

Tree Damage Type Classification Based on Forest Health Monitoring Using Mobile-Based Convolutional Neural Network

¹Faishal Hariz Makaarim Gandadipoera, ²Rico Andrian, ³Rahmat Safe'i

^{1,2}Computer Science Study Program, Faculty of Mathematics and Natural Sciences, University of Lampung, Indonesia

³Forestry Study Program, Faculty of Agriculture, University of Lampung, Indonesia

Email: ¹faishalhmg@yahoo.co.id, ²rico.andrian@fmipa.unila.ac.id, ³rahmat.safei@fp.unila.ac.id

Article Info

Article history:

Received Jan 25th, 2024

Revised Apr 10th, 2024

Accepted Apr 26th, 2024

Keyword:

Convolutional Neural Network

Forest Health Monitoring

MobileNetV2

Mobile Application

TensorFlow Lite

ABSTRACT

One of the fundamental parts of surveying forest health conditions with Forest Health Monitoring (FHM) is to visually assess the damage experienced by trees under certain conditions. This visual assessment can be facilitated using a Convolutional Neural Network (CNN) which involves building the MobileNetV2 model architecture. The model was trained using 1600 image data with 16 classes. The image data was pre-processed by resizing it to 224x224. The data was categorized into three categories: 80% was allocated for training, 10% for validation, and testing with 10% also. Training was done by changing the values from batches with a maximum of 100 epochs. The model was then incorporated into a mobile application using TensorFlow Lite and testing the application gave satisfactory results. The model results get the best accuracy rate of 98.75% and a loss of 0.0497. This research concludes that the classification of tree damage types based on FHM with CNN can be done. For accurate results, the image provided by the user must be clear and reflect the actual damage observed on the tree.

Copyright © 2024 Puzzle Research Data Technology

Corresponding Author:

Rahmat Safei,

Department of Forestry,

University of Lampung,

Jl. Prof. Dr. Ir. Sumantri Brojonegoro No.1, Gedong Meneng, District. Rajabasa, Bandar Lampung City, Lampung 35141,

Email: rahmat.safei@fp.unila.ac.id

DOI: <http://dx.doi.org/10.24014/ijaidm.v7i2.29421>

1. INTRODUCTION

Maintaining forest health requires comprehensive monitoring, including looking at four ecological indicators. These metrics, which include productivity, biodiversity, area quality, and Vitalize (tree damage and canopies conditions), The role they play in influencing the health of tropical rainforests is important. The four indicators are closely interconnected and together provide a comprehensive understanding of forest conditions [1]. The forest health of an individual tree will later spread into the organization of the forest population, so individual tree health must also be considered. Symptoms of tree damage can appear when there is a disturbance in plant growth, which causes changes in the structure, sizing, colors, and texture of the plant. Site damage occurs when the physiological processes of the tree are disrupted by sickness, bugs, and also other abiotic factors [2]. Applying the Forest Health Monitoring (FHM) approach to evaluate the forest condition, one main element that requires constant observation is the extent of tree damage under various scenarios. The FHM process starts with determining a forest sampling index to check its health status and then assessing the condition of the canopy based on ecological parameters [3].

The FHM method underlying the app's tree damage type classification feature allows users to survey and classify trees in monitoring the well-being of forest land so that users can contribute to the general administration and conservation of forests by identifying and publicizing potential problems that may affect tree health. We believe this feature will enable forest watchers and the public to play a role in forest management activities. The applications are classified using the MobileNetV2 Application features are

classified using the MobileNetV2 architecture, a Convolutional Neural Network (CNN) model. The MobileNetV2 architecture is designed to overcome barriers to running CNN models on limited-resource devices such as smartphones and tablets [4].

A few studies have demonstrated the successful use of MobileNetV2 in various fields, as well as semantic segmentation, object recognition, and image classification. A study on the MobileNetV2-based Fruit Image Classification Model obtained an accuracy value of 99% which concluded that transfer learning plays a big role in achieving better results, and dropout techniques help reduce overfitting in transfer learning [5]. classification of rice diseases and pests using MobileNetV2 got 96% accuracy with fewer parameters making the model lightweight if you want to apply it on mobile devices compared to NasNet Mobile, EfficientNet B7, Inception V3, VGG 16, and models with simple CNN [6]. in addition, a study to distinguish various types of mangoes using TensorFlow Lite obtained 95% accuracy for mango recognition [7]. Based on the success of studies using MobileNetV2 to classify images, this study emphasizes the advantages of MobileNetV2 in terms of size, accuracy, and speed of model inference that support our choice to use it in tree damage type identification applications by applying early stopping callbacks and a dropout layer to avoid overfitting.

