Mobile Application for Medicinal Plants Recognition from Leaf Image Using Convolutional Neural Network

D Sugiarto[1], J Siswantoro[*2], M F Naufal[3], B Idrus[4]

[1,2,3]Department of Informatics Engineering, University of Surabaya, Indonesia

4Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM, Bangi, Selangor D. E., Malaysia

E-mail: s160418025@student.ubaya.ac.id[1], joko_siswantoro@staff.ubaya.ac.id[2], faridnaufal@staff.ubaya.ac.id[3], bahari@ukm.edu.my[4]

Submitted: 6 December 2022, revised: 12 February 2022, accepted: 13 February 2023

Abstract. Indonesia is a country that has thousands of plant types that can be used as traditional medicine. However, some people have not utilized this potential optimally due to the lack of knowledge about medicinal plants' types, benefits, and substances. Therefore, there is a need to develop an application that can identify medicinal plants that grow in Indonesia and provide information about the benefits and content of the substances contained in them. In this study, medicinal plants will be recognized using a mobile application from leaf images based on a pre-trained convolutional neural network (CNN) with a transfer learning technique. Three pre-trained CNN architectures, namely VGG-16, MobileNetV2, and DenseNet-121, are explored for medicinal plant recognition. Hyperparameter tuning is performed at the fully connected layer of all architectures with 20 possible modifications to find the best model. The experimental results on 24 types of medicinal plants show that the model based on MobileNetV2 achieves the best classification accuracy of 97.74%. The best model is obtained by modifying the fully connected layer of MobileNetV2 into three dense layers with the number of neurons 736, 448, and 928, respectively. After the application recognizes the types of medicinal plants, information about the benefits and substances contained in them is displayed to the user.

Keywords: medicinal plants recognition; leaf image; convolutional neural network; transfer learning.

1. Introduction

Plants are an essential element in the ecosystem that act as producers for humans and other living things. Several types of plants have been used by humans in traditional medicine since ancient times [1]. Medicinal plants have been used for generations by the Indonesian people in traditional medicine since 800 AD [2]. Furthermore, medicinal plants have been used for thousands of years to maintain public health in the Asian continent and have also built a unique medical system empirically in the Asian region [3].

Currently, some Indonesians prefer to use modern medicines prescribed by physicians or purchased directly from pharmacies [4]. In fact, some modern medicines have side effects that cause serious medical problems for patients [5]. On the other hand, using traditional medicines has several advantages compared to modern medicines, such as relatively low side effects, more affordable prices, and good clinical effectiveness [6].

The low knowledge of Indonesian people, especially young people, about medicinal plants make some people hesitate to consume traditional medicines from medicinal plants. Based on research conducted by Rasna [7] regarding teenager knowledge of medicinal plants, it was found that 70% of respondents did not have knowledge of medicinal plants. This shows that teenagers' knowledge of medicinal plants is minimal. Therefore, an application is needed that can help the public know the types of medicinal plants, the substances contained, and the efficacy of these medicinal plants.

Several researchers have proposed methods for plant-type recognition using leaf images. However, most of them are not used specifically to identify medicinal plants that grow in Indonesia. In addition, these methods are not used to search for information about substances contained in plants and their properties, as reported in the following studies. Chaki et al. [8] proposed a method for plant leaves recognition using shape and texture features. Two classifiers, namely multi-layered perceptron and neuro-fuzzy controller were employed for plant leaf classification with 31 classes. However, the proposed method only achieved the highest accuracy of 67.7%. Liu and Kan [9] used improved deep belief networks (DBN) and several shapes and texture features to classify leaf species. The ICL leaf image dataset [10] consisting of 220 leaf species was used to evaluate the proposed method and obtained an average classification accuracy of 93.9%. Prasetya et al. [11] developed a mobile application to recognize the types of medicinal plants by employing the template matching method. However, the proposed application can recognize only two plant types with a classification accuracy of 81.25% and 68.75%, respectively. Yang [12] used texture and shape features in his proposed method to identify plant leaves from an image. The method classified the image using a 1-nearest neighbor classifier based on extracted features, resulting in an accuracy of more than 95% on three datasets. Zhang et al. [13] proposed a method for leaf image classification on several benchmark datasets using a bag of features and a linear support vector machine (SVM). The proposed method achieves calcification accuracy between 94.19% to 98.53%. Mahajan et al. [14] proposed a method for recognizing plant species from leaf images using morphological features and SVM with an adaptive boosting technique. The proposed method was validated using the Flavia dataset [15] and yielded an accuracy of 94.72%.

Pre-trained convolutional neural network (CNN) model has also been used to recognize medicinal plants from leaf images as reported in [15] and [16]. Sun et al. [16] used a CNN model with ResNet architecture [18] to recognize 10,000 leaf images from 100 ornamental plant species. The experimental results showed that ResNet model can achieve an accuracy of 91.78%. Lana [17] proposed the use of a CNN model with the VGG19 architecture [19] for the identification of medicinal plant species with a mobile application. The proposed model can produce an accuracy of 97.52%. However, it can only be used to identify 15 types of medicinal plants.

