An optimized transfer learning-based approach for *Crocidolomia pavonana* larvae classification

Risnawati[1], Rodiah[2], Sarifuddin Madenda[3], Diana Tri Susetianingtias[4]
[1]Department of Management, Faculty of Economy, Gunadarma University, Depok, Indonesia
[2]Department of Informatics, Faculty of Industry Technology, Gunadarma University, Depok, Indonesia
[3]Doctoral Program, Department of Information Technology, Gunadarma University, Depok, Indonesia
[4]Department of Computer System, Faculty of Information System and Information Technology, Gunadarma University, Depok, Indonesia

Article Info

Article history:

Received Aug 3, 2024
Revised Jan 29, 2025
Accepted Mar 15, 2025

Keywords:

Classification
Crocidolomia pavonana
DenseNet169
Mustard greens
ResNet50V2

ABSTRACT

The increasing demand for mustard greens has driven farmers to continuously improve mustard greens cultivation. One of the challenges in mustard greens cultivation is the presence of insect pests. A significant pest in mustard greens is *Crocidolomia pavonana (C. pavonana)*. *C. pavonana* damages plants by feeding on various parts, especially the leaves. The initial step in controlling them is insect pest monitoring. Monitoring aims to establish the control threshold. *C. pavonana* larvae have four instar stages: instar 1, 2, 3, and 4. Identification of the instar larval stages utilizes deep convolutional neural network (CNN) to classify *C. Pavonana* larvae on mustard greens using ResNet50V2 and DenseNet169 architectures optimized to enhance classification accuracy. The classification evaluation results show that both DenseNet169 and ResNet50V2 models achieve high accuracy, with DenseNet169 reaching the highest accuracy at 97.1%, while ResNet50V2 achieves an accuracy of 94.2%. The lower loss values on the test data compared to the validation data indicate that the deep learning models have successfully captured the patterns in *C. pavonana* images for classification. This classification process is expected to be one of the activities in monitoring the instar larvae to improve the accuracy of insecticide spraying and enhance mustard greens production.

Corresponding Author:

Rodiah
Department of Informatics, Faculty of Industry Technology, Gunadarma University
Margonda Raya 100, Pondok Cina, Depok, West Java, Indonesia
Email: rodiah@staff.gunadarma.ac.id

1. INTRODUCTION

The vegetables of the Brassicaceae family have important potential for body health [1]. Because of the content of vitamins, phenolic compounds, glucosinolates, carotenoids, and ascorbic acid. Some of the uses of these compounds are [2] it can provide prevention against cancer, antioxidants, anti-inflammatory, anti-diabetic, and protection for nerves [3]. Several types of vegetables in the Brassicaceae family include [4] White cabbage, green cabbage, Chinese broccoli, pak choi, and mustard greens. One of the important pests in the Brassicaceae vegetable family is *Crocidolomia pavonana (C. pavonana*) [5]. The presence of pest insects can cause damage to cabbage vegetable crops with attack intensities ranging from mild to severe, and can even lead to significant yield losses due to pests such as *C. pavonana* [6]. The monitoring of pests in cultivation areas is an activity that aligns with the concept of integrated pest management [7]. Integrated pest management is a cultivation concept involving the monitoring of crops to observe the presence of insects, types and populations of pest insects, symptoms of damage, and the percentage of plant damage caused by pest insect attacks in cultivated plants [8]. The activity of monitoring cultivated plants is conducted with the aim of establishing a threshold of action [9]. The action threshold for control using synthetic chemical insecticides against *C. pavonana* larvae is ≥3 egg clusters per 10 cabbage plants [9]. The monitoring is also conducted to avoid scheduled use of insecticides that result in residues on crop yields [10] and insects resistance [11]. One of the important pests in mustard greens is *C. pavonana*, which damages plants by feeding on their parts, especially the leaves [12]. The incident resulted in minor to severe damages, causing even farmers to experience failed harvests [13]. This necessitates the need for control measures, with the initial step being monitoring the target pest insects. Monitoring aims to establish the control threshold. One of the monitoring activities involves identifying *C. pavonana* larvae. *C. pavonana* larvae consist of instar stages 1, 2, 3, and 4 [14] which has variations in size, color, and shape. Instar 1 has the smallest size compared to later instars. Conventional identification has limitations, takes a long time, requires experts, and incurs high costs [15]. The accurate and rapid identification can be used to help farmers recognize the instar stages of *C. pavonana* pests.

One smart technique for identification is using deep convolutional neural networks (CNN). Architectures in deep CNN capable of identifying images, especially small larval forms, include residual network (ResNet) and densely connected convolutional network (DenseNet). ResNet [16] is an architecture designed to enable more efficient flow of information through deep layers in a network. ResNet can address issues related to deep network training and significantly improve image processing performance. DenseNet architecture [15] can create a dense network structure to achieve parameter efficiency, especially in extracting strong features from very small larvae images.

Pattnaik and Parvathi [17] classifying tomato plant pests using DenseNet-169 model resulted in an accuracy of 88.83%, with the highest accuracy of 88.37% achieved using the ResNet50V2 model. The model was trained for 100 epochs with a data split ratio of 7:1:2 for training, validation, and test data, respectively. Li *et al.* [18] performing the classification of rice plant pests using a pre-trained model to detect insect pests by splitting the data into a ratio of 6:2:2. This research utilized attentive recurrent generative adversarial network (ARGAN) data augmentation to address the imbalance in the dataset, resulting in a classification accuracy of 87.81%. Li *et al.* [19] performing plant pest classification in images with natural backgrounds using pre-trained models ResNet50 and ResNet152, employing data augmentation techniques such as mirror image, 90-degree rotation, adding noise to images, and cropped images. Fathimathul *et al.* [20] classifying 75 different species of butterflies with a split ratio of 7:2:1. The experimental results show a classification accuracy of 43% using the ResNet50 model. Wu *et al.* [21] performing classification into 30 classes using a pre-trained model and grad-CAM to highlight insect pest areas with a highest accuracy of 96.1% on the ResNet101 model.