The trust in CNN makes us confident that this feature will provide its users with the ability to make it easier to protect the forest. This feature is not only limited to identifying damaged trees, but we believe it will also encourage communities to take responsibility for managing forests and maintaining their health. It has been observed that community involvement plays an imperative role in the overall health of the forest, and the more actively the community is involved, the greater the health of the forest [8]. Research was carried out to identify the type of damage to trees based on mobile-based FHM to facilitate the observation of tree damage in the forest using CNN MobileNet architecture.

This research implements a Convolutional Neural Network with MobileNetV2 architecture converted into a TensorFlow Lite Model against tree damage in Mobile applications with classification to help make it even easier to observe the type of damage of the tree in the forest. Based on Forest Health Monitoring (FHM), 1600 picture files with 16 classifications of tree damage were employed in the current study.

2. RESEARCH METHOD

The research process carried out is the extraction of image data using a Convolutional Neural Network with MobileNetV2 architecture to recognize the type of damaged trees in a mobile application to help observe the health of trees in the forest. The research stages are graphically represented in Figure 1.

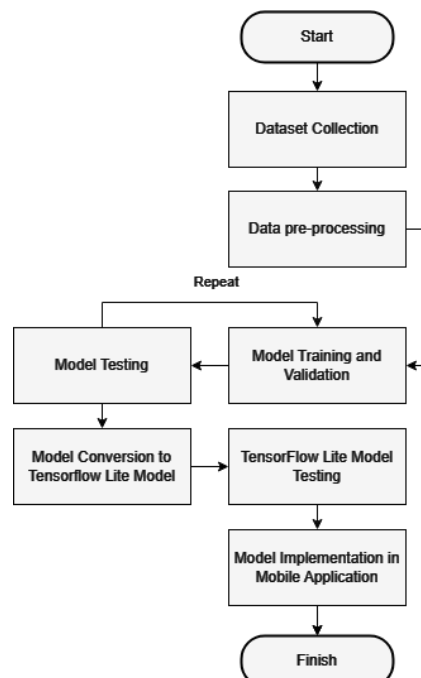


















Figure 1. The process of conducting research.

2.1. Dataset Collection

The dataset is useful for making the machine recognize the aspects present in the image. The dataset was collected by capturing images of trees that exhibited damage from the frontal perspective of the inflicted

harm. the dataset collected amounted to 1600 images of tree damage with 16 parameter classes where each class has 100 images, the class includes the code and type of tree damage as seen graphically in Figure 2 [9]. Collected datasets will be subjected to data preprocessing.

Code	Damage Type	Sample Picture	Code	Damage Type	Sample Picture
01	Cancer		13	Roots Break or die	
02	Konk		20	Liana	
03	Open Wound		21	Loss or Death of Dominant shoots	
04	Resinosis or Gumosis		22	Branches that are broken or dead	
05	Stem Break		23	Excessive Brum Branching	
06	Termite Nest		24	Broken Shoots Leaves	
11	Broken Stems or Roots		25	Discolored Leaves	
12	Brum on the Root or Stem		31	Others (Growong)	

Figuru 2. The process of conducting research.

2.2. Convolutional Neural Network

A variation of the Multilayer Perceptron algorithm built specifically to process data in two dimensions is the Convolutional Neural Network or CNN [10]. Since it has multiple layers and is typically used to analyze image data, it is categorized as an Artificial Neural Network. In contrast, the Multilayer Perceptron is not ideal for image classification as it does not store information about objects and considers pixels as separate entities, resulting in sub-par performance. CNN architecture, which consists of various stages, is capable of training data and is primarily used for datasets containing image objects. Unlike other neural network methods, CNNs excel at transforming images into understandable information. CNN architecture usually consists of two main components: Feature Learning, which consists of Convolutional and Fusion Layers, and Fully Connected Layers [11]. An illustration of the process of CNN can be seen in Figure 3.