Several pre-trained CNN models have been widely used to classify images in various fields, including leaf images. However, the model's classification accuracy varies from case to case. Therefore, it is necessary to investigate the pre-trained CNN architecture that can produce the best performance in recognizing medicinal plants from leaf images. With the advent of technology, a majority of Indonesian youths, especially university students, possess mobile phones. This is particularly evident among students, with over 95% owning a mobile phone [20]. Hence, developing a mobile application that can identify different types of medicinal plants will aid Indonesian youth in gaining knowledge about the various medicinal plants and their healing properties. In this study, a mobile application based on pre-trained CNN is developed to recognize the types of medicinal plants that grow in Indonesia from leaf images. CNN is a popular deep-learning algorithm for image recognition and classification [21], making

it suitable for recognizing medicinal plants from leaf images. The application will capture the image of medicinal plant leaves using a smartphone camera. The captured image is then used to recognize the type of plant using a pre-trained CNN model with transfer learning techniques. Transfer learning techniques can be used to quickly build a CNN model with good performance without needing a large training dataset. Furthermore, transfer learning techniques allow a CNN model that has been trained for a particular classification problem to be used in other classification problems [22]. Based on the type of plant obtained from the CNN output, information about the substances and the medicinal plant's efficacy will be displayed to the user.

2. Material and method

In this study, several steps were carried out to develop a mobile application for medicinal plat recognition from leaf images. The steps included developing a medicinal plant leaf images dataset, constructing CNN models, image pre-processing, training the CNN models, model evaluation, and implementing the model on a mobile application for medicinal plant recognition, as shown in Figure 1. The following subsections describe the detail of each step.

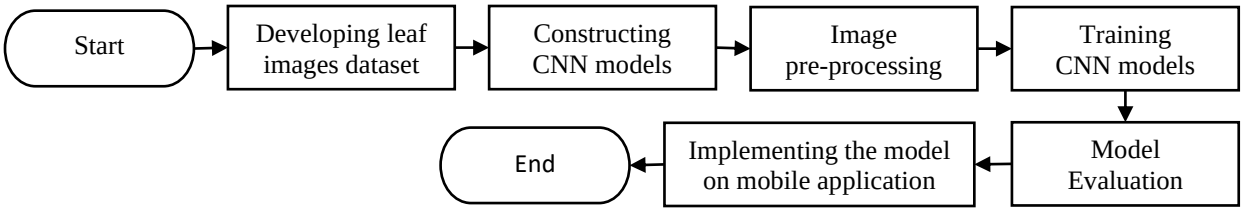


Figure 1. The steps performed in this study.

2.1. Images dataset

In this study, a medicinal plant leaf images dataset was developed, which consists of 24 types of medicinal plants. This dataset is one of the contributions of this study. The names of medicinal plants in the dataset are (a) Africa leaf, (b) Indian apple, (c) bilimbi, (d) binahong, (e) brotowali, (f) violet leaf, (g) Javanese ginseng, (h) red ginger, (i) guava, (j) lime, (k) basil, (l) aromatic ginger, (m) cat's whiskers, (n) fingerroot(o) turmeric, (p) aloe vera, (q) mangosteen, (r) Jamaica vervain, (s) papaya, (t) Indonesian bay leaf, (u) bitter leaf, (v) celery, (w) betel, and (x) temu ireng. Samples of medicinal plants were purchased from several plant sellers in Surabaya, Indonesia. The images were captured using a Samsung S21 FE smartphone camera with 6 GB RAM and camera resolution of 12 MP. Two types of background, homogeneous and inhomogeneous backgrounds, were used to capture the image of leaf with lighting from room lights. Several variations of distance, orientation, and point of view were used when capturing leaf images. Each image in the dataset was acquired in the RGB (Red, Green, Blue) color space [23] and then stored in a JPEG file. The dataset has 50 to 200 images for each type of medicinal plant. The total number of leaf images in the dataset was 4363, with a composition of 2798 leaf images captured using a homogeneous background and 1715 leaf images using an inhomogeneous background. The examples of leaf images in a dataset captured with homogeneous and inhomogeneous backgrounds can be seen in Figure 2 and Figure 3, respectively.

Furthermore, information about the benefits and substances contained in each type of medicinal plant was also collected in this study. The benefits and substances contained in medicinal plants were collected from various sources, such as scientific articles in journals, research reports on medicinal plants, and other information sources that an expert has validated. The collected data was stored in the database and displayed on the application according to the classification results obtained.