On several previous research [2], [17]–[20], the classification accuracy appears to be below excellent classification [22]. Research by Li *et al.* [19] did not evaluate the training process by analyzing the accuracy and loss of the machine learning model. As a result, the accuracy with the Res-Net architecture only reached 43%. Accuracy and loss are important metrics analyzed at the end of each training epoch to assess the model's progress during the training process. This evaluation helps in detecting overfitting in the classification model and determining whether the machine learning model has successfully learned patterns from the data.

In this research, optimization will be performed on the architectures of DenseNet169 and ResNet50V2. The architecture of the machine learning model formed consists of a combination of pre-trained models, flatten layers, dropout layers, dense layers, and output layers. Each model is trained for 30 epochs using training and validation data. After training is completed, an analysis is conducted on the loss and accuracy graphs obtained by the machine learning model during the training process. The trained machine learning models are then evaluated once again using test data to determine the performance of the model on new data [23]. The second evaluation is important to get an overview of the model's performance before the model [24] is used for classifying images of *C. pavonana* larvae in instars 1, 2, 3, and 4, this research aims to produce the best machine learning model for classifying *C. pavonana* with high accuracy. The classification results are expected to facilitate the monitoring process of larval instars, enhancing the accuracy of insecticide spraying [25]. In addition, it is economically efficient to reduce control costs and minimize environmental pollution caused by active insecticide substances.

2. METHOD

This research was conducted based on five stages of processes, namely data pre-processing stage, data splitting stage, modeling stage, model evaluation stage, and the stage of selecting the best model accuracy. This study utilized CNN to generate a model for identifying *C. pavonana* larvae. The first stage involved data pre-processing, consisting of steps to process and prepare the data for use in deep learning model training. This stage included collecting path location data and organizing it into a table or dataframe, resizing images, augmenting data using techniques such as horizontal flip, zooming, rotation, shearing, and data normalization. Data splitting was performed using an 8:1:1 ratio. The subsequent stage involved the implementation, training, and evaluation of deep learning models. Two different deep learning models were proposed and trained in this study. The difference between the models lay in the pre-trained model and additional layers used. The pre-trained ResNet50v2 model consisted of 1 flatten layer, 1 dropout layer, 1 dense layer, and 1 output layer. The pre-trained DenseNet169 model consisted of 1 flatten layer, 3 dropout layers, 3 dense layers, and 1 output layer. Each model was trained for 30 epochs using both training and validation data. Analysis of the loss and accuracy graphs obtained from the deep learning models was performed after the training process was completed. Furthermore, the evaluation of deep learning models was conducted once more using testing data to assess their performance on new data. This second evaluation was carried out to gain an understanding of the model's performance.

2.1. *C. pavonana* larvae dataset

The dataset of Larva *C. pavonana* was obtained through direct acquisition in a mustard plantation area in Depok, West Java. The data comprises 684 images of *C. pavonana*, including 131 images of instar 1 class, 132 images of instar 2 class, 154 images of instar 3 class, and 267 images of instar 4 class. Examples of the four types of datasets can be seen in Figure 1.

C. pavonana undergoes complete metamorphosis, including the stages of egg, larva, pupa, and imago [26]. Each phase has a different duration of the stage. The larval stage includes instars 1, 2, 3, and 4 [27]. Where the larval stage is a critical phase because during this stage, there is activity of eating vegetables that causes damage. The larval stage starts from when the larva emerges until instar IV.

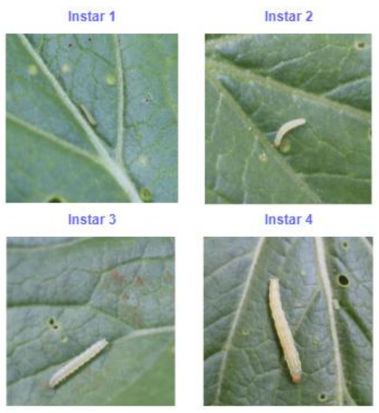

Figure 1. *C. pavonana* larvae image

2.2. *C. pavonana* larvae preprocessing

Preprocessing of *C. pavonana* larva images consists of several stages of processes, including:

– Adjusting the size of the image *C. pavonana*. The pre-trained model has a default RGB color image [28]. The default size of the *C. pavonana* image varies. Adjustment to the *C. pavonana* image is made to meet the input specifications of each pre-trained model [29]. The size to be used for the image of *C. pavonana* is 256×256 pixels. Table 1 shows the details of the input size for each pre-trained model. Table 1 provides an example of the size adjustment results for the image of *C. pavonana*.

Table 1. Example of image adjusment results for *C. pavonana*

Pre-trained model	Input size
ResNet50V2	(256, 256, 3)
DenseNet169	(256, 256, 3)

– Performing augmentation processes to increase the amount of training data [30] by modifying the data of *C. pavonana*, various techniques were employed, including horizontal flip, zooming, rotation, and shearing. Horizontal flip duplicates the image by flipping it horizontally. Zooming involves proportionally enlarging or reducing the image. Rotation is an augmentation technique that involves rotating the image at different angles. Shearing is performed by shifting the points on the image along a specific axis to create a distortion effect similar to perspective distortion. These four augmentation techniques were implemented using the parameters of the ImageDataGenerator, specifically for the training data only. The results of the augmentation techniques, namely horizontal flip, zooming, rotation, and shearing, can be observed in Figure 2.

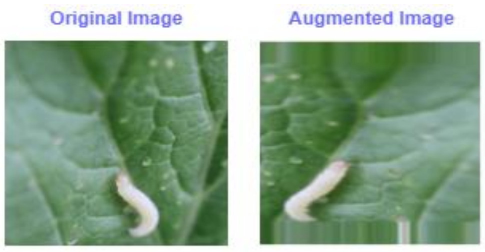

Figure 2. Example of augmented images of *C. pavonana* larvae

– Normalizing the values of each pixel in the image. Normalization is an important pre-processing technique to ensure that the model training process can run faster and more effectively [31], the task performed by dividing each pixel value by 255, resulting in new values within the range of 0-1, can be expressed in pseudocode as follows:

```
# Image Generator
ts_gen = ImageDataGenerator(preprocessing_function= scalar, rescale=1./255
)
```

The ImageDataGenerator is a declaration of an image generator that will call the *C. pavonana* images for validation and testing. The parameter rescale=1./255 indicates the value used for data normalization process.