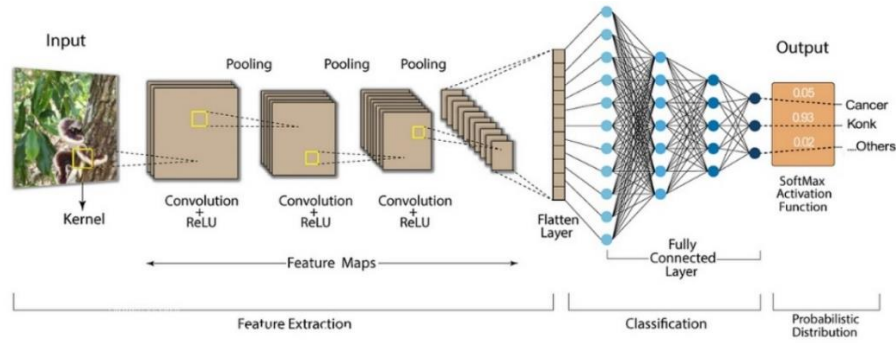


Figure 3. CNN Architecture for Image Classification Process

2.3. MobileNetV2

The MobileNet architecture is designed for mobile applications with low latency and small capacity size, the addition of bottleneck connection and shortcut features distinguishes MobileNetV2 from previous versions [12]. MobilenetV2 is a CNN architecture that has proven to be useful in detecting multiple objects and extracting them efficiently and quickly. MobileNetV2 offers advantages for mobile devices and devices with limited memory and computing capabilities, it can reduce memory usage on mobile devices and significantly improve image classification efficiency without degrading the accuracy of the model [13].

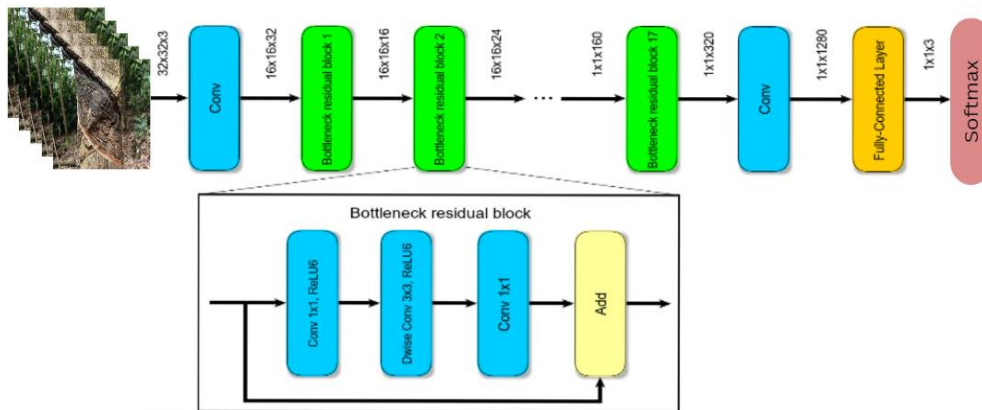


Figure 4. MobileNetV2 architecture on tree damage type

2.4. Data Pre-Processing

The data preparation process entails transforming unprocessed data into a more comprehensible format [14]. This stage is labeling the folder according to the name of the damaged tree, and resizing the image to 224x224 according to the network input on the MobileNetV2 architecture which consists of 224, 192, 160, or 128 [15]. Then we split the dataset into three categories, namely: 80% is allocated as training, 10% is for validation, and the other 10% is for testing. The model training process uses 80% of training data from the dataset. Valid data and comprising 10% of the dataset, validation is also required to ensure the model reads the data accurately. Finally, the test data is 10% of the dataset, used to test the model.

2.5. Model Training

TensorFlow performs the training process with some hyperparameters, which are settings not present in the model. Hyperparameters help find numbers that affect the model, but that are not from the data we are training [16]. The hyperparameters used are epoch, batch size, learning rate, and optimization [17]. The training process was conducted with the hyperparameters indicated in Table 1.

Table 1. Values of Hyperparameters

Parameter	Value/Type
Epoch	100
Learning rate	0,001
batch sizes	64, 32, 16, 10, 8
Optimizer	Adam
Callback Function	Validation Loss
Loss Function	Categorical Cross-Entropy

The data trains using the transfer learning method with the MobileNetV2 model. This involves extracting the weights from the pre-trained model, excluding the input layer and top layer. The input layer of the training dataset is then added for the tree decay type, and the top layer is substituted with a dense, regular layer incorporating normalization and a dropout rate of 0.5, using the Relu activation function. Finally, a dense layer with 16 tree decay type classes and the softmax activation function is added as the last layer. Training parameters batch size, epoch, and Adam optimizer with categorical_crossentropy loss function.