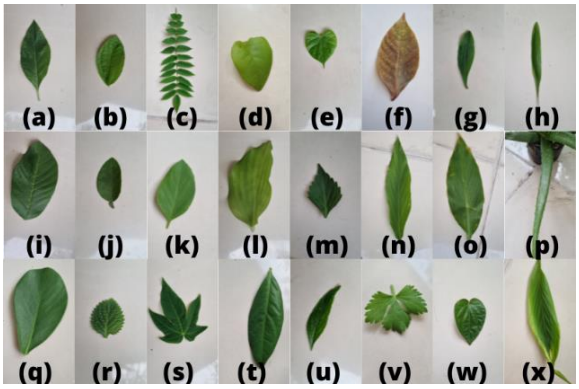


Figure 2. The examples of leaf images in the dataset captured with homogeneous background

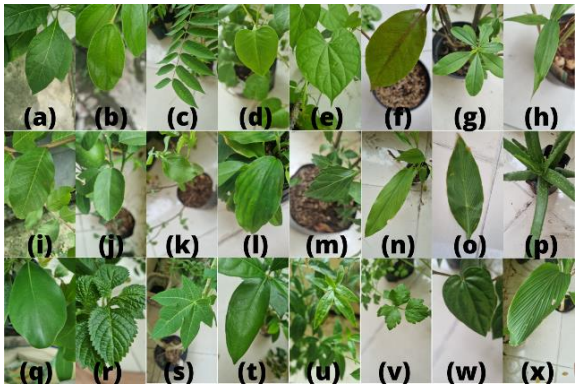


Figure 3. The examples of leaf images in the dataset captured with inhomogeneous background

2.2. Constructing CNN models

In this study, three pre-trained CNN architecture models were used to classify medicinal plant leaf images, namely VGG-16 [19], MobileNetV2 [24], and DenseNet-121 [25]. VGG-16 is a CNN pre-trained model that won the ImageNet Challenge in 2014. This model has 16 layers consisting of 16 convolutional layers, two fully connected layers, and one output layer. The activation function of the convolutional layer and fully connected layer is a rectified linear unit (ReLU) function as in equation (1). At the same time, the output layer uses the softmax activation function as in equation (2). In addition, there are five max pooling layers, each after the second, fourth, seventh, 10th, and 13th convolutional layers. The kernel sizes used in the convolutional and max pooling layers are 3×3 and 2×2, respectively. The input of the VGG-16 model is a 224×224×3 image. The details of the VGG-16 architecture can be seen in Table 1.

Table 1. The detail of VGG-16 architecture

Layer	Number of filter	Input size	Output size
Convolutional	64	224×224×3	224×224×64
Convolutional	64	224×224×64	224×224×64
Max pooling	-	224×224×64	112×112×64
Convolutional	128	112×112×64	112×112×128
Convolutional	128	112×112×128	112×112×128
Max pooling	-	112×112×128	56×56×128
Convolutional	256	56×56×128	56×56×256
Convolutional	256	56×56×256	56×56×256
Convolutional	256	56×56×256	56×56×256
Max pooling	-	56×56×256	28×28×256
Convolutional	512	28×28×256	28×28×512
Convolutional	512	28×28×512	28×28×512
Convolutional	512	28×28×512	28×28×512
Max pooling	-	28×28×512	14×14×512
Convolutional	512	14×14×512	14×14×512
Convolutional	512	14×14×512	14×14×512
Convolutional	512	14×14×512	14×14×512
Max pooling	-	14×14×512	7×7×512
Fully connected	-	1×1×25088	1×1×4096
Fully connected	-	1×1×4096	1×1×4096
Fully connected	-	1×1×4096	1×1×1000

$$f(x) = \max(0, x) \tag{1}$$

$$\sigma(x_1, x_2, \dots, x_n)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{2}$$

MobileNetV2 was the second generation of MobileNetV1 [26]. In the experiment conducted by Sandler et al. [24] using the ImageNet dataset, MobileNetV2 got better accuracy results than MobileNetV1 even though it used fewer parameters than MobileNetV1. MobileNetV2 is also suitable for mobile devices due to efficient memory consumption and fast execution time compared to other pre-trained models. The basic structure of MobileNetV2 is a depth-separable convolution bottleneck block with residuals using stride $s = 1$ or $s = 2$. This residual bottleneck block consists of three layers, which are a convolutional layer with a kernel size of 1×1 and the activation function ReLU6, a depthwise convolution layer with a kernel size of 3×3 and an activation function of ReLU6, and a convolutional layer with a kernel size of 1×1 and linear activation function. The ReLU6 function is a modification of the ReLU function with the upper limit of its activation value is 6, as defined in equation (3). In addition, this block also uses an expansion factor of $e = 6$. The structure of the residual bottleneck block, which transforms input with c channel into output with c' channel using stride s and expansion factor t can be seen in Table 2.

$$f(x) = \min(\max(0, x), 6) \tag{3}$$

Table 2. The structure of bottleneck residual layer

Layer	Input size	Output size
Convolutional, 1×1, ReLU6	$h \times w \times c$	$h \times w \times ec$
Depthwise convolutional, 3×3, ReLU6	$h \times w \times ec$	$h/s \times w/s \times ec$
Convolutional, 1×1, linier	$h/s \times w/s \times ec$	$h/s \times w/s \times c'$

The MobileNetV2 architecture consists of a convolutional layer with 32 filters and a kernel size of 3×3, followed by several residual bottleneck layers replicated r times with varying stride s values, expansion factor e, and the number of filters c. After that, there is a convolutional layer with 1280 filters, a kernel size of 1×1, and an average pooling layer with a kernel size of 7×7. The last layer is a convolutional layer with k (number of classes) filters and a kernel size of 1×1. All convolutional layers use the ReLU6 activation function. The input from MobileNetV2 is a 224×224×3 image. The detail of The MobileNetV2 architecture can be seen in Table 3.

