International Journal on Advanced Science Engineering Information Technology

Vol.12 (2022) No. 3
ISSN: 2088-5334

Convolutional Neural Networks for Herb Identification: Plain Background and Natural Environment

Supawadee Chaivivatrakul [a,*], Jednipat Moonrinta [b], Suchada Chaiwiwatrakul [c]

[a] *Agriculture Faculty, Ubon Ratchathani University, Warinchamrap, Ubon Ratchathani, 34190, Thailand*
[b] *Asian Institute of Technology, P.O. Box 4, Klong Luang, Pathumthani, 12120, Thailand*
[c] *Faculty of Humanities and Social Sciences, Ubon Ratchathani Rajabhat University, Ubon Ratchathani 34000, Thailand*
*Corresponding author: *supawadee.c@ubu.ac.th*

***Abstract*—Convolutional neural networks have achieved success in resolving object identification problems. This study contributes a suitable new approach to herb identification for educational and research purposes based on a small dataset and small-sized images. Two self-collected Thai herb datasets with either plain or natural environment backgrounds were used for experimentation to realize this objective. The plain background dataset includes 4,400 images of 11 leaf types, and the natural dataset contains 1,620 images of nine leaf types. The images were divided into a training set containing 75% of the images and a separate test set with the remaining 25%. The experiments included five-fold cross-validation applied to the training set; the InceptionV3, MobileNetV2, ResNet50V2, VGG16, and Xception convolutional neural network models RMSprop and Adam optimizers. Further, dropout rates of 0.3, 0.5, and 0.7 were considered along with five and ten epochs. Transfer learning was applied using pre-trained weights. The model with the best outcome, based on the average accuracy of the cross-validation results on both datasets (the plain background dataset was 94.55%, and the natural dataset was 90.37%), was the VGG16 with the RMSprop optimizer, which exhibited a dropout rate of 0.5 over ten epochs. The model achieved 96.64% and 92.00% accuracy on the plain background training and test sets, and 99.59% and 91.36% on the natural environment training and test sets, respectively. The results show that the method has a high potential for objective tasks and application in identifying herbs based on visual leaf information.**

***Keywords*—Deep learning; leaf identification; transfer learning.**

Manuscript received 25 May 2021; revised 11 Nov. 2021; accepted 12 Dec. 2021. Date of publication 30 Jun. 2022.

I. INTRODUCTION

Herb identification is crucial in medicine, botany, and the food industry. Typically, experts or experienced persons undertake this task. The plant parts that can be used to identify plant species are leaves, flowers, fruits, seeds, and roots. Because the leaves exist for most of the plant's life, they are suitable for herb identification. Several studies have proposed identifying plant species based on the visual information of plant parts. Yigit et al. [1] conducted experiments to identify plant leaves using an artificial neural network, the Naive Bayes algorithm, the random forest algorithm, K-nearest neighbor, and a support vector machine (SVM). The SVM returned the best accuracy of 92.91%. Soleimanipour et al. [2] proposed a flower identification method used the principal component analysis, latent Dirichlet allocation, and SVM, which achieved 99.5% accuracy. Ambarwari et al. [3] called SVM to classify plant species using leaf venation features and achieved an accuracy of 82.6% with a precision of 84% and recall of 83%. Furthermore, Chaki et al. [4] experimented on bag-of-features, fuzzy-color, and edge-texture histogram descriptors of a multi-layer perceptron for a fragmented leaf recognition problem, and Kala and Viriri [5] classified plant species based on the sinuosity coefficients of leaves. Moreover, Bao et al. [6] used a convolutional neural network (CNN) over histogram-of-oriented-gradients features with the SVM classifier on the Swedish and Flavia datasets. Muthireddy and Jawahar [7] researched Indian plant recognition in the wild using a convolutional neural network. Further, Mookdarsanit and Mookdarsanit [8] worked with Thai herb identification on a plain background.

The potential of deep learning based on a CNN has yielded high accuracy results in various problems, including plant research. Previous studies have investigated plant disease classification on citrus leaves [9], olive leaves [10], and apple leaves [11]. Moreover, weed detection experiments have been

conducted in the crop environment [12], [13], and nutrient deficiency classification has been performed with tomatoes [14]. The detection of fruit on the plant was explored as a multiclass problem [15] and was conducted with tomatoes [16] and small fruits [17].

Recently, TensorFlow [18]–[20], open-source software that provides a stable platform for CNN deep learning, has been used in various research studies and applications that include identifying plants [6], [21], [22] and plant diseases [23], [24]. Automatic herb identification is a helpful support system for botanical education and surveying. The studies mentioned above did not consider including Thai herbs in both the plain background and natural environment scenarios in a botanical education and surveying system. Because of the tasks' requirements, the classifications of herb leaves on a plain background and the plants in their natural environment were explored to gain confidence for further application development. The classification of Thai herb leaves was experimentally tested on various CNNs to select the most suitable method.

II. Materials and Method

Two datasets were collected, one of the leaves on a plain background and the other of leaves on plants in their natural environments. The proposed method included a deep convolutional neural network with convolutional neural network models, optimizer, dropout rate, and varying numbers of epochs. The experimental setups are given with specific details of the experiments.

A. Datasets

The plain background dataset included 11 leaf types: holy basil, betel, celery, asiatic pennywort, ivy gourd, kaffir lime, mint, water morning glory, mulberry, wild pepper, and *Tiliacora trianda* Diels (Fig.1). All leaf types in this dataset were classified as herbs and vegetables. During our data collection process, each leaf was pasted to a plain white background. A total of 100 leaves of each plant type were taken to obtain 100 images (1,100 images for all types). The dataset includes images with poor form leaves, immature leaves, damaged leaves, shadows, and insufficient image quality.

The dataset of the leaves in their natural environments includes nine leaf types: *Alixia reinwardtii* Blume, aloe, common rhinacanthus, frangipani, goat's foot creeper, *Muehlenbeckia platyclade* (F. Muell.) Meisn., mulberry, pagoda, and patchouli (Fig. 2). All leaf types in this dataset were classified as herbs, and some were further classified as vegetables. The dataset was collected from the Somdet Phrathep Rattana Ratchasuda Flora Herb Park in Rayong, Thailand. A total of 20 images were collected of every plant type in this dataset (180 images for all types). Because of the park environment and the nature of the plants, this dataset includes various photos: with one leaf, many leaves, old yellow leaves, damaged leaves, obstacles, an unclear camera view, and insufficient image quality. This dataset demonstrates the complexity of the plants surveying tasks in their natural environments.