2.6. Model Evaluation

The accuracy measurement model is a useful tool for assessing performance. It provides a thorough insight into the correctness of several categorization model components through the usage of a confusion matrix. The confusion matrix for the two-path clustering is shown in Table 2. "True positive" denotes that the system accurately identified a positive image, while "True negative" means that the system successfully identified a negative photo. "False Positive" describes how the system incorrectly classifies a negative picture as positive, while "False Negative" describes how the system incorrectly labels a positive image as negative. Recall evaluates any capacity of the model to recognize all positive cases in the dataset (1), precision gives us data almost the level of certainty we will have within the model's forecasts of the information being positive (2), f1-score is a metric that compares the average level of recall and precision by setting the maximum worth to 1 and the lowest to 0 (3), accuracy refers to how closely the value predicted by the system matches the actual value (4), loss is a metric that indicates the degree of inaccuracy of the model's ability to solve problems and identify objects (5) [18].

$$Recall = \frac{TP}{(TP + FN)} \quad (1)$$

$$Precision = \frac{TP}{(TP + FP)} \quad (2)$$

$$F1 - Score = \frac{2 \times (Recall \times Precision)}{(Recall + Precision)} \quad (3)$$

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \quad (4)$$

$$Loss = \frac{(FP + FN)}{(TP + TN + FP + FN)} \quad (5)$$

2.7. Model Conversion to TensorFlow Lite Model

This step will convert the model into a TensorFlow Lite model, changing the model so that the model can be used in mobile applications on the local system without requiring an intermediary. The use of the TensorFlow library with the Python programming language is required for this research step. Model conversion to a TensorFlow Lite model is necessary to ensure efficient operation and make it possible to run on mobile applications with limited memory capacity [19]. This process allows the model to be used in mobile applications on local systems without intermediaries.

2.8. Model Implementation in Mobile Application

This step will create a mobile application to run the TFlite model (TensorFlow Lite). The building of mobile applications and the testing of the TFlite model on mobile applications are the two operations that comprise this step. The mobile application development process uses the Flutter framework [20]. This process builds a mobile application with the main focus to be able to display the classification results of the model directly. The TensorFlow Lite package allows the MobileNetV2 model to run with GPU delegates and provides for its performance settings [21].

The next stage is testing several photos that have been considered tree damage and testing them on a program to evaluate the TFlite model on mobile devices. The goal is to get an overview of applications that have specifications following the needs of the forest health monitoring process [22]. This application is designed to detect healthy images of damage types on trees that appear to be damaged based on the existing damage type classes.

3. RESULTS AND ANALYSIS

This section will delve into a detailed description of the results obtained from the classification process executed through the use of MobileNetV2's Convolutional Neural Network (CNN) architecture. In addition, we will present the evaluation results obtained from comparing the actual data labeling with the predictions made

by the model. Furthermore, we will describe the results of the model on the mobile application. Model training experiments using RTX 3060 mobile GPU.

3.1. Model Training and Evaluation

The model is trained using MobileNetV2 with parameters such as epochs, loss function, batch size, adding layers, and also early stopping callbacks. The model training process will be carried out by changing the batch size from 8, 10, 16, 32, and 64. Callback early stopping is used so that the model does not learn too much data, the training process will stop at a certain epoch [23]. Table 2 displays the model training outcomes.

Table 2. Model training results

No	batch	epoch	learning rate	loss training	accuracy training	validation loss	validation accuracy
1	8	44	0.001	0.0257	0.9937	0.0835	0.9688
2	10	43	0.001	0.0329	0.9945	0.0506	0.9812
3	16	45	0.001	0.0259	0.9961	0.0497	0.9875
4	32	38	0.001	0.0486	0.9922	0.0774	0.9875
5	64	54	0.001	0.0205	0.9945	0.0789	0.9844

After analyzing the training results, we determined that the best model was the one using batch 16 because it had the highest training and validation accuracy and the lowest validation loss. The training obtained the best results from training the model on batch 16 by stopping the training at epoch 45, this result shows better accuracy and loss than other parameters. In Figure 5, accuracy and loss graphs are shown.

