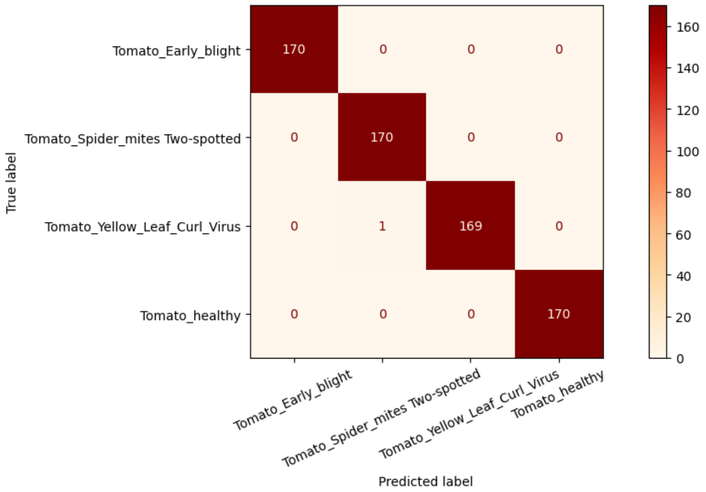


Figure 4. Plotting predicted class

From Figure 4, it is observed that each class is detected as different classes. However, for the tomato yellow leaf curl virus class, there is one misclassified data, resulting in a count of 169. Consequently, various model evaluation metrics can be calculated, such as accuracy, precision, recall, and F1-score, which are useful for assessing model performance. The testing results of the model using testing data indicate an accuracy of 99%. This high accuracy level is noteworthy as the model demonstrates proficiency in recognizing leaf disease characteristics in tomato plants, effectively minimizing the likelihood of misclassification errors.

The accuracy obtained is 0.99, indicating that the model can correctly predict data 99% of the time from the total test data. A recall value of 0.99 shows that the model excels in identifying or classifying true positive instances, achieving a success rate of 99%. A precision value of 1.00 signifies that all instances predicted as positive are indeed positive among the entire test data. The F1-score of 0.99 reflects a high level of accuracy and balance between recall and precision, indicating that the model can predict data correctly 99% of the time from the total test data.

4. CONCLUSION

Based on the findings of our research, several significant conclusions can be drawn. Firstly, the study successfully constructed a model using CNN with the ResNet-101 architecture to classify leaf diseases in tomato plants. Secondly, through experimentation with a dataset containing 170 image data points per class, the research determined optimal parameters, resulting in an epoch value of 100, batch size of 4, learning rate of 0.0001, and 16 num workers. The model trained with this configuration achieved a 99% accuracy rate in classifying the four types of tomato leaf diseases. Additionally, a user-friendly web interface was developed to facilitate users in classifying leaf diseases in tomato plants. Looking ahead, it is recommended that future research focuses on the system’s ability to recognize multiple leaf objects in a single image and introduces more diverse image data in each leaf disease class to enhance the system’s capabilities and diversity.

ACKNOWLEDGEMENTS

This work was supported in part by Del Institute of Technology, Indonesia.

REFERENCES

[1] M. M. Gitau, A. Farkas, V. Ördög, and G. Maróti, “Evaluation of the biostimulant effects of two Chlorophyta microalgae on tomato (Solanum lycopersicum),” *Journal of Cleaner Production*, vol. 364, p. 132689, Sep. 2022, doi: 10.1016/j.jclepro.2022.132689.

[2] M. Ma, P. W. J. Taylor, D. Chen, N. Vaghefi, and J. Z. He, “Major soilborne pathogens of field processing tomatoes and management strategies,” *Microorganisms*, vol. 11, no. 2, p. 263, Jan. 2023, doi: 10.3390/microorganisms11020263.

[3] P. Yang, L. Zhao, Y. G. Gao, and Y. Xia, “Detection, diagnosis, and preventive management of the bacterial plant pathogen pseudomonas syringae,” *Plants*, vol. 12, no. 9, p. 1765, Apr. 2023, doi: 10.3390/plants12091765.

[4] S. M. Javidan, A. Banakar, K. A. Vakilian, and Y. Ampatzidis, “Tomato leaf diseases classification using image processing and weighted ensemble learning,” *Agronomy Journal*, Mar. 2023, doi: 10.1002/agj2.21293.

[5] S. G. Paul *et al.*, “A real-time application-based convolutional neural network approach for tomato leaf disease classification,” *Array*, vol. 19, p. 100313, Sep. 2023, doi: 10.1016/j.array.2023.100313.