Dense convolutional network (DenseNet) is a pre-trained CNN model that connects each layer to all other layers in a feed-forward manner [25]. In DensNet, the output of each layer and the previous layer are used as input for the subsequent layers. Several advantages can be obtained by applying such a structure: it can reduce the occurrence of vanishing gradients, has fewer parameters, encourages feature reuse, and strengthens feature propagation. This study used the DenseNet-121 architecture with 121 parameterized layers consisting of one convolutional layer with a kernel size of 7×7, followed by max pooling with a kernel size of 3×3, 116 dense block layers, three transition layers, global average pooling with a kernel size of 7×7, and a fully connected layer with 1000 neurons. The fully connected layer employs a softmax as the activation function. The dense block layer consists of two operations. The first one is batch normalization with ReLu activation function folowed by a convolutional layer with 32 filters and a kernel size of 1×1. The second operation is batch normalization with ReLu activation function folowed by convolutional layer with 32 filters and 3×3 kernel size. This operation is replicated r times on each dense block. The transition layer consists of a convolutional layer with a kernel size of 1×1 and an

average pooling layer with a kernel size of 2×2. The input from DenseNet-121 is a 224×224×3 image. The detail of DenseNet-121 architecture can be seen in Table 4.

Table 3. The detail of MobileNetV2 architecture

Layer	Input size	*r*	*s*	*e*	*c*
Convolutional 3×3	224×224×3	1	2	-	32
Bottleneck	112×112×32	1	1	1	16
Bottleneck	112×112×16	2	2	6	24
Bottleneck	56×56×24	3	2	6	32
Bottleneck	28×28×32	4	2	6	64
Bottleneck	14×14×64	3	1	6	96
Bottleneck	14×14×96	3	2	6	160
Bottleneck	7×7×160	1	1	6	320
Convolutional 1×1	7×7×320	1	1	-	1280
Average pooling 7×7	7×7×1280	1	-	-	-
Convolutional 1×1	1×1×1280	-	-	-	*k*

Table 4. The detail of DenseNet-121 architecture

Layer	Detail	*r*	Output size
Input	-	-	224×224×3
Convolutional	7×7 conv stride 2	-	112×112×64
Pooling	3×3 mp stride 2	-	56×56×64
Dense block	batch normalization, ReLU, 1×1 convolutional batch normalization, ReLU, 3×3 convolutional	6	56×56×256
Transition	1×1 conv, 3×3 ap stride 2	-	28×28×128
Dense block	batch normalization, ReLU, 1×1 convolutional batch normalization, ReLU, 3×3 convolutional	12	28×28×512
Transition	1×1 conv, 3×3 ap stride 2	-	14×14×256
Dense block	batch normalization, ReLU, 1×1 convolutional batch normalization, ReLU, 3×3 convolutional	24	14×14×1024
Transition	1×1 conv, 3×3 ap stride 2	-	7×7×512
Dense block	batch normalization, ReLU, 1×1 convolutional batch normalization, ReLU, 3×3 convolutional	16	7×7×1024
Pooling	7×7 gap stride 7	-	1×1×1024
Fully connected	1000 neuron, softmax	-	1×1×1000

In this study, the three pre-trained CNN models were used to classify images of medicinal plant leaves using the transfer learning technique. With the transfer learning technique, the weights in the convolutional layer in each pre-trained CNN model were frozen for feature extraction. Meanwhile, the fully connected layers in each model were adjusted and retrained to produce the best accuracy in classifying the image of medicinal plant leaves.

The output of the convolutional layer was fed into the 2D global average pooling layer to reduce its dimensions. The fully connected layers of each pre-trained CNN model were adjusted by modifying the number of dense layers and the number of neurons in them. The number of dense layers used in this study was between two and three with the ReLU activation function. Furthermore, the number of neurons in each dense layer was randomly generated between 128 and 1024. An output layer, with 24 neurons and the softmax activation, was added to the end of the fully connected layer. In this study, 20 fully connected layer modification variations were made, as shown in Table 5.

Tabel 5. The variation of fully connected layer modification

Model	The number of neurons in each dense layer		
	VGG-16	**MobileNetV2**	**DenseNet-121**
M1	128, 128	128, 128	128, 128
M2	192, 736	800, 1024	928, 896, 128
M3	800, 832, 128	704, 544, 128	192, 864, 512
M4	736, 352, 576	960, 672, 448	256, 224, 160
M5	672, 320, 576	480, 320, 352	320, 544, 800
M6	480, 192, 288	480, 544, 192	960, 288, 448
M7	320, 160, 224	576, 608, 768	960, 800, 960
M8	416, 544, 896	960, 896, 256	320, 160, 448
M9	128, 832, 800	896, 928, 320	640, 416, 384
M10	320, 928, 160	704, 288, 640	736, 640, 864
M11	288, 352, 1024	608, 352, 544	544, 384, 128
M12	736, 320, 832	768, 512, 992	864, 928, 480
M13	224, 384, 608	352, 480, 288	512, 448, 576
M14	928, 800, 896	544, 224, 672	960, 192, 704
M15	224, 608, 864	160, 480, 960	448, 192, 160
M16	384, 672, 352	160, 256, 928	1024, 352, 640
M17	128, 192, 576	448, 800, 768	800, 832, 128
M18	512, 992, 224	544, 832, 288	768, 608, 480
M19	672, 576, 128	736, 448, 928	288, 608, 832
M20	448, 864, 832	128, 192, 832	960, 352, 864

