Transfer Learning for Caladium bicolor Classification: Proof of Concept to Application Development

Porawat Visutsak[1*], Xiabi Liu[2], Keun Ho Ryu[3], Naphat Bussabong[1], Nicha Sirikong[1], Preeyaphorn Intamong[1], Warakorn Sonnui[1], Siriwan Boonkerd[1], Jirawat Thongpiem[1], Maythar Poonpanit[1], Akarasate Homwiseswongsa[1], Kittipot Hirunwannapong[1], Chaimongkol Suksomsong[1], and Rittikait Budrit[1]

[1] Department of Computer and Information Science, Faculty of Applied Science, KMUTNB, Bangkok 10800 Thailand
[e-mail: {porawatv; s6304062856022; s6304062857011; s6304062857037; s6304062857045; s6304062857053; s6304062857061; s6304062857070; s6304062857088; s6304062857096; s6304062857100; s6304062857118}@kmutnb.ac.th]

[2] School of Computer Science and Technology, Beijing Institute Technology, Beijing 100081 China
[e-mail: liuxiabi@bit.edu.cn]

[3] Database/Bioinformatics Laboratory Chungbuk National University, Chungbuk, Republic of Korea
[e-mail: khryu@chungbuk.ac.kr]

[*]Corresponding author: Porawat Visutsak

Received April 13, 2023; revised August 7, 2023; revised September 19, 2023; accepted December 25, 2023; published January 31, 2024

Abstract

Caladium bicolor is one of the most popular plants in Thailand. The original species of Caladium bicolor was found a hundred years ago. Until now, there are more than 500 species through multiplication. The classification of Caladium bicolor can be done by using its color and shape. This study aims to develop a model to classify Caladium bicolor using a transfer learning technique. This work also presents a proof of concept, GUI design, and web application deployment using the user-design-center method. We also evaluated the performance of the following pre-trained models in this work, and the results are as follow: 87.29% for AlexNet, 90.68% for GoogleNet, 93.59% for XceptionNet, 93.22% for MobileNetV2, 89.83% for RestNet18, 88.98% for RestNet50, 97.46% for RestNet101, and 94.92% for InceptionResNetV2. This work was implemented using MATLAB R2023a.

Keywords: Classification, Feature extraction, Caladium bicolor, Software development, UX/UI concept.

This research was funded by the Faculty of Applied Science, King Mongkut's University of Technology North Bangkok, Thailand (Contract No. 662145). This research partly used computational resources from the China Scholarship Council (CSC) for the Senior Visiting Scholarship Program provided by the School of Computer Science and Technology, Beijing Institute Technology (BIT). We also express our thanks to Prof. Minghao Piao (Database/Bioinformatics Laboratory Chungbuk National University) who gave the fruitful advice to our paper.

1. Introduction

During the pandemic of COVID-19, people have been disrupted their lives by working from home and living with the limitation of day-to-day life. This disruption has impacted many populations across the world including those in Thailand. The restriction of social practice has led to national mental crisis. To reduce stress and mental health problems, families spend more time together for gardening and planting. Caladium bicolor is one of the most popular indoor plants in Thailand. Actually, the original species of Caladium bicolor was found a hundred years ago. In the present day, there are more than 500 species through multiplication, with various leaf shapes and colors [1], [2], [3], and [4]. Buying and tending Caladium bicolor is now not finding joy, but it has already become a big business in Thailand. The prices of some Caladium bicolor are triple because of its high demand, and it is rarely found in some species.

In this study, we proposed a transfer learning technique to classify Caladium bicolor by using the shape features of Caladium bicolor. We gathered images of Caladium bicolor from Google images and used an augmentation technique to increase our image dataset. The efficient 8 well-known pre-trained models were chosen for evaluation in our work (AlexNet, GoogleNet, XceptionNet, MobileNetV2, RestNet18, RestNet50, RestNet101, and InceptionResNetV2). This work was implemented using MATLAB R2023a. After gaining the experimental results (proof of concept), we selected the best candidate for these pre-trained models to be embedded in the application of Caladium bicolor classification. We designed a GUI for this application using the UX/UI concept. To evaluate which design is the most appropriate for use in the classification of Caladium bicolor in practice, we designed two variants of the GUI to test during the user-testing process. This paper is organized as follows: Section 2 describes the problem statement and literature review. The implementation is shown in Section 3. Section 4 describes the experimental results and discussion. The conclusions are mentioned in Section 5.