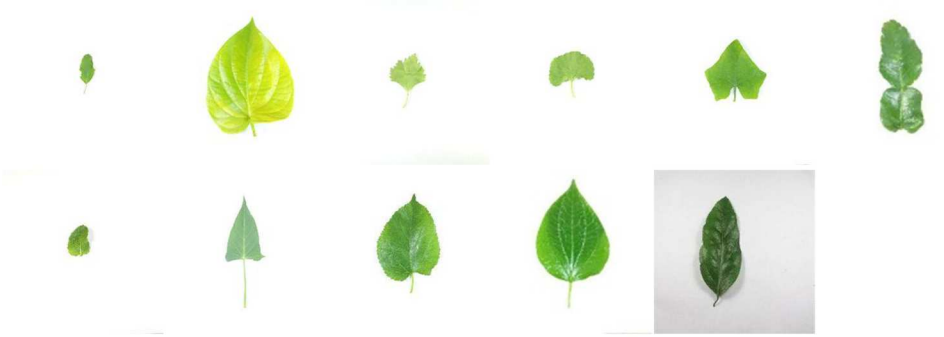


Fig. 1 Sample images from the plain background dataset. All 11 leaf types are shown, with original leaf images followed by augmented images.

Fig. 2 Sample images from the natural dataset. Nine leaf types are shown, with original leaf images followed by augmented images.

B. Method

The overall system is illustrated in Fig. 3. The training phase has two paths: the model and parameter search using five-fold cross-validation [25] and the adoption of the appropriate parameter set to train with the whole dataset. In the training phase, the images and their class labels are input, and pre-processing is conducted, which includes cropping, rescaling, and augmentation. A model is trained based on a base model, an optimizer, a dropout rate, and epochs, and a trained CNN model is received as an output.

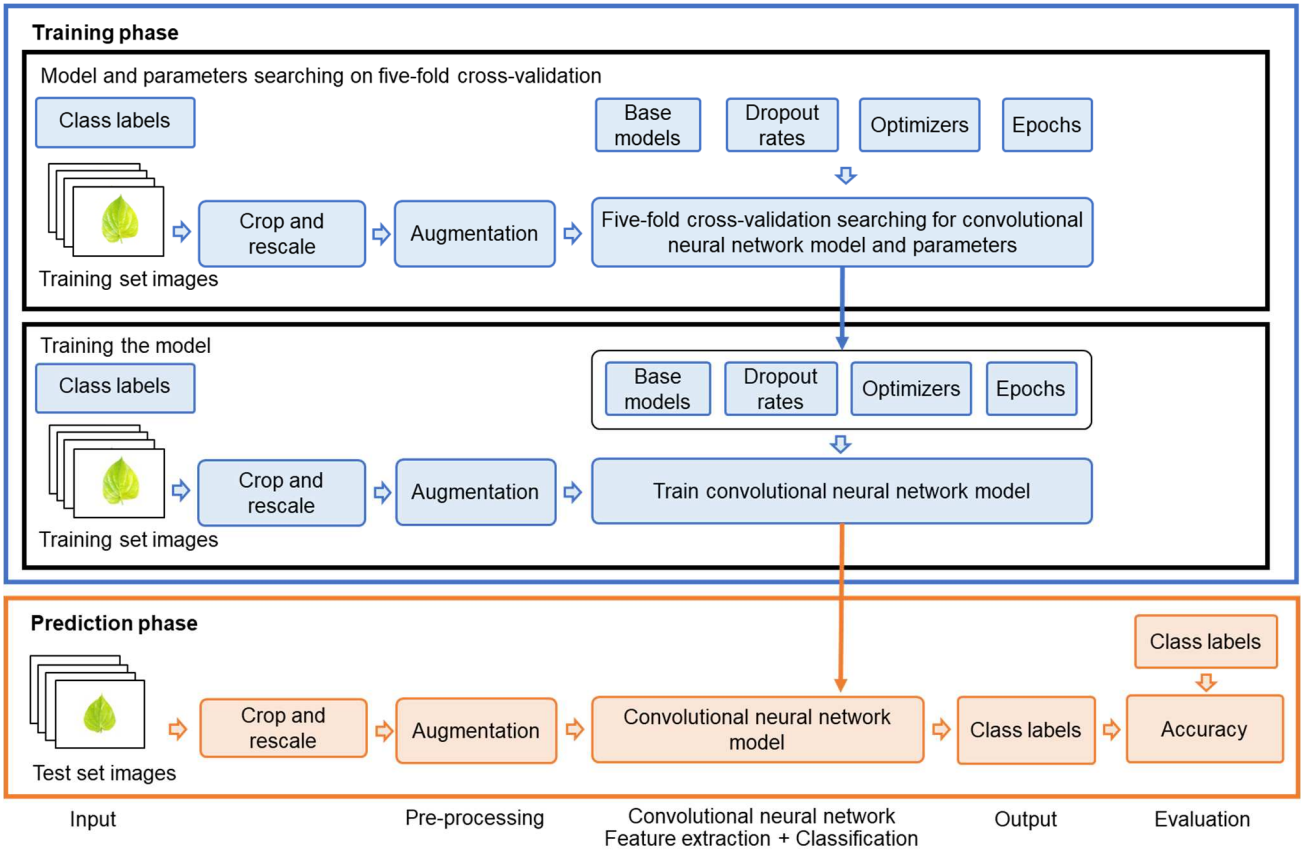


Fig. 3 System overview including training and prediction phases.

The test set images (without labels) are input in the trained model in the prediction phase. The input images are taken in the pre-processing step and classified, and finally, the output class labels are obtained. The output class labels are evaluated using the true labels, and the model's accuracy is determined.

The pre-processing steps follow either Algorithm I PREPROCESSING I or Algorithm II PRE-PROCESSING II.

The algorithm I PRE-PROCESSING I

Input: I an original image.
Input: W is the width of the output image.
Input: H is the height of the output image.
Input: C is the channel number of the output image.
Output: $\{O_i\}_{i \in 1 \ldots n}$ is a set of augmented images of I.

$I_c \leftarrow$ CROP-TO-SQURE(I)
▷ I_c is a cropped image of I.
$O_1 \leftarrow$ RESIZE(I_c,[W,H,C])
▷ O_1 is a resized image of I_c.
$O_2 \leftarrow$ HORIZONTAL-FLIP(O_1)
▷ O_2 is a horizontally flipped image of O_1.
$A_1 \leftarrow$ RANDOM(-90,90)
$O_3 \leftarrow$ ROTATE(O_1, A_1)
▷ A_1 is a rotation angle.
▷ O_3 is a rotated image *of* O_1.
$A_2 \leftarrow$ RANDOM(-90,90)
$O_4 \leftarrow$ ROTATE(O_2, A_2)
▷ A_2 is a rotation angle.
▷ O_4 is a rotated image of O_2.