2.3. *C. pavonana* larvae data splitting

The data splitting is done by dividing the dataset into three parts: training data (train_df), validation data (valid_df), and test data (test_df). The division ratio used for each data subset is 8:1:1. The data division starts by splitting the data into train_df and dummy_df with an 8:2 ratio. Then, dummy_df is further divided into a 5:5 ratio, resulting in valid_df and test_df. The data division is done using the train_test_split() function from the ScikitLearn library [32]. The division of this data is done in proportion to the respective class. Stratified sampling method is used to obtain a balanced distribution of subdata. The dataset used in this research consists of 39% instar 4 class, 23% instar 3 class, 19% instar 2 class, and 19% instar 1 class. The stratified sampling method can produce subdata with the same percentage distribution of classes [33] Through the complete dataset. This method is implemented by mapping the class column to the stratify parameter of the train_test_split() function.

2.4. Optimization *of C. pavonana* larvae classification model data splitting

Modeling is the core stage in this research, which starts with the initiation of the deep learning model. The process involves training the deep learning model using training data and evaluating the training process. All steps in this stage are repeated using different pre-trained models. Figure 3 represents the architecture of the pre-trained models and the fine-tuned models initialized in this study.

The architecture of this deep learning model consists of a combination of pre-trained models and fully connected layers. The selection of pre-trained models is done because they provide better results in a shorter training time [34]. This research utilizes two pre-trained models, namely ResNet50V2 and DenseNet169, obtained through the tf.keras.applications module. The deep learning model known as the fine-tuned model was implemented in this research. In this approach, the weights and biases of the pre-trained model were not frozen [35], so that they can be updated during training. The approach of using this fine-tuned model aims to allow the model [36] understanding the details in each image of *C. pavonana* more accurately. Default pre-trained models have their respective output layers. On average, pre-trained models available in the TensorFlow library have output layers with 1,000 classes [37]. In the output layer, it needs to be replaced with a new fully-connected layer to generate the classification of 4 classes that align with the objectives of this research. In the ResNet50V2 model, there is one fully-connected layer, one flatten layer, one dropout layer, one dense layer with 256 nodes, and one output layer with 4 nodes. In the DenseNet169 model, there is one fully-connected layer, one flatten layer, three dropout layers, three dense layers with 256, 128, 64 nodes respectively, and one output layer with 4 nodes. Optimization in the fine-tuned model is performed by utilizing the feature extraction network from the pre-trained model, combined with a new fully-connected network.

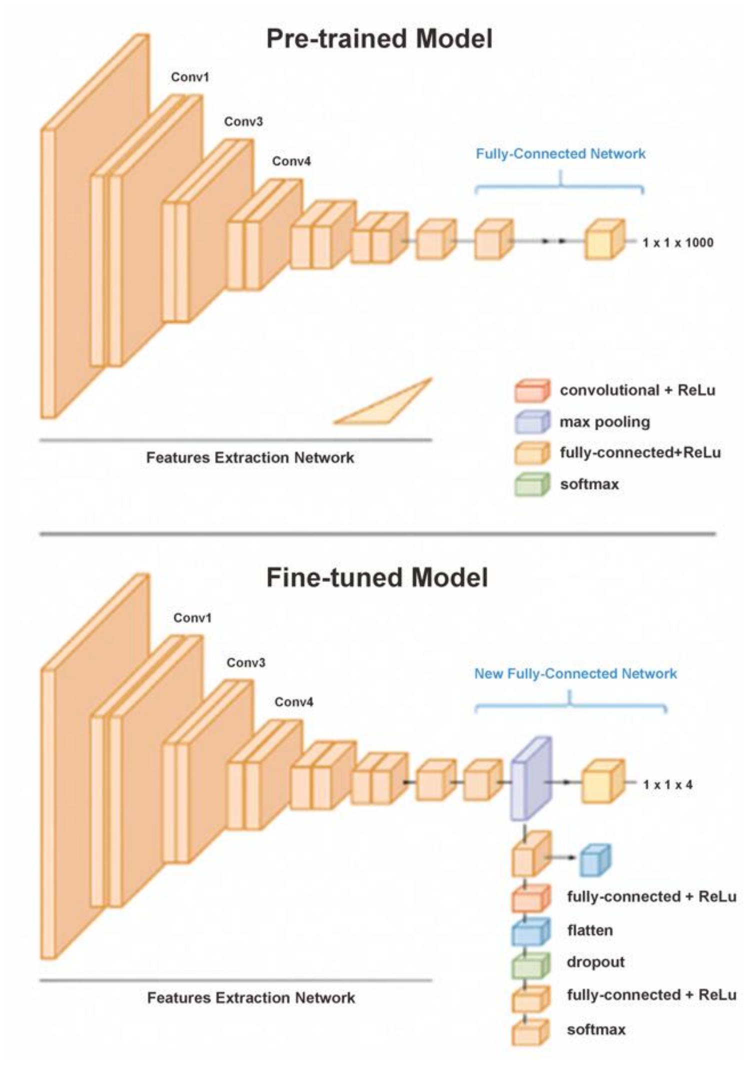

Figure 3. Architecture of pre-trained models and fine-tuned models

The feature extraction is used to read and recognize patterns from *C. pavonana* images, while the fully-connected network is responsible for learning and distinguishing patterns from each class.