2. Problem Statement and Preliminaries

There are more than 500 species of Caladium bicolor found in Thailand. Most of them are new species, which were produced through multiplication across the original one; therefore, each of them has different characteristics, especially in the color pattern and leaf shape [3] and [4]. To classify Caladium bicolor from the color pattern is very difficult because there are many diversities in color within a leaf, even though they are of the same breed. This paper presents the classification of Caladium bicolor using shape features with a transfer learning technique. This work also presents a method of using A/B testing to conduct the assessment of GUI design used to deploy the application of the Caladium bicolor classification.

Our previous work used the transfer learning technique with the Harr-like feature extracted from images to classify the Thai Buddha amulet. We gathered Buddha amulet images from Google and used the data augmentation method to increase our images for training the model. The Haar-like feature provided robustness to the training model since it was very helpful in capturing the boundary feature of the Buddha amulet with low dimensions to generate the classification model. Our previous work obtained the results of 0.9367, 0.9379, and 0.9373 for precision, recall, and F1 score, respectively [5]. To understand the revolution of plants, the study used the ML technique to learn a large, complex dataset of plant genotyping and phenotyping. The study used ML to extract the feature of the gene structure of plants to learn, analyze, and predict the mutation in plant breeding [6]. A later study claimed that the work was the first plant image dataset in a natural scene collected using a mobile phone. The dataset

contains 10000 images of 100 plant breeds captured on the Beijing Forestry University campus. The work also claimed that the model yielded 91.78% of a recognition rate [7]. The extraction of plant leaves and the usage of leaf features as training data were introduced in [8]. The study showed how to segment the plant leaf and extract the shape feature apart from plant texture to train a deep model. The evaluation of the model was also exhibited and compared with the KNN-based neighborhood classification, Kohonen network based on a self-organizing feature mapping algorithm, and SVM-based support vector machine. A simple KNN-based classifier for extracting the surface of plant was also introduced in [9]. Unlike other works, this work focuses on the surface features of plants to use in a robotized vision framework for harvesting and farming objectives. Dried plant images have also been used to study whether the plant organ could be observed if the water component was extracted [10]. The objective of this work was to focus on plant development under a restricted water condition, especially during the dry season. The methods used in this work were SVM, ANN, and some image-processing techniques to highlight the dry texture of a plant. In [11], the evaluation of eight effective approaches (e.g., Random Forest, SVM, RestNet50, CNN, VGG16, VGG19, PNN, and KNN) was done on the Flavia plant-leaf image dataset. By aiming that the results of the study can help save plants worldwide, this work also used plant leaves as an important feature to improve information on plant classification and recognition. Recent works on plant disease classification based on ML were reviewed in [12] and [13]. Similar to plant classification applications, the major contribution of plant disease classification is the preparation of plant disease datasets. The gathering of plant disease images is an important step because researchers must trade off which plants are important to the market, and the image gathering must be taken only in the harvesting seasons; otherwise, they must wait until the next year. The popular models used to classify plant disease are GoogleNet, RestNet, VGG, KNN, etc.