Algorithm II PRE-PROCESSING II

Input: I an original image.
Input: W is the width of the output image.
Input: H is the height of the output image.
Input: C is the channel number of the output image.
Output: $\{O_i\}_{i \in 1 \ldots n}$ is a set of augmented images of I.

$I_c \leftarrow$ CROP-TO-SQURE(I)
▷ I_c is a cropped image of I.
$O_1 \leftarrow$ RESIZE(I_c,[W,H,C])
▷ O_1 is a resized image of I_c.
$O_2 \leftarrow$ HORIZONTAL-FLIP(O_1)
▷ O_2 is a horizontally flipped image of O_1.
$O_3 \leftarrow$ VERTICAL-FLIP(O_1)
▷ O_3 is a vertically flipped image of O_1.
$A_1 \leftarrow$ RANDOM(-90,0)
$O_4 \leftarrow$ ROTATE(O_1, A_1)
$A_2 \leftarrow$ RANDOM(0,90)
$O_5 \leftarrow$ ROTATE(O_1, A_2)
▷ A_1, A_2 are rotation angles.
▷ O_4, O_5 are rotated images of O_1.
$A_3 \leftarrow$ RANDOM(-90,0)
$O_6 \leftarrow$ ROTATE(O_2, A_3)
$A_4 \leftarrow$ RANDOM(0,90)
$O_7 \leftarrow$ ROTATE(O_2, A_4)
▷ A_3, A_4 are rotation angles.
▷ O_6, O_7 are rotated images of O_2.
$A_5 \leftarrow$ RANDOM(-90,0)
$O_8 \leftarrow$ ROTATE(O_3, A_5)
$A_6 \leftarrow$ RANDOM(0,90)
$O_9 \leftarrow$ ROTATE(O_3, A_6)
▷ A_5, A_6 are rotation angles.
▷ O_8, O_9 are rotated images of O_3.

Five experimental models were proposed for use on the datasets. The general architecture of a CNN for image classification includes feature extraction: a convolutional kernel with subsampling and classification as depicted in Fig. 4. Each model differs in its details, such as the number of layers, convolution, and the subsampling method. InceptionV3 [26], [27] is a model obtained from a series of deep-learning convolutional architectures. MobileNetV2 [28], [29] is a model proposed for mobile devices based on an inverted residual structure, which uses an intermediate expansion layer and lightweight depthwise convolutions. ResNet50V2 [27], [30] is a modified version of ResNet50 (50 weight layers) with changes in how the propagation connections between blocks are formulated. VGG16 [27], [30], [31] is a weighted 16-layer model. Xception [32] involves depthwise separable convolutions, its architecture being inspired by the Inception model.

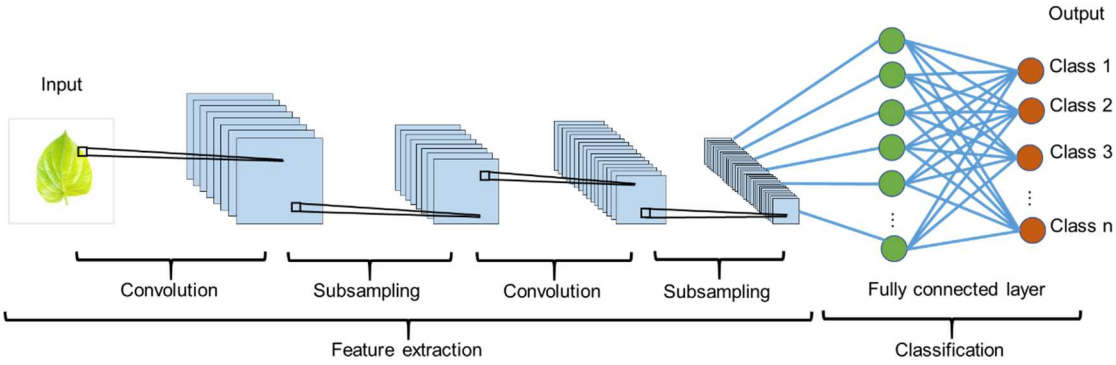


Fig. 4 General architecture of CNN for image classification.

An optimizer changes the attributes of a network to reduce loss. The Root Mean Square Propagation (RMSprop) optimizer accumulates the gradients in a fixed window instead of gathering all gradients [33], [34]. The RMSprop is calculated as follows:

$$v_t^{(j)} = \rho v_{t-1}^{(j)} + (1-\rho)(g_t^{(j)})^2 \quad (1)$$

$$w_{t+1}^{(j)} = w_t^{(j)} - \frac{\alpha}{\sqrt{v_t^{(j)}+\varepsilon}} g_t^{(j)} \quad (2)$$

where $v_t^{(j)}$ is the exponential average of the gradients' squares, α is an initial learning rate, $g_t^{(j)}$ is the gradient at step t, $w_t^{(j)}$, ρ is a hyperparameter, t is the step in the stochastic gradient descent algorithm, j indicates the associated component weight, and $w_t^{(j)}$ is the weight of step t and component j.

Adaptive Moment Estimation (Adam) utilizes the concept of momentum fractions of previous gradients added to the current one [34]–[36]. The details are as follows:

$$m_t^{(j)} = \beta_1 m_{t-1}^{(j)} + (1-\beta_1) g_t^{(j)} \quad (3)$$

$$v_t^{(j)} = \beta_2 v_{t-1}^{(j)} + (1-\beta_2)(g_t^{(j)})^2 \quad (4)$$

$$\hat{m}_t^{(j)} = \frac{m_t^{(j)}}{1-\beta_1^t} \quad (5)$$

$$\hat{v}_t^{(j)} = \frac{v_t^{(j)}}{1-\beta_2^t} \quad (6)$$

$$w_{t+1}^{(j)} = w_t^{(j)} - \frac{\alpha}{\sqrt{\hat{v}_t^{(j)}+\varepsilon}} \hat{m}_t^{(j)} \quad (7)$$

where β_1 and β_2 are parameters, $v_t^{(j)}$ is an exponential average of the squares of gradients, ε is the regularization term, α is an initial learning rate, $g_t^{(j)}$ is the gradient at step t along $w_t^{(j)}$, ρ is a hyperparameter, t is the step in the stochastic gradient descent algorithm, j indicates the associated component weight, and $w_t^{(j)}$ is the weight of step t and component j.

The pre-trained weight is an effective alternative method to initiate the weight of a CNN. This approach is suited for small datasets and meets the time-saving criterion for training. The pre-trained weight on the ImageNet dataset [37] was adopted for our method. This pre-trained weight was acquired from the training process with a large-scale dataset consisting of 14 million images with 1,000 classes for multipurpose use.

Algorithm III FIT-MODEL

Input: T is a training set.
Input: V is a validation set.
Input: B is a based model.
Input: W is the initial weight.
Input: P is an optimizer option.
Input: D is the dropout rate.
Input: E is the number of epochs.
Input: N is the number of classes.
Input: H is the batch size of the image.
Input: S is image size.
Output: M is an output model.