- ResNetV50V2 model initiation: the first pre-trained model initialized in this research is the pre-trained Res-NetV50V2 model, which is available in the *tf.keras.applications.resnet_v2.ResNet50V2* module. The initialization of the pre-trained model is included without the default output layers so that new output layers can be added [38] used fully-connected layer.
- DenseNet169 initiation model: the second pre-trained model initialized in this research is the Dense-Net169 pre-trained model available in the *tf.keras.applications.resnet_v2.DenseNet169* module. The pre-trained model initialization is included without the default output layers, allowing the addition of new output layers using fully-connected layers.
- Fully-connected model Res-NetV50V2 layer initiation: the pre-trained model without the output layer used from the TensorFlow library needs to be supplemented with a fully-connected layer [39]. To generate a trainable deep learning model, the pre-trained ResNet50V2 model is provided with fully-connected layers and compiled using optimization functions and training metrics using the following pseudocode:

```
x = Flatten()(base_model.output)
x = Dropout(rate=0.2)(x)
x = Dense(256, activation='relu')(x)
outputs = Dense(class_count, activation='softmax')(x)

model = tf.keras.models.Model(base_model.input, outputs)

model.compile(Adamax(learning_rate=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
```

The parameter x=Flatten() (base_model.output) will add a flatten layer to the fully-connected layer and x=Dropout(rate=0.2)(x) will add a dropout layer to the fully-connected layer. As many as 20% of input values passing through this layer are nullified. Then x=Dense(256, activation='relu')(x) will add a dense layer to the fully-connected layer. The activation function used is relu. Parameter outputs=Dense(class_count, activation='softmax')(x) will add the dense layer as an output layer to the fully-connected layer. Class_count is the number of classes for the classification task of this research, namely 4 with softmax as the activation function. Using model=tf.keras.models.Model(base_model.input, outputs) as a deep learning model is created by combining the CNN layer from the pre-trained model with the newly created fully-connected layer and stored in the model variable. This research uses the Adamax optimization function with a learning_rate of 0.001 [40]. Categorical_crossentropy used as loss function and accuracy as training metric.

– Fully-connected model DenseNet169 layer initiation: the pre-trained model without an output layer used from the TensorFlow library needs to be added with a fully-connected layer to produce a deep learning model that can be trained [41]. The pre-trained DenseNet169 model is given fully-connected layers and compiled with optimization functions and training metrics.

2.5. Training classification model of *C. pavonana*

The model training stage implemented in deep learning models is initiated through a training process called fine-tuning [41]. The process of training the deep learning model involves training data and validation data, where the training data is used to learn the features of the *C. Pavonana* images, and the validation data is used to assess the model's performance during training. The training process is carried out for 6×5 epochs, and the results of each epoch are monitored using callback functions. This research uses a batch size of 32 with 100 epochs. A threshold of 0.9 is used in this study to determine the variables monitored as loss. If the training accuracy is less than the threshold, the loss will be equal to the training accuracy. If it is greater than the threshold, the loss will be equal to the validation loss. The training model evaluation process can be seen in Figure 4.

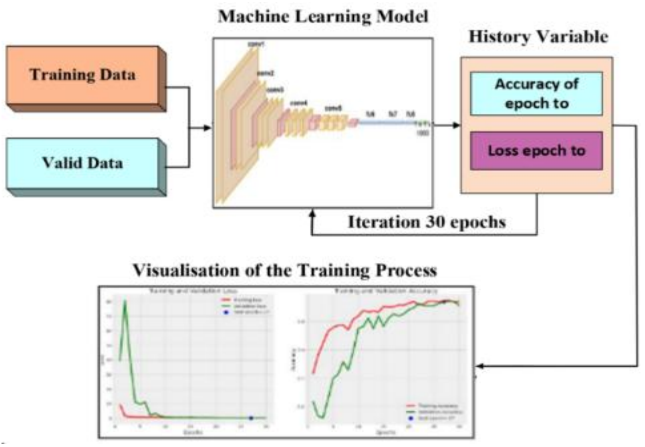

Figure 4. Classification model training evaluation process

3. RESULTS AND DISCUSSION

The research results consist of the results of preprocessing of the *C. pavonana* image dataset, data splitting, modeling, evaluation, results of the best model accuracy. Tests were carried out using several *C. pavonana* images from test data to obtain classification results for each model, as well as the best accuracy.

3.1. *C. pavonana* image processing result

In this research, the dataset preprocessing stage was carried out to add variation to the training data and prepare the data so that the training process could run effectively. This stage begins with entering the data path location into the dataframe, adjusting the image size, performing horizontal flip, zooming, rotation, shearing, and normalization. The steps in the data pre-processing stage in this research were carried out using the ImageDataGenerator() function so that the results were only visible when executing the image calling code. The results of *C. pavonana* 's image preprocessing can be seen in Figure 5.