More works on plant disease detection and flower classification using image processing are introduced in [14], [15], [16], [17], and [18]. In [14], the deep learning technique was applied to improve plant health and food crop productivity in the ecology-based farming system. By using CNN-based computer vision and remote sensing techniques, the farmer can access the system and observe the farming environment via the cloud. The disease experts can give their advice, and the system can also give the basic recommendation of plant diseases. The image dataset includes the major diseases of rice, potato, and tomato, with more than 7500 images. The classification results gave 0.98% of accuracy. In [15] and [16], these recent works proposed the feature extraction techniques for flower images. These features were also fed to the well-known pre-trained models such as DenseNet121, ResNet 50, ResNet 101, Inception ResNet V2, Inception V2, NAS, and MobileNet V2. The classification analysis was compared in these works. In [17], the segmentation technique was applied to leaf images to segment a leaf region from the overlapped background image. The dataset contains 2500 leaf images of 15 species with a complicated background. The method used 2000 images to train a model for leaf segmentation by using a mask Region-based convolutional neural network (Mask R-CNN). By fine-tuning the hyperparameters of segmentation, the method gained 91.5% accuracy with 1.15% of misclassification error. In [18], this work proposed the modified VGG16 model to classify five categories of common flowers, namely, daisy, dandelion, sunflower, rose and tulip flowers. The model was trained using 3520 flower images with 3×3 filters. This simple model gained an accuracy of 0.95 on the Kaggle dataset.

3. Implementation

We collected Caladium bicolor images from Google images by using the Google custom

image search and the JavaScript console in Chrome and retrieved a bunch of Caladium bicolor images in Thai local names according to five classes as follows: 1) Bon Bai Klom, 129 images; 2) Bon Bai Kab, 107 images; 3) Bon Bai Thai, 125 images; 4) Bon Bai Pai, 104 images; and 5) Bon Bai Yao, 120 images (total images collected from Google = 585 images). Some examples of Caladium bicolor images are shown in **Table 1**. We implemented our work using MATLAB R2023a (9.14.0.2286388). We created an image data store according to the folder labeled by the class name with the MATLAB command `imageDatastore`. We separated our data store into a training set (80% of the data store) and a validation set (20% of the data store) using the MATLAB command `splitEachlabel`. The major problem with our work was the small dataset; therefore, we had to increase the number of our images. We used the MATLAB command `augmentedImageDatastore` for both training and validating datasets. MATLAB data augmentation provides some useful image processing techniques, e.g., rotating, flipping, resizing, and translating, to increase the number of Caladium bicolor images. **Fig. 1** shows the MATLAB augmentation results.

Table 1. Caladium bicolor images from Google custom image search

Caladium bicolor	**Images retrieved from Google**
Bon Bai Klom	
Bon Bai Kab	
Bon Bai Thai	

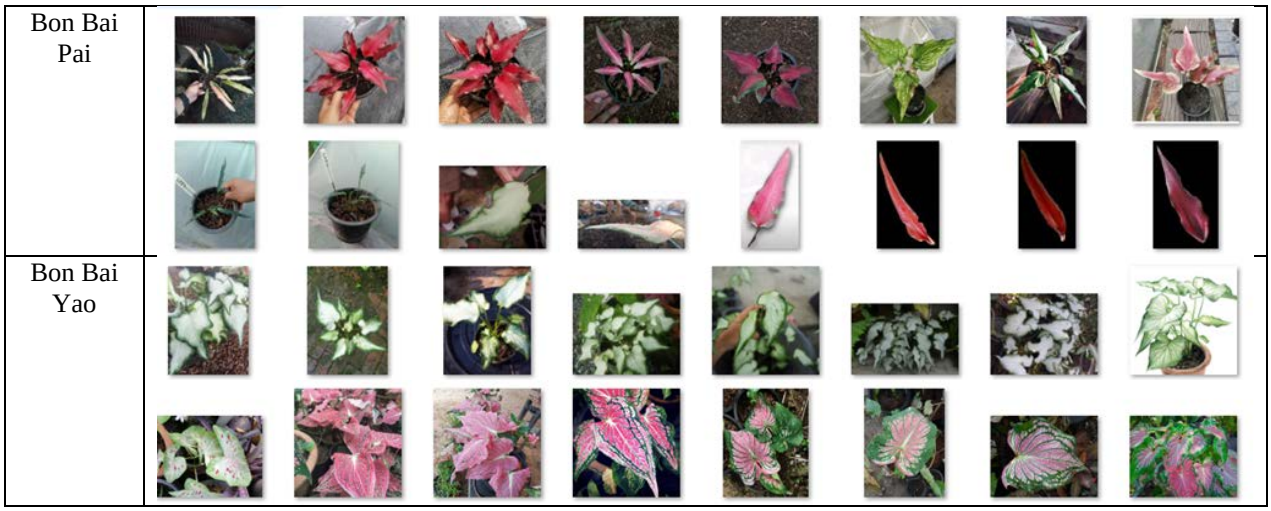


Fig. 1. Data augmentation for Caladium bicolor images.