Load T
$T \leftarrow$ SHUFFLE(T)
$T' \leftarrow$ NORMALIZED(T)
▷ Normalized RGB value of T [0,255] to [0,1]
$B' \leftarrow$ SET-BASED-MODEL(B,S,W,)
▷ B' is based model with setting
$M' \leftarrow$ MODEL-SEQUENTIAL(B',D,N)
$M'' \leftarrow$ MODEL-COMPILE(M',P)
▷ Compile the model with
▷ loss='sparse_categorical_crossentropy',
▷ and metrics=['accuracy']
$M \leftarrow$ MODEL-FIT(M'',T',E,V)

The CNN models were trained to follow the steps in the Algorithm III FIT-MODEL. The algorithm takes a base model, an optimizer option, a dropout rate (dropout), epochs, the training set, the validation set, the number of classes, the

batch size, and the image size as inputs and returns a trained output model. The training set was shuffled, and the RGB value was normalized from 0–255 to 0–1. The image batch size was set to 32; the image size was set to 128 × 128 × 3. The training process took the training set as input and returned the output model. The output model was then evaluated (Algorithm IV EVALUATE), and the training and test sets' results were reported separately.

Algorithm IV EVALUATE

Input: *M* is a model.
Input: *Z* is a test set without labeling.
Input: *ZL* is a test set label.
Output: *C* is the accuracy.
Output: *L* is the precision.
Output: *R* is the recall.
Output: *F* is the F1-score.

1: Load *Z*
2: *Z′* ← NORMALIZED(*Z*)
3: ▷ Normalized RGB value of *Z* [0,255] to [0,1]
4: *ZL′* ← MODEL-PREDICT(*M*, *Z′*)
5: ▷ *ZL′* is the predicted label of *Z*
6: *C,L,R,F* ← MODEL-EVALUATE(*ZL*,*ZL′*)

The methods were evaluated by machine learning criteria [25], [38]. Accuracy (8) is the percentage of correct events overall. Precision (9) is the percentage of correct events overall predicted positively. Recall (10) is the percentage of correct events among all positive events. The F1-score (11) is the harmonic mean of precision and recall. Let TP be the number of true-positive events, TN the number of true-negative events, FN the number of false-negative events, and FP the number of false-positive events. However, TN is not considered and is set to zero in this research.

$$Accuracy = \frac{TP}{TP+FN+TN+FP} \quad (8)$$

$$Precision = \frac{TP}{TP+FP} \quad (9)$$

$$Recall = \frac{TP}{TP+FN} \quad (10)$$

$$F1-score = \frac{2Precision \cdot Recall}{Precision+Recall} \quad (11)$$

C. Experimental Setup

In pre-processing the plain background dataset, a white border of each image was cropped to obtain a square image with the original aspect ratio of the leaf unchanged. The images were resized to 128 × 128 × 3. A horizontal flip and a rotation between −90° and 90° were randomly applied to obtain a larger dataset. Algorithm I PRE-PROCESSING I gives detail of the steps. The augmented dataset contained a total of 4,400 images.

In pre-processing of the natural dataset, the border of each image was cropped to obtain a square image with the original aspect ratio of the leaf unchanged. The image was then resized to 128 × 128 × 3. Steps of a horizontal flip, a vertical flip, a random rotation of between −90° and 0°, and a random rotation of 0°–90° were applied. The obtained dataset contained 1,620 images. The steps follow Algorithm II PRE-PROCESSING II.

Each dataset was divided into a training set with 75% of the images and a test set with 25%. Original and augmented image proportions are shown in TABLE I, and the samples of the original images with their augmented images are depicted in Fig. 5.

TABLE I
NUMBERS OF IMAGES IN THE DATASETS, SHOWING THE PROPORTION OF ORIGINAL AND AUGMENTED IMAGES IN THE TRAINING AND TEST SETS

Dataset Type	Total images			Original images			Augmented images		
	Total	Train	Test	Total	Train	Test	Total	Train	Test
	100%	75%	25%	100%	75%	25%	100%	75%	25%
Plain background									
All 11 types	4,400	3,300	1,100	1,100	825	275	3,300	2,475	825
Each type	400	300	100	100	75	25	300	225	75
Natural									
All 9 types	1,620	1,215	405	180	135	45	1,440	1,080	360
Each type	180	135	45	20	15	5	160	120	40

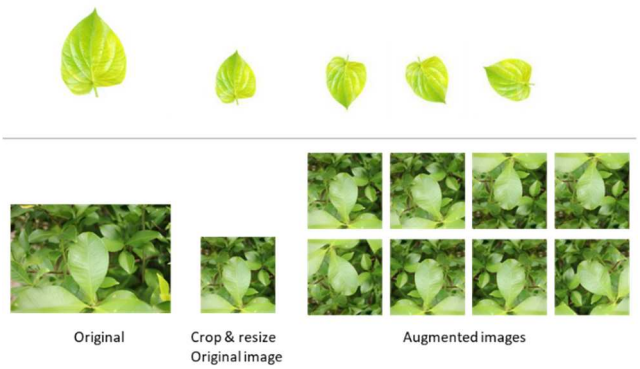


Fig. 5 Samples of pre-processing step

The CNN model fitting was provided by calling the steps of the algorithm in Algorithm III FIT-MODEL and evaluated the result follows Algorithm IV EVALUATE. The experiment was configured to work with CNN implemented on TensorFlow (Core v2.4.1) [18] using Python in a Google Colab environment (GPU Jupyter notebook) [39]. The experiment was performed on the default parameter set of five CNN models: InceptionV3, MobileNetV2, ResNet50V2, VGG16, and Xception. Transfer learning was applied using pre-trained weights on ImageNet. Each model was evaluated using either of the two optimizers, namely RMSprop and Adam, with a dropout rate of 0.3, 0.5, and 0.7, and epoch numbers of 5 and 10.

III. Results and Discussion

The five-fold cross-validation results are shown in Fig. 6 and Fig. 7. The best model on both datasets was VGG16 with transfer learning, pre-trained weight ImageNet, optimizer RMSprop, dropout 0.5, and 10 epochs. The average accuracy of the plain background dataset was 94.55%, and that of the natural dataset was 90.37%.