2.3. Image pre-processing

The medicinal plant leaf image must be pre-processed to match the input characteristics of the pre-trained CNN model. The first pre-processing performed in this study was resizing the image to fit the input size of the pre-trained CNN model. This study used the bilinear interpolation method to resize the image. The next pre-processing was image normalization. Image normalization was performed by dividing the pixel intensity in the range [0,255] with 255 to produce a value in the range [0,1].

2.4. Training CNN model

Before training the CNN models, the image dataset was partitioned into two mutually exclusive subsets with a proportion of 80% for training data and 20% for testing data. The training data was used to train the fully connected layers of each pre-trained CNN model with epoch 50. The training algorithm used in this study was stochastic gradient descent (SGD), with a learning rate of 0.004 and a momentum of 0.9. The loss function used in the training process was categorical cross entropy as defined in equation (4),

$$Loss = -\sum_{i=1}^{N_c} y_i \log \hat{y}_i \quad (4)$$

where N_c is the number of classes, y_i is the target value of the i^{th} class, and $\hat{y}_i$ is the model output of the i^{th} class. This loss function was used to measure how close the model output is to the target value in each class. A small loss function value will characterize the best model.

2.5. Model evaluation

Some experiments have been conducted to evaluate the CNN models in classifying medicinal plant leaf images. Evaluation was carried out using data testing on the CNN models that have been retrained to measure model performance. Several metrics were used to measure the performance of each CNN model with various modifications, namely accuracy, precision, recall, and F_1 score. Accuracy was used to assess the global performance of each CNN model and was defined as the proportion of the number of leaf images in the testing data that can be classified correctly by the CNN model to the total of leaf images in the testing data, as in equation (5),

$$accuracy = \frac{N_t}{N_T} \times 100\% \quad (5)$$

where N_t is the number of leaf images in the testing data, which the CNN model correctly classifies, and N_T is the total of leaf images in the testing data.

Since the number of leaf images in each class is different, the performance of the CNN model for each class was further evaluated using precision, recall, and F_1 scores. In the multi-class case, a class's precision, recall, and F_1 score were calculated by assuming the class is positive and the other classes are negative. After each class's precision, recall, and F_1 scores were obtained, they were then averaged to obtain the overall precision, recall, and F_1 scores. Precision, recall, and F_1 score of each class were calculated using equations (6), (7), and (8), respectively,

$$precision = \frac{TP}{FP + TP} \quad (6)$$

$$recall = \frac{TP}{FN + TP} \quad (7)$$

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \quad (8)$$

where TP is the amount of data in the positive class, classified correctly, and FN is the amount of data in the positive class, classified as a negative class. FP is the amount of data in the negative class, which is classified as a positive class.

2.6. Implementation CNN model on mobile application

The last process was implementing the best CNN model on a mobile application to recognize the types of medicinal plant leaves and display information about the benefits and substances contained therein. To perform recognition, the user can capture a leaf image directly through the smartphone camera or select an image previously captured from the gallery, as shown in Figure 3.

The mobile application was developed using Kotlin programming language. While the leaf image recognition process was implemented using Python programming language. Several open-source libraries for Python were also used, such as OpenCV [27] for image pre-processing, Scikit-learn [28] for creating training data and testing and evaluating the CNN models, TensorFlow [29] for training the CNN model, and TensorFlow lite for implementing the CNN model in mobile application.

A series of processes will be carried out on the mobile application to recognize the medicinal plant from the leaf image, as shown in Figure 4. After the leaf image is inputted into the application, pre-processing is performed to resize and normalize the image as described in section 2.3. The processed image is then inputted into the trained CNN model to predict the type of medicinal plant. The type of plant predicted by the CNN model is then used to find data on the benefits and substance content of medicinal plants that have been stored in a database. Finally, information about the type of medicinal plant, its benefits, and its substances contained are displayed to the user. The application should be installed on a smartphone with at least 6GB of RAM to function properly.