To train our model, we used efficient pre-trained models to reduce the computational cost of training a brand-new deep learning model. In our experiment, we chose AlexNet, GoogleNet, XceptionNet, MobileNetV2, RestNet18, RestNet50, RestNet101, and InceptionResNetV2 as the pre-trained models to solve our new classification problem. We used a MATLAB network designer and adopted some parameters for our experiment. **Table 2** provides some important parameters, which we used in our training process.

Table 2. Parameter configuration for the MATLAB network designer

Pre-trained model	Optimization algorithm	Execution environment	Mini batch size	Learning rate
AlexNet	sgdm	Parallel	100	0.0003
GoogleNet	sgdm	Parallel	100	0.0003
XceptionNet	adam	Parallel	64	0.001
MobileNetV2	sdgm	Parallel	50	0.0003
RestNet18	sdgm	Parallel	100	0.0003

RestNet50	adam	Parallel	128	0.0003
RestNet101	sgdm	Parallel	20	0.0003
InceptionResNetV2	adam	Parallel	65	0.0001

Fig. 2 shows the system block diagram and the proposed model. Briefly, the contribution of this work includes: 1) creating the image data store (Caladium image dataset), 2) augmentation of the image dataset using MATLAB image processing, 3) training eight pre-trained models, 4) model evaluation/selection, 5) design MATLAB GUI based on the user-centered concept, 6) conducting the A/B testing and user testing, and 7) deployment of the model.

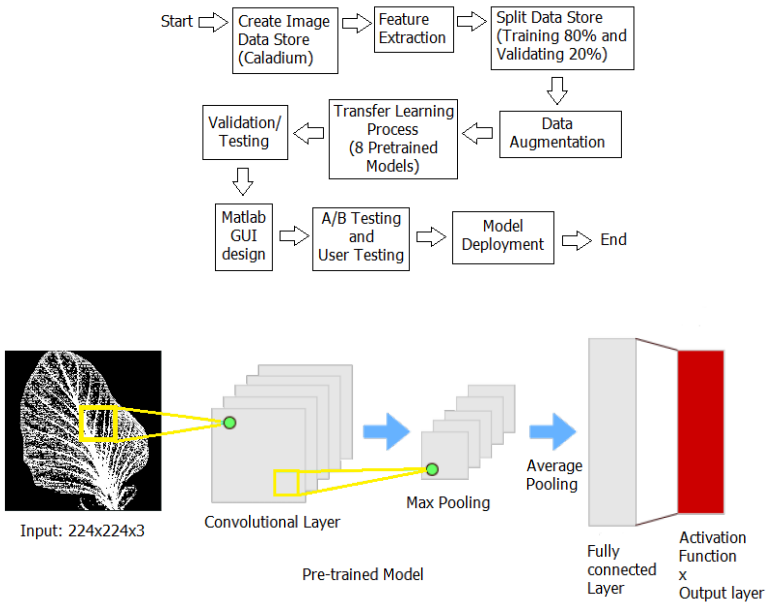


Fig. 2. System model.

We set the output size to five, according to the five classes of Caladium bicolor images. We set the network architecture with the following input and output layers: 1) the input layer (Caladium bicolor input image): the input = 224×224×3, and generated the output = 7×7×64; 2) the bypass layer 1: the input = 1×1×64, and generated the output = 1×1×256; 3) the bypass layer 2: the input = 1×1×128, and generated the output = 1×1×512; 4) the bypass layer 3: the input = 1×1×256, and generated the output = 1 × 1 × 1024; 5) the bypass layer 4: the input = 1 × 1 × 512, and generated the output = 1 × 1 × 2048; 6) the output layer: we used the average pooling and five layers of the fully connected layer. The training results and parameter comparisons are shown in Section 4.