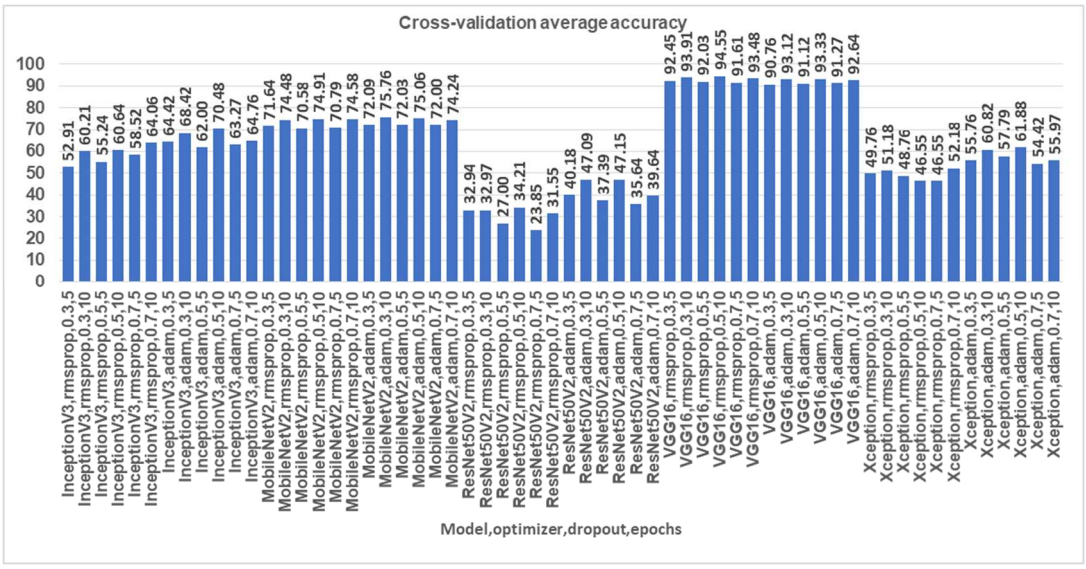


Fig. 6 Average accuracy results of the five-fold cross-validation of all combinations with the plain background dataset.

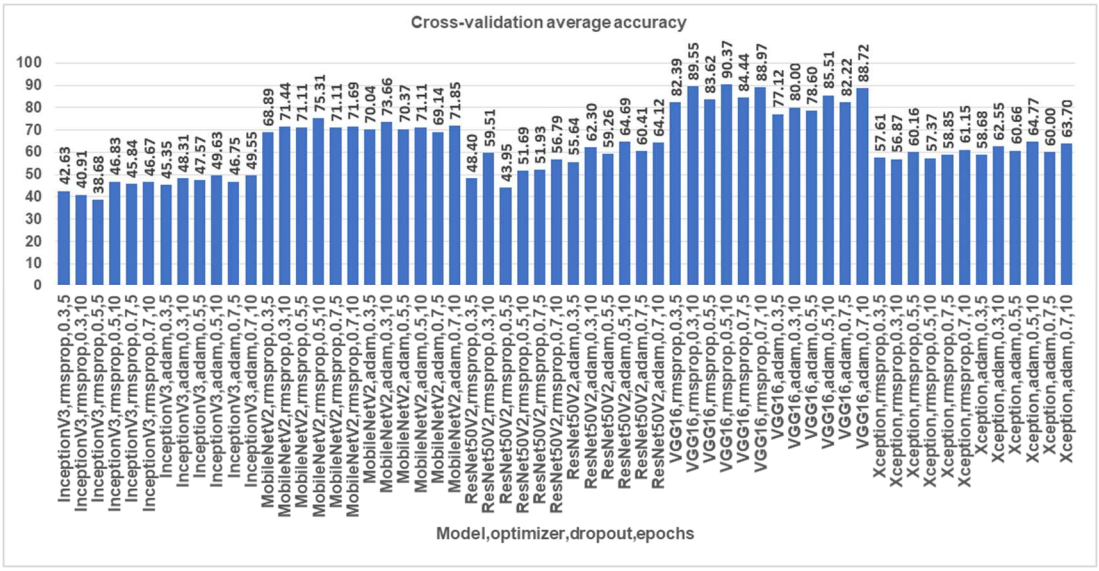


Fig. 7 Average accuracy results of the five-fold cross-validation of all combinations with the natural dataset.

The best model was adopted to apply to the entire training set and evaluate the output model. The evaluation was performed with the entire training and test sets. The accuracy, precision, recall, and F1-score of all leaf classes on the training and test sets were calculated and reported in table II. The true negative case was not considered (set to zero).

TABLE II
RESULTS OF THE BEST MODEL ON THE TRAINING AND TEST SETS.

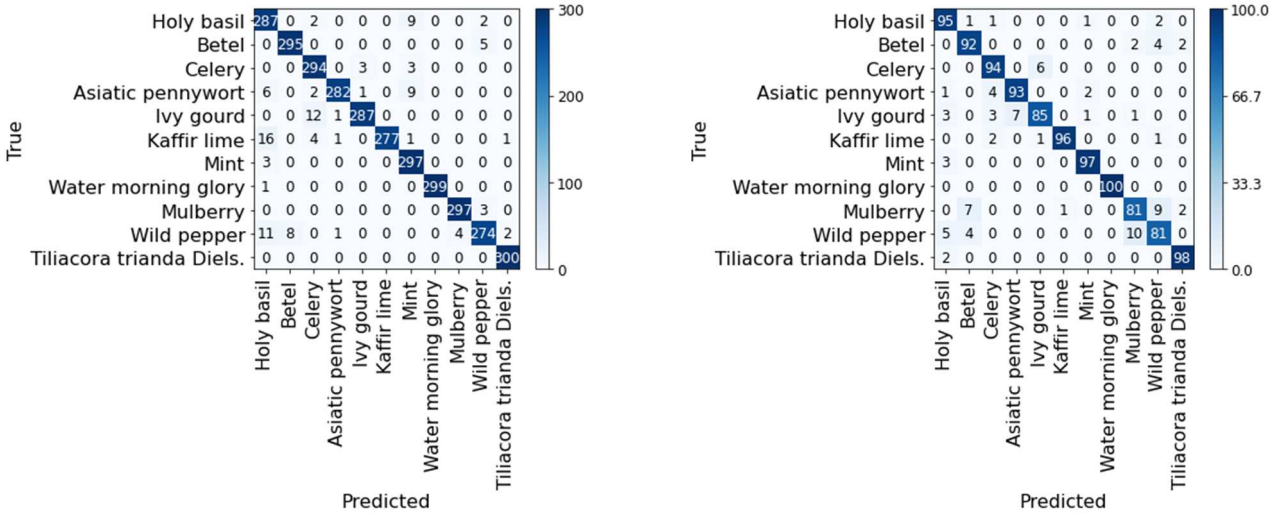
Dataset Model	Result	Training set	Test set
Plain Background VGG16 RMSprop Dropout 0.5 Epochs 10	Accuracy	0.9664	0.9200
	Precision	0.9676	0.9201
	Recall	0.9664	0.9200
	F1-score	0.9665	0.9196
Natural VGG16 RMSprop Dropout 0.5 Epochs 10	Accuracy	0.9959	0.9136
	Precision	0.9960	0.9222
	Recall	0.9959	0.9136
	F1-score	0.9959	0.9142

The confusion matrices of the best model on the training and test sets are depicted in Fig. 8–Fig. 9. In Fig. 8, the confusion matrices of the plain background dataset show the classification of water morning glory and *Tiliacora trianda Diels.* achieved high accuracy (98%–100%) on the training and test sets because the leaf shapes of these two plant types are distinct from the others. Moreover, the images of these groups had a small number of poor forms leaves and no immature or damaged leaves. In the test set, the classification of mulberry and wild pepper achieved 81% (81 of 100), and that of ivy gourd reached 85% (85 of 100) because of their similar shape to others, especially when using a small-size image. The model achieved an accuracy of over 90% with the remaining leaf types in this dataset.