[6] T. Sanida, A. Sideris, M. V. Sanida, and M. Dasygenis, “Tomato leaf disease identification via two–stage transfer learning approach,” *Smart Agricultural Technology*, vol. 5, p. 100275, Oct. 2023, doi: 10.1016/j.atech.2023.100275.

[7] A. Guerrero-Ibañez and A. Reyes-Muñoz, “Monitoring tomato leaf disease through convolutional neural networks,” *Electronics (Switzerland)*, vol. 12, no. 1, p. 229, Jan. 2023, doi: 10.3390/electronics12010229.

[8] A. Aldoseri, K. N. Al-Khalifa, and A. M. Hamouda, “Re-thinking data strategy and integration for artificial intelligence: concepts, opportunities, and challenges,” *Applied Sciences (Switzerland)*, vol. 13, no. 12, p. 7082, Jun. 2023, doi: 10.3390/app13127082.

[9] O. Ali, W. Abdelbaki, A. Shrestha, E. Elbasi, M. A. A. Alryalat, and Y. K. Dwivedi, “A systematic literature review of artificial intelligence in the healthcare sector: benefits, challenges, methodologies, and functionalities,” *Journal of Innovation and Knowledge*, vol. 8, no. 1, p. 100333, Jan. 2023, doi: 10.1016/j.jik.2023.100333.

[10] M. M. Taye, “Understanding of machine learning with deep learning: architectures, workflow, applications and future directions,” *Computers*, vol. 12, no. 5, p. 91, Apr. 2023, doi: 10.3390/computers12050091.

[11] J. M. Górriz *et al.*, “Computational approaches to explainable artificial intelligence: advances in theory, applications and trends,” *Information Fusion*, vol. 100, p. 101945, Dec. 2023, doi: 10.1016/j.inffus.2023.101945.

[12] S. F. Ahmed *et al.*, “Deep learning modelling techniques: current progress, applications, advantages, and challenges,” *Artificial Intelligence Review*, vol. 56, no. 11, pp. 13521–13617, Nov. 2023, doi: 10.1007/s10462-023-10466-8.

[13] J. Kotwal, D. R. Kashyap, and D. S. Pathan, “Agricultural plant diseases identification: from traditional approach to deep learning,” *Materials Today: Proceedings*, vol. 80, pp. 344–356, 2023, doi: 10.1016/j.matpr.2023.02.370.

[14] N. Ullah *et al.*, “An effective approach for plant leaf diseases classification based on a novel DeepPlantNet deep learning model,” *Frontiers in Plant Science*, vol. 14, Oct. 2023, doi: 10.3389/fpls.2023.1212747.

[15] A. Ahmad, D. Saraswat, and A. El Gamal, “A survey on using deep learning techniques for plant disease diagnosis and recommendations for development of appropriate tools,” *Smart Agricultural Technology*, vol. 3, p. 100083, Feb. 2023, doi: 10.1016/j.atech.2022.100083.

[16] T. A. Kumar, R. Rajmohan, S. A. Ajagbe, T. Gaber, X. J. Zeng, and F. Masmoudi, “A novel CNN gap layer for growth prediction of palm tree plantlings,” *PLoS ONE*, vol. 18, no. 8 August, p. e0289963, Aug. 2023, doi: 10.1371/journal.pone.0289963.

[17] S. A. Wagle and R. Harikrishnan, “Comparison of plant leaf classification using modified alexnet and support vector machine,” *Traitement du Signal*, vol. 38, no. 1, pp. 79–87, Feb. 2021, doi: 10.18280/TS.380108.

[18] J. Liu, S. Yang, Y. Cheng, and Z. Song, “Plant leaf classification based on deep learning,” in *Proceedings 2018 Chinese Automation Congress, CAC 2018*, Nov. 2018, pp. 3165–3169, doi: 10.1109/CAC.2018.8623427.

[19] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, “Tomato crop disease classification using pre-trained deep learning algorithm,” *Procedia Computer Science*, vol. 133, pp. 1040–1047, 2018, doi: 10.1016/j.procs.2018.07.070.

[20] K. Zhang, Q. Wu, A. Liu, and X. Meng, “Can deep learning identify tomato leaf disease?,” *Advances in Multimedia*, vol. 2018, pp. 1–10, Sep. 2018, doi: 10.1155/2018/6710865.

[21] M. A. I. Aquil and W. H. W. Ishak, “Evaluation of scratch and pre-trained convolutional neural networks for the classification of tomato plant diseases,” *IAES International Journal of Artificial Intelligence*, vol. 10, no. 2, pp. 467–475, Jun. 2021, doi: 10.11591/IJAI.V10.I2.PP467-475.