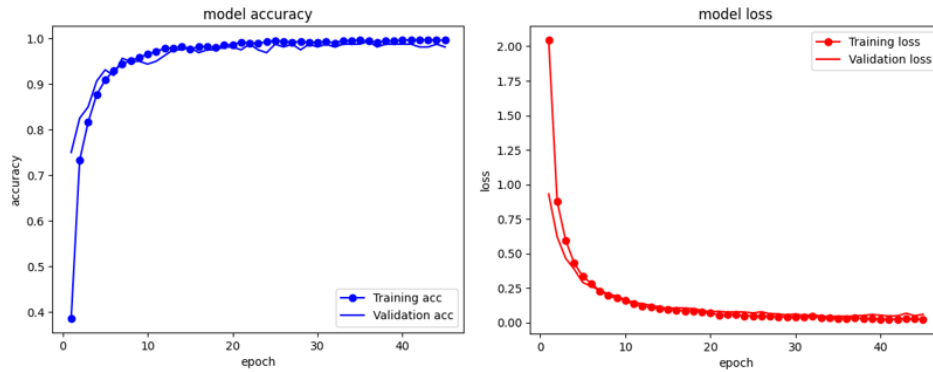


Figure 5. Graph of accuracy and loss with batch 16

The highest accuracy of the model is reported in Table 3, which includes the results of testing. In this test, 10% of the testing dataset is utilized to generate a confusion matrix and assess the model. The calculation formula (2) determines that the maximum precision is 100%, while the lowest precision is owned by broken or dead branches (code 22) and excessive branching (code 23). The maximum value in a recall is 100%, and the lowest value is 90%, which is included in the categories of Broken stems or roots (code 11)) and Broken leaf shoots (code 24)). We calculate these values using the formula (1). Broken stem or root owns the lowest value in the f1-score (95%), while Broken shoot leaf, Excessive branching, and the maximum value (100%) are owned by 11) (Broken stem or root), 24) (Broken shoot leaf), and 23) (Excessive branching), respectively. The values are calculated using formula (3). The accuracy obtained is 98.75% by dividing the total value of TP plus TN and then dividing by the total value of the increase in TP, FP, FN, and TN based on calculation formula (4), and the error is 1.25%

Table 3. MobileNetV2 Results of the trained MobileNet models

Tree Damage Type Class	Results		
	Precision	Recall	F1-score
Roots break or die	100%	100%	100%
Broken stems or roots	100%	90%	95%
Broken stem	100%	100%	100%
Brum on the root or stem	100%	100%	100%
Branches that are broken or dead	91%	100%	95%
Discoloured leaves	100%	100%	100%
Broken shoot leaves	100%	90%	95%
Gerowong	100%	100%	100%
loss or death of dominant shoots	100%	100%	100%
Cancer	100%	100%	100%

Tree Damage Type Class	Results		
	Precision	Recall	F1-score
Konk	100%	100%	100%
Liana	100%	100%	100%
Open wounds	100%	100%	100%
Excessive brum branching	91%	100%	95%
Resinosis gumosis	100%	100%	100%
Termite nest	100%	100%	100%
Accuracy	98.75%		
Error	1.25%		

After that, conversion into the TensorFlow Lite Model and testing with confusion matrix and testing on mobile applications. Information about the classification results performed on the TensorFlow Lite Model information is presented in Table 4. We found that there is a misclassification between the tree damage type code 11 against code 22 and code 12 against code 24. This occurs due to the similarity of the recognized pattern.

Table 4. Confusion Matrix Result of TensorFlow Lite Model

Tree Damage Type	Prediction															
	01	02	03	04	05	06	11	12	13	20	21	22	23	24	25	31
01	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0
03	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0
04	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0
05	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0	0
06	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	9	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	0	0	9	0	0	0	0	0	1	0	0
13	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10

3.2. Classification of Tree Damage Types in the Mobile Application

The next process is to convert the model that has been made into a TensorFlow Lite Model and continue the model's implementation on an application for mobile devices created using the Flutter framework. Mobile applications to identify and classify tree damage types will make it easier to monitor forest health. This application can analyze and provide predictive results of the type of tree damage classified by the model. After the user takes a picture of tree damage using the application, it will immediately analyze the image.