In Fig. 9, the confusion matrix of the natural dataset shows that the model achieved 100% accuracy in the prediction of aloe and Muehlenbeckia platyclade (F. Muell.) Meisn. in the training and test sets, the leaf shape of this plant type is distinct from that of the others. In the test set, the model

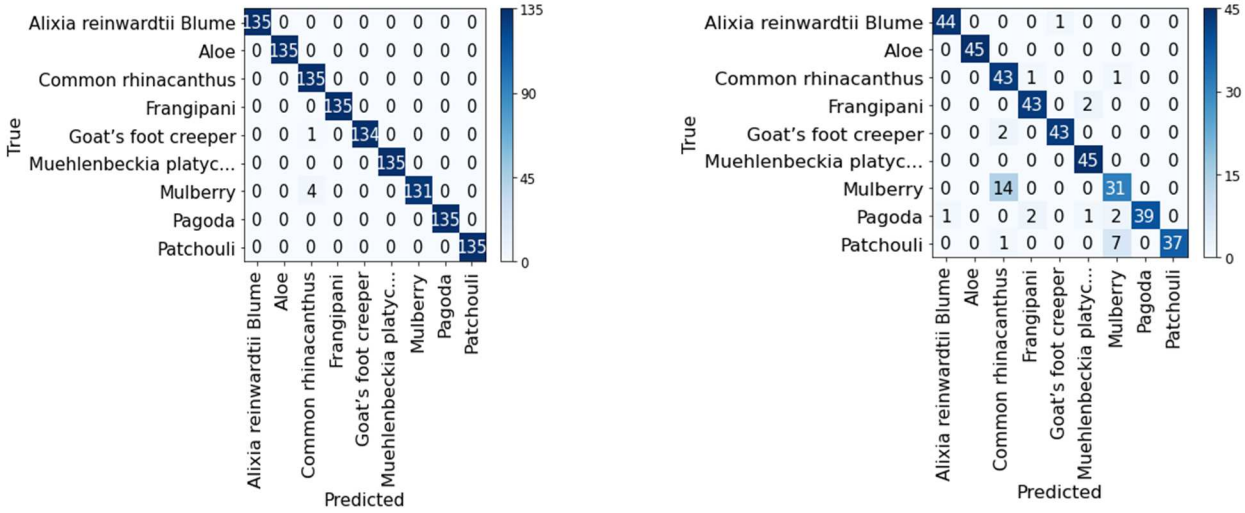
achieved 68.89% (31 of 45) with mulberry images, 82.22% (37 of 45) with patchouli, and 86.67% (39 of 45) with pagoda plant.

The unstable light conditions of the outdoor environment, the camera view, and many leaves in the same image made the images look very similar, and the shapes were not clearly defined. The model achieved an accuracy of over 90% with the remaining plant types in this dataset. Fig. 10 and Fig. 11 show the incorrect prediction samples from the respective datasets with the true and predicted classes.



(a) (b)

Fig. 8 Confusion matrix of the plain-background dataset. (a) Confusion matrix for the training set. (b) Confusion matrix for the test set.



(a) (b)

Fig. 9 Confusion matrix of the natural dataset. (a) Confusion matrix for the training set. (b) Confusion matrix for the test set.

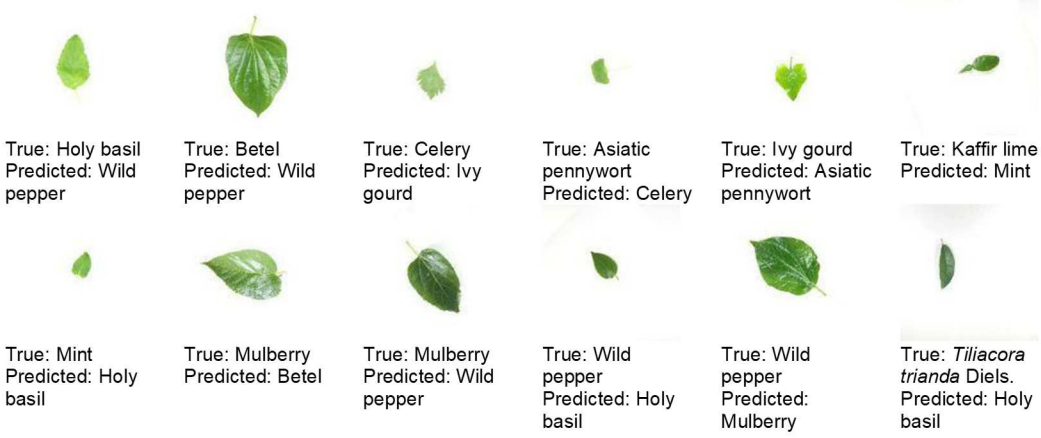


Fig. 10 Incorrect prediction samples from the plain background dataset.