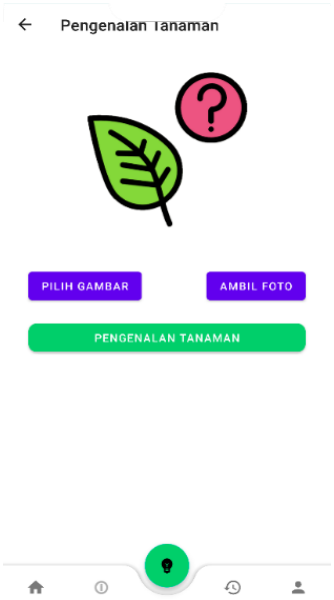


Figure 3. Recognition page on the mobile application

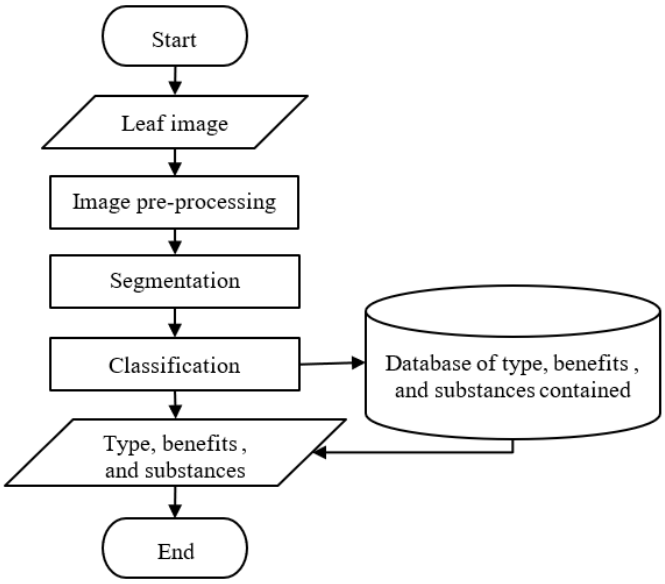


Figure 4. The process flow of leaf image recognition in a mobile application

3. Results and discussion

The classification accuracy of the 20 modified models for each pre-trained CNN model is tabulated in Table 6. As shown in Table 6, the classification accuracy of each pre-trained CNN architecture varies between 17.86% to 97.47%. The modified model based on the MobileNetV2 architecture resulted in the highest average classification accuracy of 95.87%, followed by DenseNet-121 at 94.50%, and the lowest was VGG-16 at 32.16%. The bold texts in Table 6 indicate each model's three highest classification accuracy.

For the MobileNetV2 architecture, the highest classification accuracy of 97.47% was achieved by the M19 model, which has three dense layers with 736, 448, and 928 neurons. The second highest classification accuracy of 97.24% was achieved by the M9 model, which has three dense layers with 896, 928, and 320 neurons. The third highest classification accuracy of 97.00% was achieved by the M11 model, which has three dense layers with 576, 608, and 768 neurons.

For the DenseNet-121 architecture, the highest classification accuracy of 96.54% was achieved by the M12 model, which has three dense layers with 864, 928, and 480 neurons. The second highest classification accuracy of 96.08% was achieved by the M17 model, which has three layers dense with 800, 832, and 128 neurons, and the M18 model, which has three dense layers with 768, 608, and 480 neurons.

For the VGG-16 architecture, the highest classification accuracy of 39.17% was achieved by the M5 model, which has three dense layers with 672, 320, and 576 neurons. The second highest classification accuracy of 38.25% was achieved by the M18 model, which has three layers dense with 512, 992, and 224 neurons. The third highest classification accuracy of 38.02% was achieved by the M10 model, which has three dense layers with 320, 928, and 160 neurons.

Further analysis was performed to see the performance of the best CNN model, which is the M19 model based on the MobleNetV2 architecture in each class, by calculating the precision, recall, and F_1 scores, as tabulated in Table 7. As can be seen in Table 7, the precision, recall, and F1 scores for the M19 model based on the MobileNetV2 architecture are greater than or equal to 0.92, 0.89, and 0.92, respectively. The average precision, recall, and F_1 score for all classes is 0.97. From 24 medicinal plant classes, the M19 model based on the MobileNetV2 architecture produced an F_1 score greater than or equal to 0.95 in 20 classes. The best performance with precision, recall, and F_1 score of 1.00 occurred in the violet leaf, aromatic ginger, aloe vera, and mangosteen leaf classes. These results indicate that the model can recognize all leaf images in the classes of violet leaf, aromatic ginger, aloe vera, and mangosteen, and no leaf images from other classes are recognized as leaf images from those classes. This can happen because violet leaves have a different color from other leaves. Meanwhile, aromatic ginger, aloe vera, and mangosteen have different leaf shapes than other leaves.

Table 6. Accuracy of the pre-trained CNN model

Model	Accuracy (%)		
	VGG-16	**MobileNetV2**	**DenseNet-121**
M1	26,96	94,12	91,24
M2	31,45	96,31	95,51
M3	35,14	95,85	93,32
M4	33,87	96,20	93,20
M5	**39,17**	95,51	92,40
M6	34,79	95,85	95,39
M7	27,76	96,67	95,74
M8	34,33	96,54	93,43
M9	22,58	**97,24**	95,05
M10	**38,02**	96,54	95,05
M11	27,19	**97,00**	94,59
M12	35,25	96,77	**96,54**
M13	26,50	95,51	94,70
M14	36,41	95,51	94,12
M15	31,80	94,35	93,89
M16	33,53	94,01	95,16
M17	17,86	96,20	**96,08**
M18	**38,25**	95,85	**96,08**
M19	37,21	**97,47**	93,89
M20	35,14	93,89	94,70
Average	32,16	95,87	94,50