4. Results and discussion

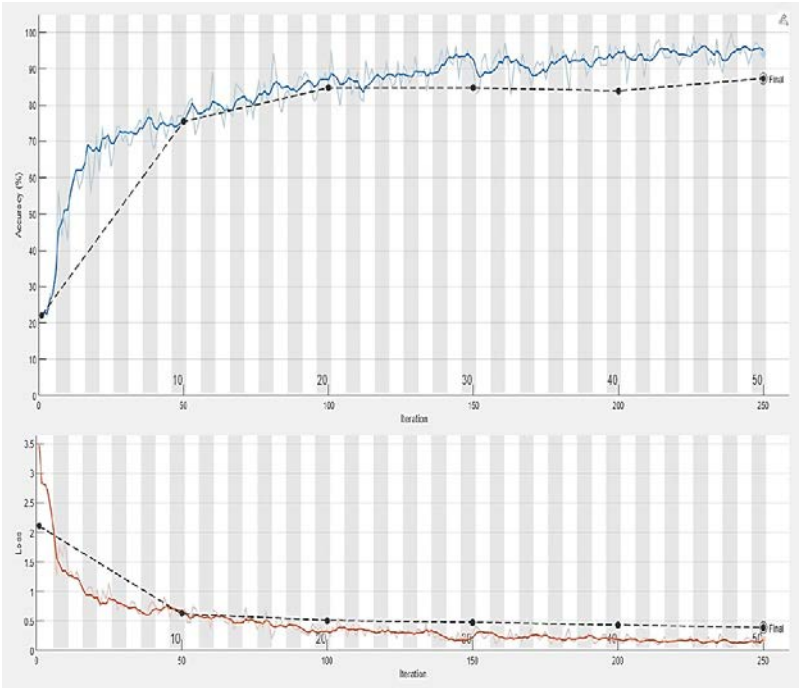
In this section, we compare the training parameters gained from the MATLAB network designer, and we choose the best candidate for the pre-trained model to deploy in the Caladium bicolor classification application. We also show a user-testing process based on the UX/UI concept to guarantee that our application meets user requirements. **Table 3** shows the experimental results.