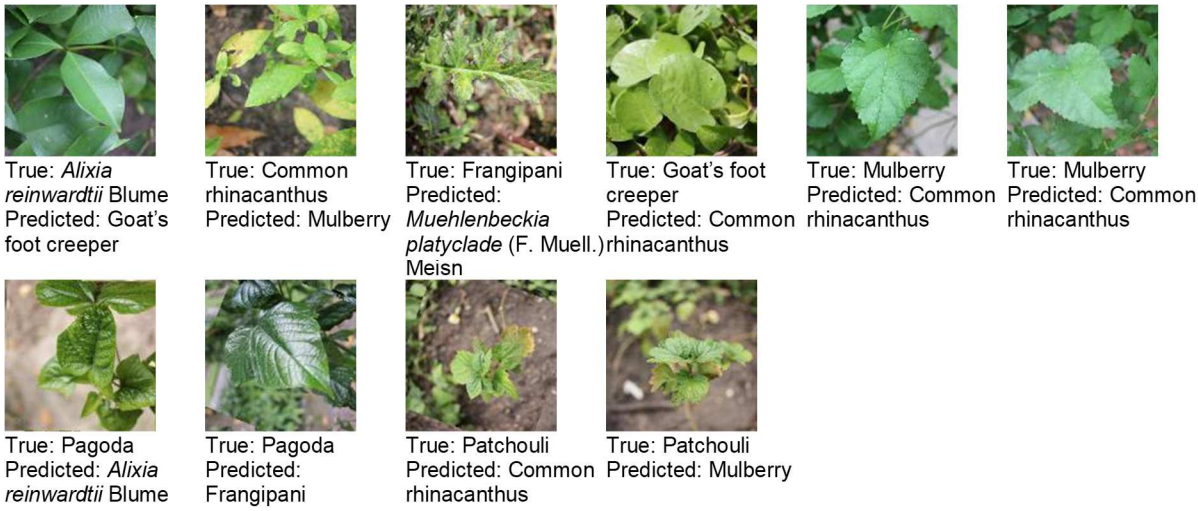


Fig. 11 Incorrect prediction samples from the natural dataset.

IV. Conclusion

Two datasets were assembled to identify Thai herbs using visual leaf information. The examined datasets included 11 leaf types in the plain background dataset and nine leaf types in the natural dataset. Augmentation was applied to obtain a larger dataset for the CNN. VGG16 with transfer learning (pre-trained weight ImageNet) yielded the highest accuracy. The model achieved an accuracy of 96.64% and 92.00% for the training and test sets of the plain background dataset, respectively, and 99.59% and 91.36% for the natural dataset training and test sets, respectively.

CNN requires a large dataset to obtain a good model for the training process. However, our datasets were small, although augmentation was applied. One alternative option for using a small dataset is transfer learning with a pre-trained weight. Furthermore, this approach has the advantage of saving training time. Thus, transfer learning was adopted with a pre-trained weight (ImageNet) for this research.

Incorrect predictions with the plain background dataset were obtained in cases of similar leaf shapes, poor form leaves, damaged leaves, immature leaves, varied light conditions, strong shadows, and insufficient image quality. Incorrect predictions with the natural dataset were old yellow leaves, unclear background, varied light conditions, unclear camera view, and inadequate image quality. Avoiding all of these issues when constructing the dataset would yield better results.

The small size of the image was used to obtain benefits concerning data loading and algorithm running time. The overall accuracy was sufficiently high to show that the small size did not negatively affect the method. Thus, the high accuracy of small image size indicates that the technique has a high potential for applications in limited systems such as mobile-device and real-time systems. An approach to obtaining higher accuracy with the same method and image size would be to use multiple identifications and a suitable image view.

The potential of the CNN model could be confirmed by comparison with previous methods. For identifying Thai herbs from images on a plain background, the best-investigated model achieved a recall and precision of over 90% compared to the model by Mookdarsanit and Mookdarsanit [8] that achieved a recall of over 75% and a precision of over 80%. Compared with other plant identification datasets obtained in the natural environment, our results were better than those of Muthireddy and Jawahar [7], with the image width and height we used (128 × 128 pixels) being almost half (224 × 224 pixels) of theirs. The smaller image size confers better management of resources such as running time and storage. Thus, the proposed CNN-model approach and the input image size are more suitable for both datasets.

More leaf types and a larger dataset will be investigated in future work to obtain a wider perspective of the problem and a more reliable trained model for leaf identification, botanical education, and surveying. This requires a larger dataset and a practical means of capturing images of leaves on the plants by managing the process in the outdoor environment. Moreover, this system will be developed further in a future project.

Acknowledgment

The author thanks the Coordinating Center for Thai Government Science and Technology Scholarship Students (CSTS) and the National Science and Technology Development Agency (NSTDA) for their financial support. The author is grateful to the Somdet Phrathep Rattana Ratchasuda Flora Herb Park in Rayong, Thailand, for the data collection and the agricultural community's support.

References

[1] E. Yigit, K. Sabanci, A. Toktas, and A. Kayabasi, "A study on visual features of leaves in plant identification using artificial intelligence techniques," *Comput. Electron. Agric.*, vol. 156, no. June 2018, pp. 369–377, 2019, doi: 10.1016/j.compag.2018.11.036.

[2] A. Soleimanipour, G. R. Chegini, and J. Massah, "Classification of anthurium flowers using combination of PCA, LDA and support vector machine," *Agric. Eng. Int. CIGR J.*, vol. 20, no. 1, pp. 219–228, 2018.

[3] A. Ambarwari, Q. J. Adrian, Y. Herdiyeni, and I. Hermadi, "Plant species identification based on leaf venation features using SVM," *TELKOMNIKA (Telecommunication Comput. Electron. Control.*, vol. 18, no. 2, p. 726, Apr. 2020, doi: 10.12928/telkomnika.v18i2.14062.

[4] J. Chaki, N. Dey, L. Moraru, and F. Shi, "Fragmented plant leaf recognition: Bag-of-features, fuzzy-color and edge-texture histogram descriptors with multi-layer perceptron," *Optik (Stuttg).*, vol. 181, no.

December 2018, pp. 639–650, Mar. 2019, doi: 10.1016/j.ijleo.2018.12.107.

[5] J. R. Kala and S. Viriri, "Plant Specie Classification Using Sinuosity Coefficients of Leaves," *Image Anal. Stereol.*, vol. 37, no. 2, p. 119, Jul. 2018, doi: 10.5566/ias.1821.

[6] T. Q. Bao, N. T. T. Kiet, T. Q. Dinh, and H. X. Hiep, "Plant species identification from leaf patterns using histogram of oriented gradients feature space and convolution neural networks," *J. Inf. Telecommun.*, vol. 4, no. 2, pp. 140–150, Apr. 2020, doi: 10.1080/24751839.2019.1666625.

[7] V. Muthireddy and C. V. Jawahar, "Indian Plant Recognition in the Wild," in *Computer Vision, Pattern Recognition, Image Processing, and Graphics*, N. V. P. Babu R.V., Prasanna M., Ed. Springer, Singapore, 2020, pp. 439–449.

[8] L. Mookdarsanit and P. Mookdarsanit, "Thai Herb Identification with Medicinal Properties Using Convolutional Neural Network," *Suan Sunandha Sci. Technol. J.*, vol. 06, no. 2, pp. 34–40, 2019, doi: 10.14456/ssstj.2019.8.

[9] U. Barman, R. D. Choudhury, D. Sahu, and G. G. Barman, "Comparison of convolution neural networks for smartphone image based real time classification of citrus leaf disease," *Comput. Electron. Agric.*, vol. 177, no. July, p. 105661, Oct. 2020, doi: 10.1016/j.compag.2020.105661.