[22] C. Vengaiah and M. Priyadharshini, “CNN model suitability analysis for prediction of tomato leaf diseases,” in *2023 6th International Conference on Information Systems and Computer Networks, ISCON 2023*, Mar. 2023, pp. 1–4, doi: 10.1109/ISCON57294.2023.10111996.

[23] N. K. Trivedi *et al.*, “Early detection and classification of tomato leaf disease using high-performance deep neural network,” *Sensors*, vol. 21, no. 23, p. 7987, Nov. 2021, doi: 10.3390/s21237987.

[24] T. Takase, “Dynamic batch size tuning based on stopping criterion for neural network training,” *Neurocomputing*, vol. 429, pp. 1–11, Mar. 2021, doi: 10.1016/j.neucom.2020.11.054.

[25] N. Xu, “Research on the influence of convolutional neural network parameters on fruit classification,” in *2019 IEEE 3rd International Conference on Electronic Information Technology and Computer Engineering, EITCE 2019*, Oct. 2019, pp. 606–610, doi: 10.1109/EITCE47263.2019.9094902.

[26] P. Probst, A. L. Boulesteix, and B. Bischl, “Tunability: importance of hyperparameters of machine learning algorithms,” *Journal of Machine Learning Research*, vol. 20, 2019.

[27] M. A. Rafidison *et al.*, “Image classification based on light convolutional neural network using pulse couple neural network,” *Computational Intelligence and Neuroscience*, vol. 2023, pp. 1–17, Mar. 2023, doi: 10.1155/2023/7371907.

[28] T. A. Prasetyo *et al.*, “Sales forecasting of marketing using adaptive response rate single exponential smoothing algorithm,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 31, no. 1, pp. 423–432, Jul. 2023, doi: 10.11591/ijeecs.v31.i1.pp423-432.

[29] G. W. Wicaksono and Andreawan, “ResNet101 model performance enhancement in classifying rice diseases with leaf images,” *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 7, no. 2, pp. 345–352, Mar. 2023, doi: 10.29207/resti.v7i2.4575.

[30] T. A. Prasetyo, V. L. Desrony, H. F. Panjaitan, R. Sianipar, and Y. Pratama, “Corn plant disease classification based on leaf using residual networks-9 architecture,” *International Journal of Electrical and Computer Engineering*, vol. 13, no. 3, pp. 2908–2920, Jun. 2023, doi: 10.11591/ijece.v13i3.pp2908-2920.

BIOGRAPHIES OF AUTHORS

Tegar Arifin Prasetyo is current lecturer and researcher member in Information Technology Department at Institut Teknologi Del since 2020. Have experience specializing in building mathematical models, machine learning, analytical android tools development, control system, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, machine learning, computational algorithms, optimal control, mathematical model in epidemiology and bioinformatics. He can be contacted at email: tegar.prasetyo@del.ac.id or arifintegar12@gmail.com.

Tiurma Lumban Gaol is current senior lecturer and researcher member in Information Technology Department at Institut Teknologi Del. Have experience specializing in building software engineering and re-design UI UX. Her research interests include artificial intelligence in plant disease. She can be contacted at email: tiur@del.ac.id.

Nico Felix Sipahutar is currently the final year of studies at Information Technology at Del Institute of Technology. He is a highly motivated individual with a strong interest in web development, data analytics, and project management. His research interests include artificial intelligence and machine learning models. He can be contacted at email: if320037@students.del.ac.id.

Tessalonika Siahaan is currently learning about information technology at the Del Institute of Technology. She enthusiastic in exploring AI technology using machine learning and deep learning algorithms and extending knowledge in web development. His research interests include artificial intelligence and deep learning model. She can be contacted at email: if320012@students.del.ac.id.

Trito Exaudi Manik is currently learning about information technology at the Del Institute of Technology. She is enthusiastic in exploring web development. His research interests include web architecture. He can be contacted at email: if320001@students.del.ac.id.

Rudy Chandra current lecturer and researcher member in Information Technology Department at Institut Teknologi Del since 2022. Have experience at specializing in building machine learning model, artificial neural network, decision support system, distributed system, software testing, and computer programming. He dedicates himself to university teaching and conducting research. His research interests include artificial intelligence, neural network, machine learning and algorithm computational, decision support system, distributed system, and software testing. He can be contacted at email: rudychandra@del.ac.id or rudychandra0@gmail.com.