Table 3. Training process and parameters

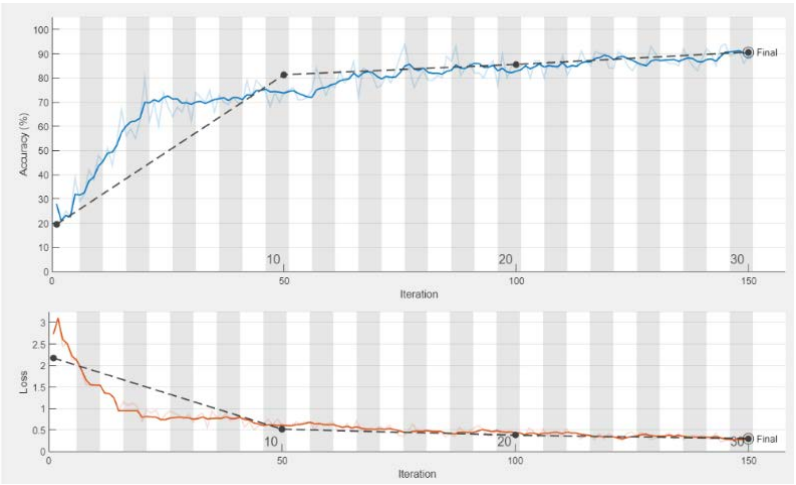
Pre-trained model	Epoch	Iteration	Time elapsed (hh:mm:ss)	Mini batch accuracy (%)	Validation accuracy (%)	Mini batch loss	Validation loss	Base learning rate
AlexNet	1	1	00:00:02	21.00	22.03	3.4731	2.1082	0.0003
	10	50	00:00:41	77.00	75.42	0.6125	0.6275	0.0003
	20	100	00:01:20	86.00	84.75	0.3746	0.5124	0.0003
	30	150	00:02:01	90.00	84.75	0.2903	0.4813	0.0003
	40	200	00:02:41	95.00	83.90	0.2560	0.4369	0.0003
	50	250	00:03:22	94.00	87.29	0.2116	0.3902	0.0003
GoogleNet	1	1	00:00:21	28.00	19.49	2.7256	1.2415	0.0003
	10	50	00:11:01	74.00	81.36	0.6186	0.3318	0.0003
	20	100	00:22:15	85.00	85.59	0.4293	0.2726	0.0003
	30	150	00:32:02	89.00	90.68	0.3284	0.2281	0.0003
XceptionNet	1	1	00:03:28	17.19	44.44	1.5895	1.9206	0.0010
	1	5	00:11:26	67.19	77.78	0.8741	0.6176	0.0010
	2	10	00:22:19	87.50	84.19	0.3799	0.5683	0.0010
	3	15	00:33:34	76.56	91.03	0.6243	0.3108	0.0010
	3	20	00:44:13	96.88	92.74	0.1134	0.2554	0.0010
	4	25	00:54:44	95.31	88.89	0.0961	0.3281	0.0010
	5	30	01:04:34	90.62	83.33	0.2120	0.4810	0.0010
	5	35	01:15:30	90.62	89.32	0.2747	0.4699	0.0010
	6	40	01:26:06	93.75	83.03	0.2685	0.3409	0.0010
	7	45	01:36:38	90.62	83.33	0.4378	0.4626	0.0010
	8	50	01:48:56	90.62	88.46	0.2022	0.3814	0.0010
	8	55	02:01:27	89.06	91.88	0.3503	0.3325	0.0010
	8	56	02:05:23	93.75	91.45	0.2431	0.3529	0.0010
RestNet18	1	1	00:02:47	14.00	14.41	2.3456	1.8494	0.0003
	10	50	02:23:40	60.00	80.51	0.9926	0.7212	0.0003
	20	100	04:49:23	86.00	90.68	0.4438	0.4253	0.0003
	30	150	06:59:16	87.00	89.83	0.4201	0.3202	0.0003
MobileNetV2	1	1	0:00:58	8.00	12.71	1.9866	1.8877	0.0003
	3	30	0:12:51	64.00	57.63	1.0174	1.0314	0.0003
	5	50	0:21:27	78.0	64.41	0.6756	0.7946	0.0003
	7	70	0:27:29	90.00	77.12	0.3830	0.6622	0.0003
	9	90	0:33:10	84.00	79.66	0.4150	0.6094	0.0003
	11	110	0:38:45	84.00	79.66	0.3854	0.5300	0.0003
	13	130	0:44:27	90.00	84.75	0.3494	0.4802	0.0003
	15	150	0:49:55	96.00	82.20	0.1890	0.4596	0.0003
	17	170	0:55:25	92.00	83.90	0.2481	0.4254	0.0003
	19	190	1:00:51	98.00	83.90	0.1295	0.4143	0.0003
	23	230	1:11:40	100.00	92.37	0.0875	0.3582	0.0003