[10] S. Uğuz and N. Uysal, "Classification of olive leaf diseases using deep convolutional neural networks," *Neural Comput. Appl.*, vol. 33, no. 9, pp. 4133–4149, May 2021, doi: 10.1007/s00521-020-05235-5.

[11] C. Bi, J. Wang, Y. Duan, B. Fu, J.-R. Kang, and Y. Shi, "MobileNet Based Apple Leaf Diseases Identification," *Mob. Networks Appl.*, Aug. 2020, doi: 10.1007/s11036-020-01640-1.

[12] P. Lottes, J. Behley, A. Milioto, and C. Stachniss, "Fully Convolutional Networks With Sequential Information for Robust Crop and Weed Detection in Precision Farming," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 2870–2877, Oct. 2018, doi: 10.1109/LRA.2018.2846289.

[13] H. Jiang, C. Zhang, Y. Qiao, Z. Zhang, W. Zhang, and C. Song, "CNN feature based graph convolutional network for weed and crop recognition in smart farming," *Comput. Electron. Agric.*, vol. 174, no. April, p. 105450, Jul. 2020, doi: 10.1016/j.compag.2020.105450.

[14] T.-T. Tran, J.-W. Choi, T.-T. Le, and J.-W. Kim, "A Comparative Study of Deep CNN in Forecasting and Classifying the Macronutrient Deficiencies on Development of Tomato Plant," *Appl. Sci.*, vol. 9, no. 8, p. 1601, Apr. 2019, doi: 10.3390/app9081601.

[15] H. Kuang, C. Liu, L. L. H. Chan, and H. Yan, "Multi-class fruit detection based on image region selection and improved object proposals," *Neurocomputing*, vol. 283, pp. 241–255, 2018, doi: https://doi.org/10.1016/j.neucom.2017.12.057.

[16] C. Hu, X. Liu, Z. Pan, and P. Li, "Automatic Detection of Single Ripe Tomato on Plant Combining Faster R-CNN and Intuitionistic Fuzzy Set," *IEEE Access*, vol. 7, pp. 154683–154696, 2019, doi: 10.1109/ACCESS.2019.2949343.

[17] X. Mai, H. Zhang, X. Jia, and M. Q. H. Meng, "Faster R-CNN With Classifier Fusion for Automatic Detection of Small Fruits," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1–15, 2020, doi: 10.1109/TASE.2020.2964289.

[18] M. Abadi *et al.*, "TensorFlow: A System for Large-Scale Machine Learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, Nov. 2016, pp. 265–283, [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.

[19] K. Wongsuphasawat *et al.*, "Visualizing Dataflow Graphs of Deep Learning Models in TensorFlow," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 1, pp. 1–12, Jan. 2018, doi: 10.1109/TVCG.2017.2744878.

[20] J. Persano, S. M. Mikki, and Y. M. M. Antar, "Gradient Population Optimization: A Tensorflow-Based Heterogeneous Non-Von-Neumann Paradigm for Large-Scale Search," *IEEE Access*, vol. 6, pp. 77097–77122, 2018, doi: 10.1109/ACCESS.2018.2868236.

[21] X. Liu, F. Xu, Y. Sun, H. Zhang, and Z. Chen, "Convolutional Recurrent Neural Networks for Observation-Centered Plant Identification," *J. Electr. Comput. Eng.*, vol. 2018, pp. 1–7, 2018, doi: 10.1155/2018/9373210.

[22] T. Boston and A. Van Dijk, "Some experiments in automated identification of Australian plants using convolutional neural networks," in *MODSIM2019, 23rd International Congress on Modelling and Simulation.*, Dec. 2019, no. October, pp. 15–21, doi: 10.36334/modsim.2019.A1.boston.

[23] M. Chohan, R. Adil Khan, S. H. K. Chohan, and M. S. Mahar, "Plant Disease Detection using Deep Learning," *Int. J. Recent Technol. Eng.*, vol. 9, no. 1, pp. 909–914, May 2020, doi: 10.35940/ijrte.A2139.059120.

[24] D. Argüeso *et al.*, "Few-Shot Learning approach for plant disease classification using images taken in the field," *Comput. Electron. Agric.*, vol. 175, no. June, p. 105542, Aug. 2020, doi: 10.1016/j.compag.2020.105542.

[25] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning*. Cham: Springer International Publishing, 2019.

[26] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, vol. 31, no. 1.

[27] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, May 2018, doi: 10.1016/j.patcog.2017.10.013.

[28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 4510–4520, doi: 10.1109/CVPR.2018.00474.

[29] B. Singh, D. Toshniwal, and S. K. Allur, "Shunt connection: An intelligent skipping of contiguous blocks for optimizing MobileNet-V2," *Neural Networks*, vol. 118, pp. 192–203, Oct. 2019, doi: 10.1016/j.neunet.2019.06.006.

[30] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.

[31] A. Brock, J. Donahue, and K. Simonyan, "Large Scale GAN Training for High Fidelity Natural Image Synthesis," *7th Int. Conf. Learn. Represent. ICLR 2019*, pp. 1–35, Sep. 2018, [Online]. Available: http://arxiv.org/abs/1809.11096.

[32] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, vol. 2017-Janua, pp. 1800–1807, doi: 10.1109/CVPR.2017.195.

[33] A. K. A. S. P. M. and P. R. Vinod Kumar, "Design and implementation of web-based expert tool for selection of climate resilient rapeseed-mustard varieties," *J. Oilseed Brassica*, vol. 0, no. 0, pp. 168–175, 2018.

[34] V. H. Nhu *et al.*, "Effectiveness assessment of Keras based deep learning with different robust optimization algorithms for shallow landslide susceptibility mapping at tropical area," *Catena*, vol. 188, no. November 2019, p. 104458, 2020, doi: 10.1016/j.catena.2020.104458.

[35] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, "A Sufficient Condition for Convergences of Adam and RMSProp," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019, vol. 2019-June, no. 1, pp. 11119–11127, doi: 10.1109/CVPR.2019.01138.

[36] A. Sharma, "Guided Stochastic Gradient Descent Algorithm for inconsistent datasets," *Appl. Soft Comput. J.*, vol. 73, pp. 1068–1080, 2018, doi: 10.1016/j.asoc.2018.09.038.

[37] K. He, R. Girshick, and P. Dollar, "Rethinking ImageNet Pre-Training," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, vol. 2019-Octob, no. ii, pp. 4917–4926, doi: 10.1109/ICCV.2019.00502.

[38] A. Berger and S. Guda, "Threshold optimization for F measure of macro-averaged precision and recall," *Pattern Recognit.*, vol. 102, p. 107250, Jun. 2020, doi: 10.1016/j.patcog.2020.107250.

[39] E. Bisong, "Google Colaboratory," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Berkeley, CA: Apress, 2019, pp. 59–64.