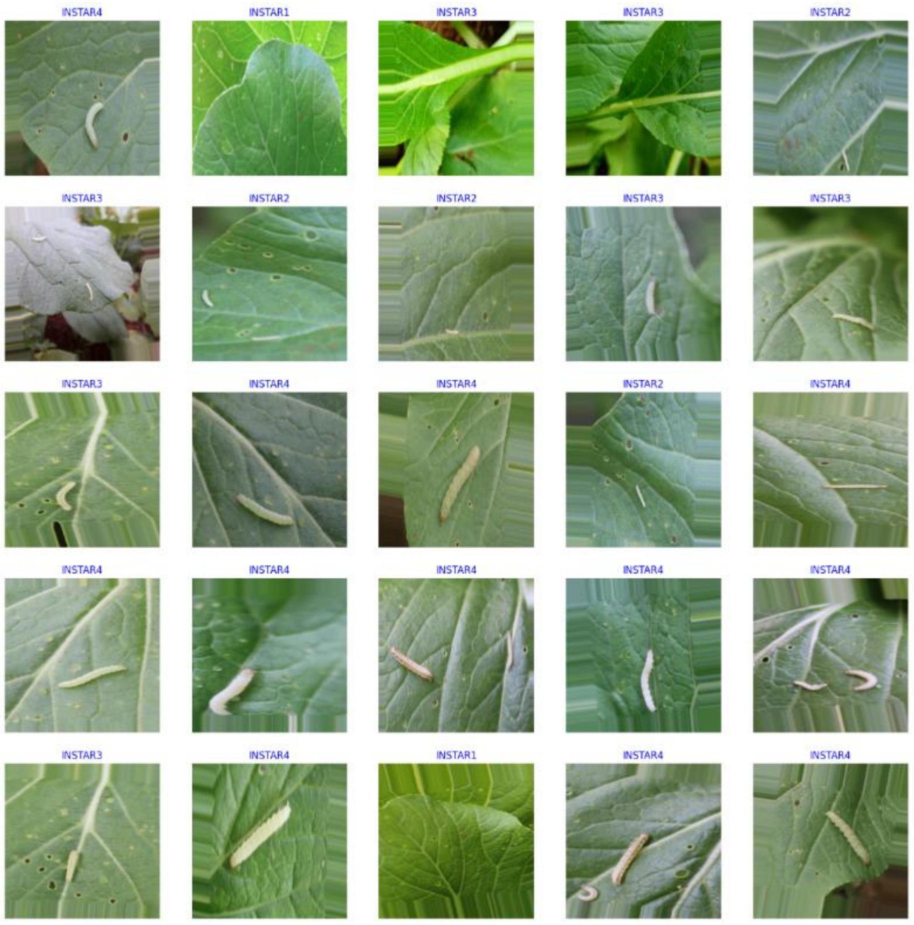

Figure 5. Preprocessing *C. pavonana* result

3.2. Classification model initialization results

Model initiation in this research was carried out by combining a pre-trained model with a fully-connected layer to obtain a fine-tuned model. Two different pre-trained models are used via the tf.keras.applications library to initiate two fine-tuned models. The TensorFlow API with functional types is implemented through this stage so that the architectural layers of the model can be seen. The architecture of the fine-tuned model initiated from the pre-trained ResNet50V2 model. This model has a total of 24,090,372 parameters and 24,044,932 parameters that can be trained. The architecture of the fine-tuned model initialized from the pre-trained DenseNet169 model. This model has a total number of parameters of 13,110,532 and 12,952,132 parameters that can be trained.

3.3. Results of classification model accuracy evaluation

The evaluation of the training model is done using accuracy and loss records from the deep learning model process. Training evaluation aims to assess the model's performance on both training and validation data. This stage is crucial to determine whether the model has understood patterns in the data or is merely making random guesses. In this phase, line graphs are used to visualize accuracy and loss values at each epoch. These line graphs include training data accuracy, validation data accuracy, training data loss, validation data loss, the best epoch based on accuracy and loss, as well as the best values for accuracy and loss. Each graph consists of two subplots displaying loss and accuracy. A good model is one with low loss and high accuracy. The *x* and *y* axis values are set to the same scale to provide a proportional representation of the training process for each model.

Figure 6 shows the progress of the training process for a machine learning model based on the pre-trained ResNet50V2 model over 30 epochs. The training loss data demonstrates a steady decrease from 80 towards zero, and the training data accuracy shows a stable increase towards 100%. Validation loss data shows a decreasing trend. Meanwhile, validation data accuracy has an upward trend with two low-value dips. The best epoch based on loss occurs at epoch 27 with a loss value of 0.140. Based on accuracy, the best epoch also occurs at epoch 27 with a validation accuracy of 94.11%.

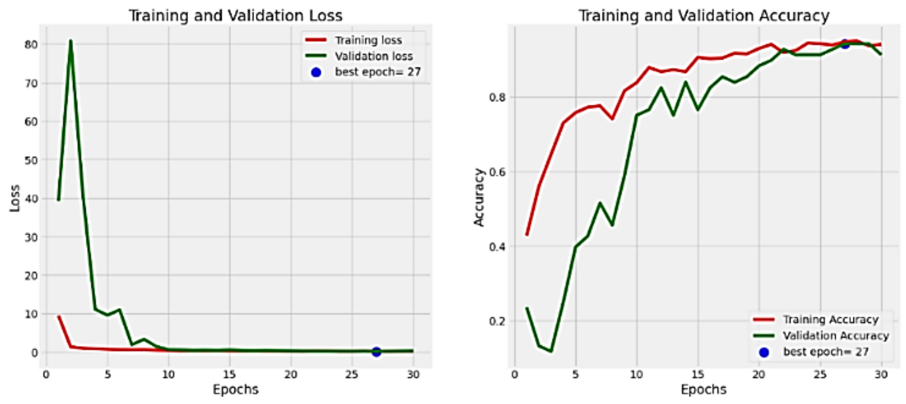

Figure 6. ResNet50V2 accuracy model result

Figure 7 shows the progress of the machine learning model training process based on the pre-trained DenseNet169 model for 30 epochs. The training loss data shows a stable decreasing value from 1.8 towards zero and the training data accuracy also shows a stable increasing trend towards 100%. Validation loss data shows a downward trend. Meanwhile, the accuracy of the validation data has an upward trend with two valleys with low values. The best epoch based on loss occurred at the 29th epoch with a loss value of 0.093. Based on accuracy, the best epoch occurred in the 26th epoch with a validation accuracy value of 97.05%.

The evaluation results of the deep learning model training process show that all models have an average accuracy on training data and validation data above 80%. The trend of increasing accuracy values towards 100% in validation data indicates that the model has been able to understand patterns in the data in making classifications. The accuracy and loss obtained at this stage indicate that the model can be used for further evaluation using test data. Table 2 shows the results of model evaluation using the evaluate() function. Table 2 shows the accuracy and loss of each model on training data, validation data and testing data. The DenseNet169 model achieved the highest accuracy on training data of 96.8% and the smallest loss of 0.090, validation data of 96.8% and the smallest loss of 0.133, testing data of 97.1% with the smallest loss of 0.112.