	30	300	1:30:52	100.00	89.83	0.0898	0.3876	0.0003
	50	500	2:24:17	100.00	90.68	0.0453	0.2609	0.0003
RestNet50	1	1	00:03:24	20.00	52.00	2.0021	1.2030	0.0003
	2	5	00:22:24	82.12	68.02	0.4800	0.9824	0.0003
	3	10	00:46:13	88.70	82.00	0.2020	0.4812	0.0003
	4	15	01:16:24	97.78	84.28	0.1882	0.4800	0.0003
	5	20	01:34:34	98.24	86.14	0.1800	0.4880	0.0003
	6	25	01:44:06	98.12	86.00	0.1414	0.4812	0.0003
	7	30	01:52:56	98.00	86.16	0.1202	0.4400	0.0003
	8	32	02:04:08	98.98	88.00	0.1200	0.4224	0.0003
RestNet101	1	1	00:00:26	15.00	14.41	1.8060	2.1063	0.0003
	5	110	00:05:39	95.00	88.14	0.1331	0.3962	0.0003
	10	240	00:11:08	100.00	91.53	0.0453	0.2653	0.0003
	15	370	00:16:42	100.00	96.61	0.0116	0.1555	0.0003
	20	500	00:22:28	95.00	94.92	0.0815	0.1629	0.0003
	25	630	00:28:01	100.00	94.92	0.0416	0.1304	0.0003
	30	760	00:33:31	100.00	94.92	0.0031	0.1631	0.0003
InceptionResNetV2	1	1	00:01:50	21.54	34.75	1.5927	1.4709	0.0001
	7	50	00:44:05	100.00	94.07	0.0041	0.3262	0.0001
	13	100	01:26:49	100.00	94.92	0.0016	0.2851	0.0001
	19	150	02:09:19	96.92	95.76	0.1204	0.1929	0.0001
	25	200	02:52:20	100.00	91.53	0.0022	0.4907	0.0001
	32	250	03:35:12	100.00	94.07	0.0022	0.2772	0.0001
	38	300	04:18:11	100.00	94.92	0.0002	0.1696	0.0001
	44	350	05:00:54	100.00	93.22	0.0010	0.2287	0.0001
	50	400	05:43:38	100.00	93.22	0.0003	0.3125	0.0001

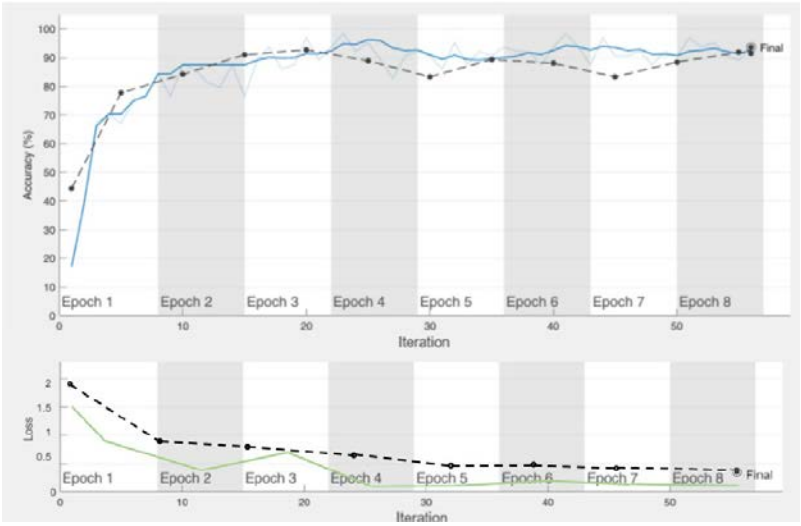
The experimental results are shown: AlexNet yielded 87.29% accuracy at 50 epochs, GoogleNet yielded 90.68% accuracy at 30 epochs, XceptionNet yielded 93.59% accuracy at 8 epochs, MobileNetV2 yielded 93.22% accuracy at 50 epochs, RestNet18 yielded 89.83% accuracy at 30 epochs, RestNet50 yielded 88.98% accuracy at 8 epochs, RestNet101 yielded 97.46% accuracy at 30 epochs, and InceptionResNetV2 yielded 94.92% accuracy at 50 epochs. The accuracy and loss of all pre-trained models, together with the overall training time, are shown in **Fig. 3** and **Table 4**, respectively. As the results of our experiment, we focused on the two best candidates among the eight pre-trained models, which were RestNet101 and InceptionResNetV2. RestNet101 gained a validation accuracy of 97.46% and validation loss of 0.01% with an overall running time of 34 minutes and 23 seconds, whereas InceptionResNetV2 gained a validation accuracy of 94.92% and validation loss of 0.28% with an overall running time of 344 minutes and 5 seconds. Therefore, we considered RestNet101 as the classification model to the application deployment in the next step.