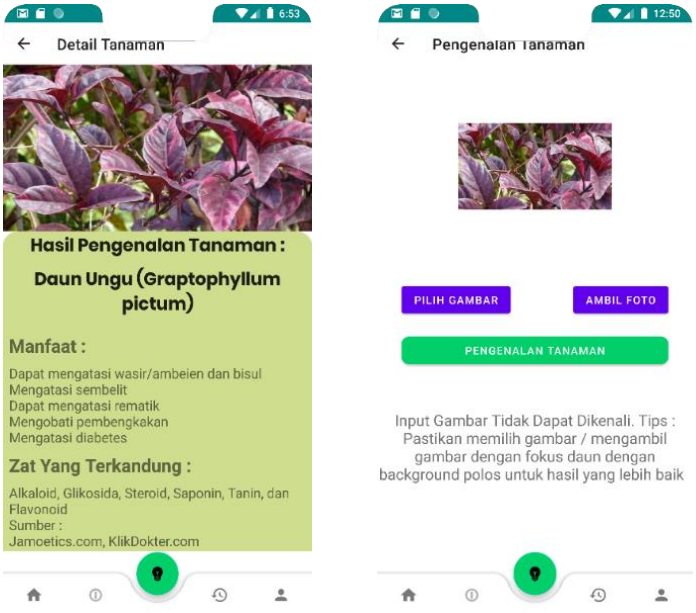
The number of images that cannot be recognized correctly by the M19 model based on the MobileNetV2 architecture is less than or equal to three images per class. Since there are several classes that have fewer images than other classes, F1 scores in those classes are smaller compared to other classes. From the experimental results, there are only four out of 24 classes that have an F_1 score below 0.95 but still greater than 0.90, namely basil (0.92), fingerroot (0.92), betel (0.92), and temu ireng (0.93). The results indicate that the M19 model based on the MobleNetV2 architecture has good performance and can be implemented in mobile applications to identify the type of medicinal plant.

After the best CNN model is implemented in the mobile application, a series of tests were performed to determine the threshold value for the highest output probability of the model to determine whether the application can recognize plants or not with good results. From the test results, the optimum threshold value was 0.5. However, the user can change the threshold value through the settings menu provided in the application.

If the value of the highest output probability of the CNN model is greater than or equal to the predetermined threshold value, then the prediction of medicinal plant type from the CNN model will be displayed to the user along with the benefits and substances contained therein, as shown in Figure 5(a). Otherwise, a message will be displayed to the user that the application cannot recognize the input leaf image, and the user will be asked to re-enter a better leaf image, as shown in Figure 5(b).

Table 7. Precision, recall, and F_1 score for the M19 model based on the MobileNetV2 architecture

Classes	Precision	Recall	F_1 score
Africa leaf	1,00	0,95	0,98
Indian appe	0,93	1,00	0,96
Belimbi	1,00	0,91	0,95
Benahong	1,00	0,96	0,98
Brotowali	0,95	1,00	0,97
Violet leaf	1,00	1,00	1,00
Javanese ginseng	0,97	0,97	0,97
Red ginger	1,00	0,97	0,99
Guava	0,97	1,00	0,99
Lime	0,95	0,97	0,96
Basil	0,94	0,89	0,92
Aromatic ginger	1,00	1,00	1,00
Cat's whiskers	1,00	0,98	0,99
Fingerroot	0,92	0,92	0,92
Turmeric	0,94	1,00	0,97
Aloe vera	1,00	1,00	1,00
Mangosteen	1,00	1,00	1,00
Jamaica vervain	0,98	1,00	0,99
Papaya	1,00	0,98	0,99
Indonesian bay leaf	0,94	1,00	0,97
Bitter leaf	0,98	0,95	0,96
Celery	0,95	1,00	0,97
Betel	0,96	0,89	0,92
Temu ireng	0,96	0,90	0,93
Average	0,97	0,97	0,97



(a) (b)

Figure 5. Classification results shown to the user: (a) the application can recognize leaf image, (b) the application cannot recognize leaf image.

4. Conculsion

In this research, a mobile application was developed to recognize medicinal plants from leaf images based on a convolutional neural network (CNN). Three pre-trained model architectures CNN, VGG-16, MobileNetV2, and DenseNet-121, were investigated as a basis for classifying 24 medicinal plants by modifying the fully connected layer. The experimental results showed that the MobileNetV2-based model obtains the highest classification accuracy of 97.74% by modifying the fully connected layer into three dense layers with 736, 448, and 928 neurons. The model with the highest classification accuracy was then implemented in a mobile application for medicinal plant recognition. After recognizing the type of medicinal plant, the developed application will display information about the benefits and the substances contained in the plant.

5. Acknowledgement

The authors would like to thank the University of Surabaya for providing financial support and facilities to conduct this research through Research Grant No. 028/ST-Lit/LPPM-01/FT/V/2022.

References

[1] F. Jamshidi-Kia, Z. Lorigooini, and H. Amini-Khoei, "Medicinal plants: Past history and future perspective," *J. herbmed Pharmacol.*, vol. 7, no. 1, 2018.

[2] S. Astutik, J. Pretzsch, and J. Ndzifon Kimengsi, "Asian medicinal plants' production and utilization potentials: A review," *Sustainability*, vol. 11, no. 19, p. 5483, 2019.