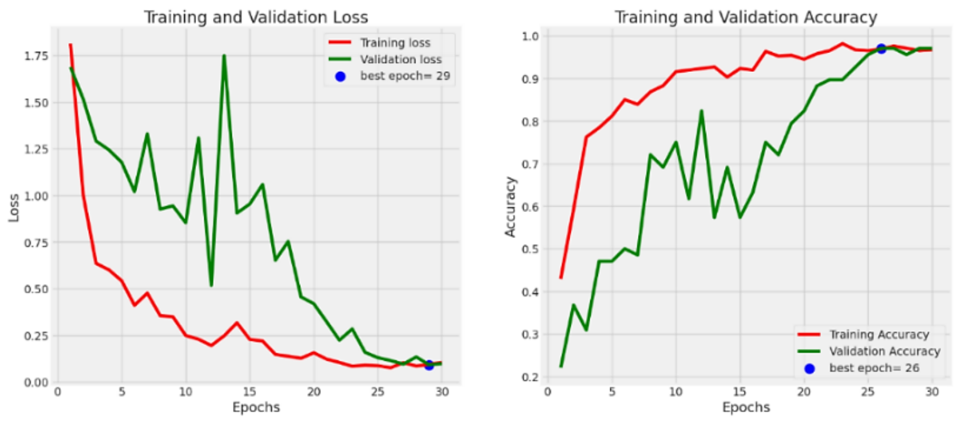

Figure 7. DenseNet169 accuracy model result

Table 2. Results of model evaluation with accuracy and loss

Model	Train data		Val data		Test data	
	Acc	Loss	Acc	Loss	Acc	Loss
ResNet50V2	0.937	0.121	0.906	0.160	0.942	0.156
DenseNet169	0.968	0.090	0.968	0.133	0.971	0.112

Figure 8 classification results from one of *C. pavonana*'s images, test data completed with instar 2 with the ResNet50V2 and DenseNet169 models. Based on Table 3, it was obtained from various tests on the DenseNet169 model, in the first test using a batch size of 32 and using 1 additional layer, namely, 1 dropout layer and 1 dense layer and running for 30 epochs, the accuracy on the test data was 91% and loss of 0.25. In the second test, using the same number of batch sizes with 3 additional layers, 3 dropout layers, 3 dense layers and running for 30 epochs, the accuracy on the test data was 97.1% and the loss was 0.11. The third and fourth trials carried out using a batch size of 64 and using 1 or 3 additional layers experienced errors when running model training.

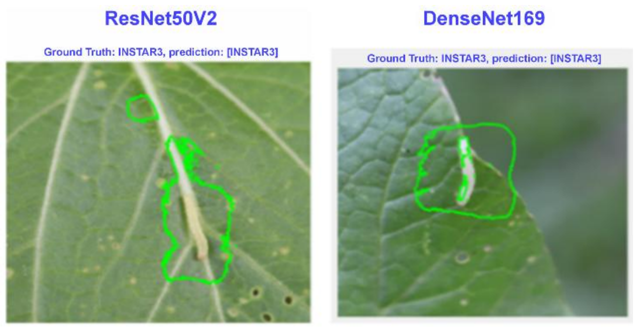
ResNet50V2
Ground Truth: INSTAR3, prediction: [INSTAR3]
DenseNet169
Ground Truth: INSTAR3, prediction: [INSTAR3]

Figure 8. Classification results with ResNet50V2 and DenseNet169 models

Table 3. Comparison results of accuracy and loss of DenseNet169 model

epoch	Batch size	Add layer	Train data		Val data		Test data	
			Acc	Loss	Acc	Loss	Acc	Loss
30	32	1	0.93	0.12	0.90	0.16	0.94	0.15
30	32	3	0.96	0.12	0.87	0.54	0.86	0.36
45	64	1	0.98	0.11	0.89	0.26	0.92	0.13
35	64	3	0.93	0.12	0.89	0.34	0.92	0.12

4. CONCLUSION

Based on the implementation results of *C. pavonana* larva identification using the CNN model as well as the pre-trained architectures ResNet50V2 and DenseNet169 on *C. pavonana* images in this study, several conclusions can be drawn. The accuracy of the second trial of the DenseNet169 model yielded the most optimal accuracy and loss, with accuracy exceeding 90%, and the loss figures for each training data, validation data, and test data were relatively close. This indicates that there was no overfitting in the first trial. In the first trial, the accuracy was above 90%, but the loss figures for each training data, validation data, and test data showed significant differences, indicating that the model experienced overfitting. Meanwhile, the third and fourth trials could not be executed due to errors. Meanwhile, the ResNet50V2 model achieved an accuracy of 94.2% and a loss of 0.156. All models demonstrated good performance on the test data with accuracy above 90%. In future research, it is expected that the development of the deep learning model for classifying *C. pavonana* larvae will be carried out using other pre-trained methods to achieve even more optimal accuracy.

ACKNOWLEDGEMENTS

The authors would like to express our appreciation to the Research Institutions of Gunadarma University for their continued facilitation and support throughout the research activities.

FUNDING INFORMATION

This research was funded by the Fundamental Research Grant from the Ministry of Research, Technology, and Higher Education (Kemenristekdikti) under contract number LLDIKTI 3 Number 795/LL3/AL.04/2024, dated June 26, 2024.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Risnawati	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Rodiah	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Sarifuddin Madenda	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Diana Tri Susetianingtias	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

This study did not involve human participants; therefore, informed consent was not required.

ETHICAL APPROVAL

This study utilized pest dataset and did not involve human or vertebrate animal subjects; therefore, ethical approval was not required.

DATA AVAILABILITY

The data collected for this study are part of ongoing research with potential commercial applications. Therefore, we are unable to make the data publicly available at this time due to intellectual property restrictions and the ongoing commercialization process. However, researchers interested in accessing the dataset are welcome to contact the corresponding author at rodiah@staff.gunadarma.ac.id for further information and to discuss potential data sharing under appropriate conditions.

REFERENCES

[1] F. P. Alloggia, R. F. Bafumo, D. A. Ramirez, M. A. Maza, and A. B. Camargo, “Brassicaceae microgreens: A novel and promissory source of sustainable bioactive compounds,” *Current Research in Food Science*, vol. 6, 2023, doi: 10.1016/j.crfs.2023.100480.

[2] M. Hasanuzzaman, S. Araújo, and S. S. Gill, *The plant family fabaceae: Biology and physiological responses to environmental stresses,* Springer Singapore, 2020, doi: 10.1007/978-981-15-4752-2.

[3] D. Ramirez, A. Abellán-Victorio, V. Beretta, A. Camargo, and D. A. Moreno, “Functional ingredients from brassicaceae species: Overview and perspectives,” *International Journal of Molecular Sciences*, vol. 21, no. 6, 2020, doi: 10.3390/ijms21061998.