The results of tree damage classification will be presented in a user-friendly manner, showing the accuracy of the classified damage by displaying the highest predicted value, the name of the damage type, damage code, weighting value, date, and location. This can be seen in Figure 6 on the left side of the image which shows the results of the damage type class of loss or death of dominant shoots with a predictive value of 0.999 or 99.9%, damage code 21, weighting value 1.3, and location based on coordinates. This application can allow multiple outputs and inputs simultaneously according to needs. This can be seen in Figure 6 on the right part which shows the overall image results inputted by the user. The results can be seen in detail one by one, as the image shown is the result of the damage type class of loss or death of dominant shoots with a predictive value of 0.983 or 98.3%, damage code 21, weight value 1.3, and no location metadata. Additionally, user input can be done by opening the camera or selecting an image from storage.

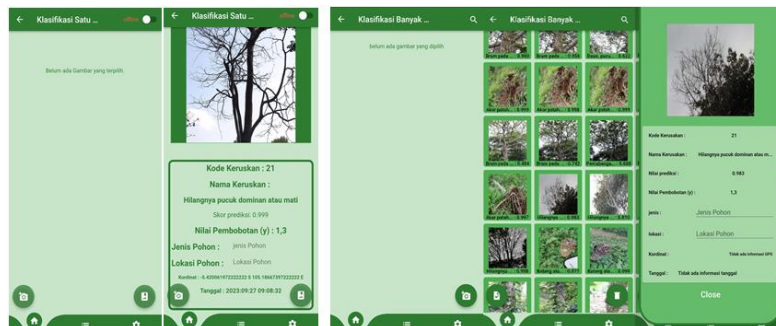


Figure 6. Prediction on mobile application

4. CONCLUSION

The primary result of this research is that the prediction accuracy of the model obtained during the training process reaches 98.75% and a loss of 0.0497 with MobileNetV2 architecture obtained from different batch and epoch trial variation schemes using early stopping callbacks. The model is also very effective in correctly classifying tree damage types and makes the results of testing the model through mobile applications, especially Android using Tensorflow Lite, very satisfactory. However, it is important to note that for accurate analysis, the image submitted by the user should be clear and not too far from the observed damage on the tree. This application is expected to help the observation of trees in the forest to be easier and faster.