[3] W.-Y. Lee, C.-Y. Lee, Y.-S. Kim, and C.-E. Kim, "The methodological trends of traditional herbal medicine employing network pharmacology," *Biomolecules*, vol. 9, no. 8, p. 362, 2019.

[4] S. Ramadhani, J. Iskandar, R. Partasasmita, and E. N. Rohmatullayaly, "Local knowledge of Sundanese village people on traditional medicine: A case study in Cibeurih Hamlet, Nagarawangi Village, Sumedang District, Indonesia," *Biodiversitas J. Biol. Divers.*, vol. 22, no. 5, 2021.

[5] N. Tran, B. Pham, and L. Le, "Bioactive compounds in anti-diabetic plants: From herbal medicine to modern drug discovery," *Biology (Basel).*, vol. 9, no. 9, p. 252, 2020.

[6] A. Mirzaie, M. Halaji, F. S. Dehkordi, R. Ranjbar, and H. Noorbazargan, "A narrative literature

review on traditional medicine options for treatment of corona virus disease 2019 (COVID-19)," *Complement. Ther. Clin. Pract.*, vol. 40, p. 101214, 2020, doi: https://doi.org/10.1016/j.ctcp.2020.101214.

[7] I. W. Rasna, "Pengetahuan dan sikap remaja terhadap tanaman obat tradisional di Kabupaten Buleleng dalam rangka pelestarian lingkungan: Sebuah kajian ekolinguistik," *J. Bumi Lestari*, vol. 10, no. 2, pp. 321–332, 2010.

[8] J. Chaki, R. Parekh, and S. Bhattacharya, "Plant leaf recognition using texture and shape features with neural classifiers," *Pattern Recognit. Lett.*, vol. 58, pp. 61–68, 2015.

[9] N. Liu and J. Kan, "Improved deep belief networks and multi-feature fusion for leaf identification," *Neurocomputing*, vol. 216, pp. 460–467, 2016.

[10] R. Hu, W. Jia, H. Ling, and D. Huang, "Multiscale distance matrix for fast plant leaf recognition," *IEEE Trans. image Process.*, vol. 21, no. 11, pp. 4667–4672, 2012.

[11] S. K. Prasetya, R. K. Budhi, and D. T. Hidayat, "Rancang Bangun Aplikasi Untuk Identifikasi Daun Tanaman Toga Berbasis Mobile Menggunakan Template Matching," in *Seminar Nasional Ilmu Terapan*, 2017, vol. 1, no. 1, pp. C14-1.

[12] C. Yang, "Plant leaf recognition by integrating shape and texture features," *Pattern Recognit.*, vol. 112, p. 107809, 2021.

[13] Y. Zhang, J. Cui, Z. Wang, J. Kang, and Y. Min, "Leaf image recognition based on bag of features," *Appl. Sci.*, vol. 10, no. 15, p. 5177, 2020.

[14] S. Mahajan, A. Raina, X.-Z. Gao, and A. Kant Pandit, "Plant recognition using morphological feature extraction and transfer learning over SVM and AdaBoost," *Symmetry (Basel).*, vol. 13, no. 2, p. 356, 2021.

[15] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, "A leaf recognition algorithm for plant classification using probabilistic neural network," in *2007 IEEE international symposium on signal processing and information technology*, 2007, pp. 11–16.

[16] Y. Sun, Y. Liu, G. Wang, and H. Zhang, "Deep learning for plant identification in natural environment," *Comput. Intell. Neurosci.*, vol. 2017, 2017.

[17] C. G. O. Lana, "Pengembangan Aplikasi Mobile Untuk Mengidentifikasi Spesies Tanaman Obat Menggunakan Metode Convolutional Neural Network," Universitas Atma Jaya Yogyakarta, 2020.

[18] S. Targ, D. Almeida, and K. Lyman, "Resnet in resnet: Generalizing residual architectures," *arXiv Prepr. arXiv1603.08029*, 2016.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv Prepr. arXiv1409.1556*, 2014.

[20] D. Sunaryono, J. Siswantoro, and R. Anggoro, "An android based course attendance system using face recognition," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 33, no. 3, pp. 304–312, 2021, doi: https://doi.org/10.1016/j.jksuci.2019.01.006.

[21] I. Gogul and V. S. Kumar, "Flower species recognition system using convolution neural networks and transfer learning," in *2017 fourth international conference on signal processing, communication and networking (ICSCN)*, 2017, pp. 1–6.

[22] T. Lu, B. Han, L. Chen, F. Yu, and C. Xue, "A generic intelligent tomato classification system for practical applications using DenseNet-201 with transfer learning," *Sci. Rep.*, vol. 11, no. 1, p. 15824, 2021, doi: 10.1038/s41598-021-95218-w.

[23] J. Siswantoro, "Application of Color and Size Measurement in Food Products Inspection," *Indones. J. Inf. Syst.*, vol. 1, no. 2, pp. 90–107, Feb. 2019, doi: 10.24002/ijis.v1i2.1923.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional

networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[26] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv Prepr. arXiv1704.04861*, 2017.

[27] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. Sebastopol, CA: O'Reilly Media, Inc., 2008.

[28] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.

[29] M. Abadi *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.