[4] H. S. Hillen, G. Kokic, L. Farnung, C. Dienemann, D. Tegunov, and P. Cramer, “Structure of replicating SARS-CoV-2 polymerase,” *Nature*, vol. 584, no. 7819, pp. 154–156, 2020, doi: 10.1038/s41586-020-2368-8.

[5] S. Srinivasan *et al.*, “Structural genomics of SARS-CoV-2 indicates,” *Viruses*, vol. 12, no. 360, pp. 1–17, 2020.

[6] N. Mpumi, R. S. Machunda, K. M. Mtei, and P. A. Ndakidemi, “Selected insect pests of economic importance to Brassica oleracea, their control strategies and the potential threat to environmental pollution in Africa,” *Sustainability*, vol. 12, no. 9, 2020, doi: 10.3390/su12093824.

[7] P. B. Angon *et al.*, “Integrated pest management (IPM) in agriculture and its role in maintaining ecological balance and biodiversity,” *Advances in Agriculture*, vol. 2023, 2023, doi: 10.1155/2023/5546373.

[8] R. Collier, “Pest insect management in vegetable crops grown outdoors in northern Europe–approaches at the bottom of the IPM pyramid,” *Frontiers in Horticulture*, vol. 2, pp. 1–11, Apr. 2023, doi: 10.3389/fhort.2023.1159375.

[9] L. Prabaningrum and T. K. Moekasan, "Incidence and diversity of insect pests and their natural enemies in control threshold-based cabbage cultivation," *AAB Bioflux*, vol. 12, no. 1, pp. 12–21, 2020.

[10] F. Apple, G. Kashi, N. Nourieh, P. Mostashari, and F. Khushab, "Food chemistry: X optimization of extraction conditions and determination of the chlorpyrifos, diazinon, and malathion residues in environment samples," *Food Chemistry: X*, vol. 12, 2021, doi: 10.1016/j.fochx.2021.100163.

[11] D. Dono, Y. D. Pratiwi, S. Ishmayana, and D. Prijono, "Resistance level of crocidolomia pavonana against profenofos synthetic insecticide and its susceptibility to azadirachta indica seed extract," *Cropsaver: Journal of Plant Protection,* vol. 1, no. 2, pp. 74–84, 2018, doi: 10.24198/cs.v1i2.19912.

[12] E. Syahputra and Minarti, "Joint action of Azadirachta indica and Barringtonia asiatica seed extracts against Crocidolomia pavonana," *Agrivita*, vol. 44, no. 1, pp. 40–47, 2022, doi: 10.17503/agrivita.v44i1.2809.

[13] S. Ramasamy, M.-Y. Lin, F.-C. Su, and C.-C. Huang, "Towards developing an integrated pest management strategy for cabbageproduction systems in lowlands of Taiwan, Republic of China," *Mysore Journal of Agriculture Science*, vol. 51, no. A, pp. 185–197, 2017.

[14] E. Shonga and E. Getu, "Population dynamics of cabbage aphid, Brevicoryne brassicae L. (Homoptera: Aphididae) in relation to weather factors on major brassica crops in central rift valley of Ethiopia: Baseline studies for the management of the pest," *International Journal of Tropical Insect Science*, vol. 41, no. 1, pp. 455–462, Mar. 2021, doi: 10.1007/s42690-020-00226-4.

[15] W. Li, T. Zheng, Z. Yang, M. Li, C. Sun, and X. Yang, *Classification and detection of insects from field images using deep learning for smart pest management: A systematic review*, vol. 66. 2021.

[16] A. Surya, D. B. Peral, A. VanLoon, and A. Rajesh, "A mosquito is worth 16x16 larvae: Evaluation of deep learning architectures for mosquito larvae classification," *arXiv-Computer Science*, pp. 1–6, 2022.

[17] G. Pattnaik and K. Parvathi, "Machine learning-based approaches for tomato pest classification," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 20, no. 2, pp. 321–328, 2022, doi: 10.12928/TELKOMNIKA.v20i2.19740.

[18] Z. Li, X. Jiang, X. Jia, X. Duan, Y. Wang, and J. Mu, "Classification method of significant rice pests based on deep learning," *Agronomy*, vol. 12, no. 9, 2022, doi: 10.3390/agronomy12092096.

[19] Y. Li, H. Wang, L. M. Dang, A. S. -Niaraki, and H. Moon, "Crop pest recognition in natural scenes using convolutional neural networks," *Computers and Electronics in Agriculture*, vol. 169, 2020, doi: 10.1016/j.compag.2019.105174.

[20] R. P. P. Fathimathul *et al.*, "A novel method for the classification of butterfly species using pre-trained CNN models," *Electronics*, vol. 11, no. 13, pp. 1–20, 2022, doi: 10.3390/electronics11132016.

[21] F. Wu *et al.*, "A new coronavirus associated with human respiratory disease in China," *Nature*, vol. 579, no. 7798, pp. 265–269, 2020, doi: 10.1038/s41586-020-2008-3.

[22] T. Yang and Y. Ying, "AUC maximization in the era of big data and AI : A survey," *ACM Computing Surveys*, 2022.

[23] D. Motta *et al.*, "Optimization of convolutional neural network hyperparameters for automatic classification of adult mosquitoes," *PLoS ONE*, vol. 15, no. 7, pp. 1–30, 2020, doi: 10.1371/journal.pone.0234959.

[24] M. A. M. Fuad *et al.*, "Detection of aedes aegypti larvae using single shot multibox detector with transfer learning," *Bulletin of Electrical Engineering and Informatics*, vol. 8, no. 2, pp. 514–518, 2019, doi: 10.11591/eei.v8i2.1263.

[25] S. Boyer *et al.*, "Monitoring insecticide resistance of adult and larval *Aedes aegypti* (Diptera: Culicidae) in Phnom Penh, Cambodia," *Parasites and Vectors*, vol. 15, no. 1, pp. 1–7, 2022, doi: 10.1186/s13071-022-05156-3.