REFERENCES

- [1] R. Safe'i, A. Darmawan, A. R. Irawati, A. Y. Pangestu, E. R. Arwanda, and A. N. Syahiib, "Cluster Analysis on Forest Health Conditions in Lampung Province," *Int. J. Des. Nat. Ecodynamics*, vol. 17, no. 2, pp. 257–262, 2022.
- [2] D. Pertiwi, R. Safe, and H. Kaskoyo, "IDENTIFIKASI KONDISI KERUSAKAN POHON MENGGUNAKAN METODE FOREST HEALTH MONITORING DI TAHURA WAR PROVINSI LAMPUNG Identification of Tree Damage Condition Using the Forest Health Monitoring Method at Tahura WAR Lampung Province," *J. Perenn.*, vol. 15, no. 1, pp. 1–7, 2019.
- [3] F. Ardiansyah and R. Safe'i, "Analysis of changes in health of coastal mangrove forest on the East Coast of lampung," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 755, no. 1, 2021.
- [4] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," no. October, 2017.
- [5] Y. Gulzar, "Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique," *Sustain.*, vol. 15, no. 3, 2023.
- [6] O. V. Putra, M. Z. Mustaqim, and D. Muriatmoko, "Transfer Learning untuk Klasifikasi Penyakit dan Hama Padi Menggunakan MobileNetV2," *Techno.Com*, vol. 22, no. 3, pp. 562–575, 2023.
- [7] M. R. Mustafa, A. A. Idris, L. N. Abdullah, and N. A. Nasharuddin, "Deep learning mango fruit recognition based on TensorFlow lite," *Int. J. Adv. Intell. Informatics*, vol. 9, no. 3, pp. 565–576, 2023.
- [8] I. R. Maulana, R. Safe'I, I. G. Febryano, H. Kaskoyo, and A. Rahmat, "The Relationship between the Health of Mangrove Forests and the Level of Community Welfare," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 1027, no. 1, 2022.
- [9] R. Safe'I, E. N. Puspita, and R. Hilmanto, "Assessment of tree vitality as an indicator of monitoring the health condition of community forest in agroforestry patterns," *Folia For. Pol. Ser. A*, vol. 64, no. 4, pp. 206–213, 2022.
- [10] K. Shridhar, F. Laumann, and M. Liwicki, "A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference," pp. 1–38, 2019.
- [11] N. N. Kamaron Arzar, N. Sabri, N. F. Mohd Johari, A. Amilah Shari, M. R. Mohd Noordin, and S. Ibrahim, "Butterfly Species Identification Using Convolutional Neural Network (CNN)," *2019 IEEE Int. Conf. Autom. Control Intell. Syst. I2CACIS 2019 - Proc.*, no. March 2021, pp. 221–224, 2019.
- [12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4510–4520, 2018.
- [13] F. D. Adhinata, N. A. F. Tanjung, W. Widayat, G. R. Pasfica, and F. R. Satura, "Comparative Study of VGG16 and MobileNetV2 for Masked Face Recognition," *J. Ilm. Tek. Elektro Komput. dan Inform.*, vol. 7, no. 2, p. 230, 2021.
- [14] S. Tang, S. Yuan, and Y. Zhu, "Data Preprocessing Techniques in Convolutional Neural Network Based on Fault Diagnosis towards Rotating Machinery," *IEEE Access*, vol. 8, pp. 149487–149496, 2020.
- [15] J. Ma, Y. Ushiku, and M. Sagara, "The Effect of Improving Annotation Quality on Object Detection Datasets: A Preliminary Study," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2022-June, pp. 4849–4858, 2022.
- [16] Afis Julianto, Andi Sunyoto, and Ferry Wahyu Wibowo, "Optimasi Hyperparameter Convolutional Neural Network Untuk Klasifikasi Penyakit Tanaman Padi," *Tek. Teknol. Inf. dan Multimed.*, vol. 3, no. 2, pp. 98–105, 2022.
- [17] T. Gressling, *84 Automated machine learning*. 2020.
- [18] E. Bagli and G. Visani, "METRICS FOR MULTI-CLASS CLASSIFICATION: AN OVERVIEW," *A WHITE Pap.*, pp. 1–17, 2020.
- [19] R. David *et al.*, "TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems," 2020.
- [20] P. Jindal, "A Research Paper on a Progress Tracking Application using Flutter and Firebase," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 5, pp. 1981–1985, 2021.
- [21] M. Reda, R. Suwwan, S. Alkafri, Y. Rashed, and T. Shanableh, "AgroAid: A Mobile App System for Visual Classification of Plant Species and Diseases Using Deep Learning and TensorFlow Lite," *Informatics*, vol. 9, no. 3, 2022.
- [22] A. Y. Pangestu, R. Safe'i, A. Darmawan, H. Kaskoyo, and R. Andrian, "Black-Box Testing on Web-GIS of Forest Health Monitoring Using Equivalence Partitioning Techniques," *AIP Conf. Proc.*, vol. 2563, no. October, 2022.
- [23] X. Ying, "An Overview of Overfitting and its Solutions," *J. Phys. Conf. Ser.*, vol. 1168, no. 2, 2019.

BIBLIOGRAPHY OF AUTHORS

Faishal Hariz Makaanim Gandadipoera is currently an undergraduate at the University of Lampung, seeking a degree in computer science. He has been enrolled in the study program since 2019 and continues to actively engage in various research areas within the field. His primary research interests revolve around mobile development, computer vision, and deep learning. His dedication to these subjects is evident in his commitment to staying up-to-date with the latest advancements and techniques.



Rico Andrian In 1999, he earned his Bachelor's in Computer Science from the Faculty of Mathematics and Natural Sciences, Padjadjaran University. In 2013, By earning a Master of Science in computer science from Bogor Agricultural Institute's prestigious Faculty of Mathematics and Natural Sciences, he broadened his knowledge and experience. Since 2005, he has actively contributed his expertise as a lecturer in Lampung University's prestigious Computer Science Study Program, demonstrating his dedication to education



Rahmat Safe'i has proudly served as an instructor at the Department of Forestry, Faculty of Agriculture, University of Lampung since 2006. His commitment to excellence in education and research has been recognized, as he has been entrusted with the honorable position of Head of the Master of Forestry Study Program starting in 2021. In 1999, he graduated with a Bachelor of Science in Forest Management from the IPB Faculty of Forestry. Subsequently, he continued his education to the master's level in the prestigious Forestry Science Study Program at the IPB Postgraduate Program, which he successfully achieved in 2005. With high dedication in his field, he continued his academic journey by earning his Doctorate in Forest Management Science from the College of Postgraduates in 2015. If you would like to get in touch with him, he can be reached