[26] A. Afandhi, I. Fernando, T. Widjayanti, A. K. Maulidi, H. I. Radifan, and Y. Setiawan, "Impact of the fall armyworm, Spodoptera frugiperda (J. E. Smith) (Lepidoptera: Noctuidae), invasion on maize and the native Spodoptera litura (Fabricius) in East Java, Indonesia, and evaluation of the virulence of some indigenous entomopathogenic fungus i," *Egyptian Journal of Biological Pest Control*, vol. 32, no. 1, pp. 56–67, 2022, doi: 10.1186/s41938-022-00541-7.

[27] I. Y. Vajri, Trizelia, and H. Rahma, "Induction of resistance to larvae Crocidolomia pavonana F. (Lepidoptera: Crambidae) using Rhizobacteria to the cabbage," *Andalasian International Journal of Entomology*, vol. 2, no. 1, pp. 15–23, 2024, doi: 10.25077/aijent.2.1.15-23.2024.

[28] A. G. Magaña, J. Cerda, and S. Ramírez-Zavala, "Pretrained convolutional neural networks performance assessment under different color variability indexes," *Procedia Computer Science*, vol. 215, pp. 869–877, 2022, doi: 10.1016/j.procs.2022.12.089.

[29] H. Lee and J. Song, "Introduction to convolutional neural network using Keras; An understanding from a statistician," *Communications for Statistical Applications and Methods*, vol. 26, no. 6, pp. 591–610, 2019, doi: 10.29220/CSAM.2019.26.6.591.

[30] K. Kusrini *et al.*, "Data augmentation for automated pest classification in Mango farms," *Computers and Electronics in Agriculture*, vol. 179, 2020, doi: 10.1016/j.compag.2020.105842.

[31] T. N. Doan, "Large-scale insect pest image classification," *Journal of Advances in Information Technology*, vol. 14, no. 2, pp. 328–341, 2023, doi: 10.12720/jait.14.2.328-341.

[32] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] T. Parr and J. D. Wilson, "Technical report: Partial dependence through stratification," *arXiv-Computer Science*, pp. 1-13, 2019.

[34] S. N. A. M. Johari, S. Khairunniza-Bejo, A. R. M. Shariff, N. A. Husin, M. M. M. Masri, and N. Kamarudin, "Automatic classification of bagworm, metisa plana (walker) instar stages using a transfer learning-based framework," *Agriculture*, vol. 13, no. 2, 2023, doi: 10.3390/agriculture13020442.

[35] G. Yuan *et al.*, "Layer freezing and data sieving: Missing pieces of a generic framework for sparse training," *Advances in Neural Information Processing Systems*, vol. 35, no. NeurIPS, 2022.

[36] F. Directions, "Understanding of machine learning with deep learning," *Computers MDPI*, vol. 12, no. 91, pp. 1–26, 2023.

[37] L. Yu, B. Li, and B. Jiao, "Research and implementation of CNN based on TensorFlow," *IOP Conference Series: Materials Science and Engineering*, vol. 490, no. 4, pp. 1–6, 2019, doi: 10.1088/1757-899X/490/4/042022.

[38] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0197-0.

[39] C. Zhang *et al.*, "ResNet or DenseNet? Introducing dense shortcuts to ResNet," in *2021 IEEE Winter Conference on Applications of Computer Vision, WACV 2021*, 2021, pp. 3549–3558, doi: 10.1109/WACV48630.2021.00359.

[40] M. J. Uddin, Y. Li, M. A. Sattar, Z. M. Nasrin, and C. Lu, "Effects of learning rates and optimization algorithms on forecasting accuracy of hourly typhoon rainfall: Experiments with convolutional neural network," *Earth and Space Science*, vol. 9, no. 3, 2022, doi: 10.1029/2021EA002168.

[41] E. Bartz, T. Bartz-Beielstein, M. Zaefferer, and O. Mersmann, *Hyperparameter tuning for machine and deep learning with R A practical guide*. 2022.

BIOGRAPHIES OF AUTHORS

Risnawati earned her S.P. and M.Si. (Agronomy) from Tanjung Pura University and (Entomology) from IPB University, Indonesia in 2007 and 2013, respectively. She is currently a Lecturer at the Faculty of Industrial Technology, Agrotechnology Study Programme, Gunadarma University, Jakarta, Indonesia. She is also a researcher in artificial intelligence at Gunadarma University since January 2023. His research includes botanical insecticides, and machine learning. He has published more than 25 papers in national journals and international conferences. From November 2013 to August 2024, he was a research fellow at Gunadarma University, Indonesia. She can be contacted at email: risnawati@staff.gunadarma.ac.id.

Rodiah is currently Researcher, Lecturer and Vice Head of Postgraduate Academic System Development at Gunadarma University. From 2012 until Now, won 9 research grants from Indonesian Directorate General for Higher Education DIKTI (RISTEKDIKTI). Nowadays, authoring 2 books about medical image processing for retinal fundus image, 1 books about retinal biometric. In other hand, she has more than 60 publication within; journals, proceeding and book chapter. She also has more than 20 intellectual property rights (IPR)and 6 patent. She can be contacted at email: rodiah@staff.gunadarma.ac.id.

Sarifuddin Madenda holds a Doctor of Electronics and Image Processing, Universite de Bourgogne, France. Currently he is active as the Head of Ph.D. Programs of Information Technology and a lecturer at Ph.D. Program at Gunadarma University. His research interest is in image and video processing, multimedia data compression, content based image and video retrieval, steganography: encryption, decryption, coding and decoding of multimedia secret documents, real time system architecture (FPGA and ASIC design). He can be contacted at email: sarif@staff.gunadarma.ac.id.

Diana Tri Susetianingtias is currently Researcher and Lecturer at Gunadarma University. From 2017 until Now, won 4 research grants from Indonesian Directorate General for Higher Education DIKTI (RISTEKDIKTI). Nowadays, authoring 2 books about medical image processing for retinal fundus image, 1 books about retinal biometric. In other hand, she has more than 30 publication within; journals, proceeding and book chapter. She also has more than 10 intellectual property rights (IPR)and 3 patent. She can be contacted at email: diants@staff.gunadarma.ac.id.