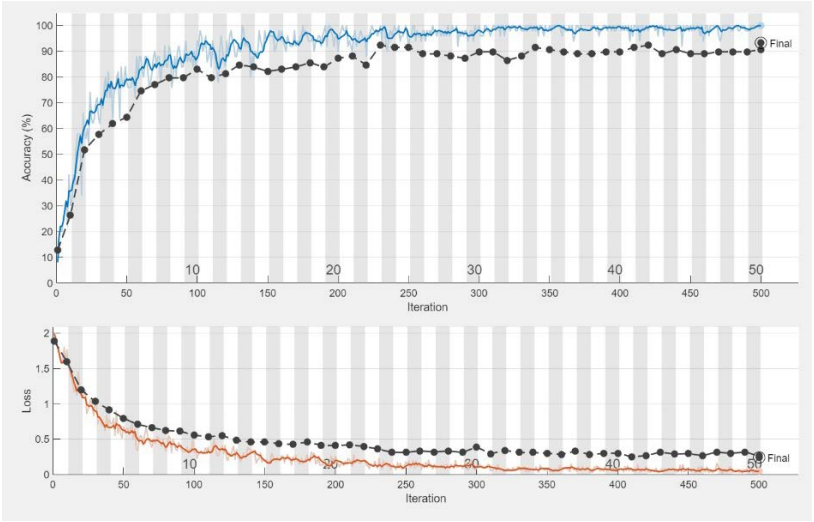
(a) AlexNet



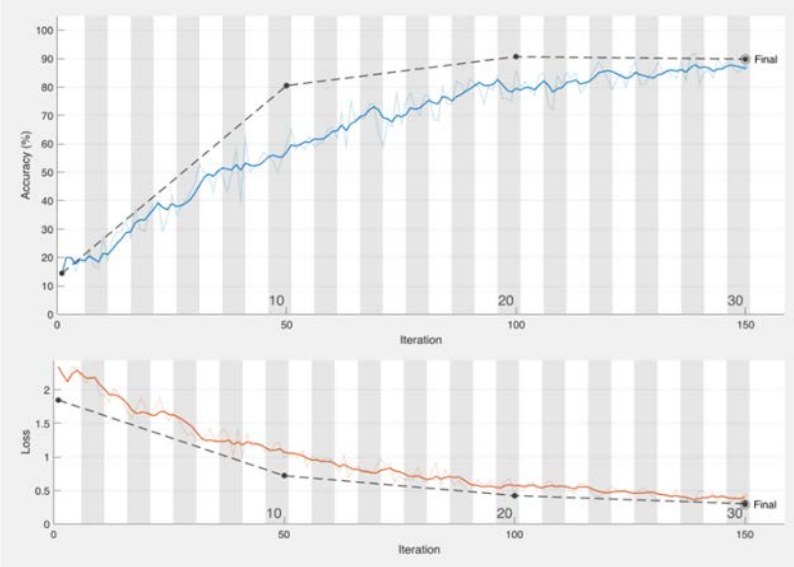
(b) GoogleNet



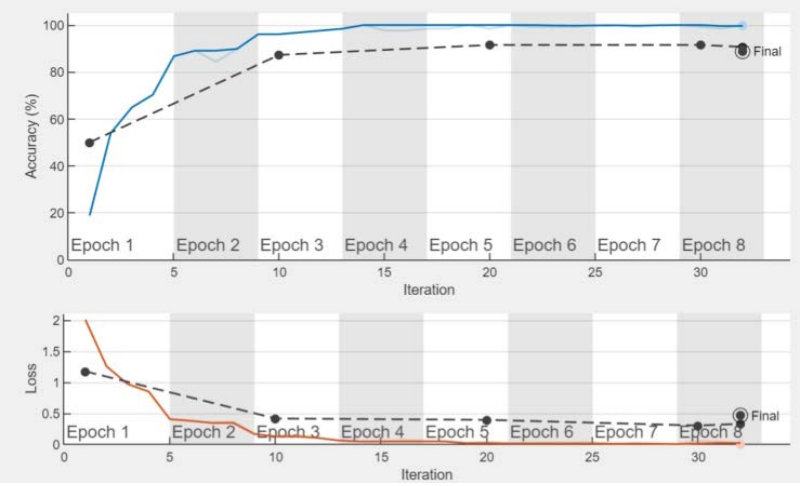
(c) XceptionNet



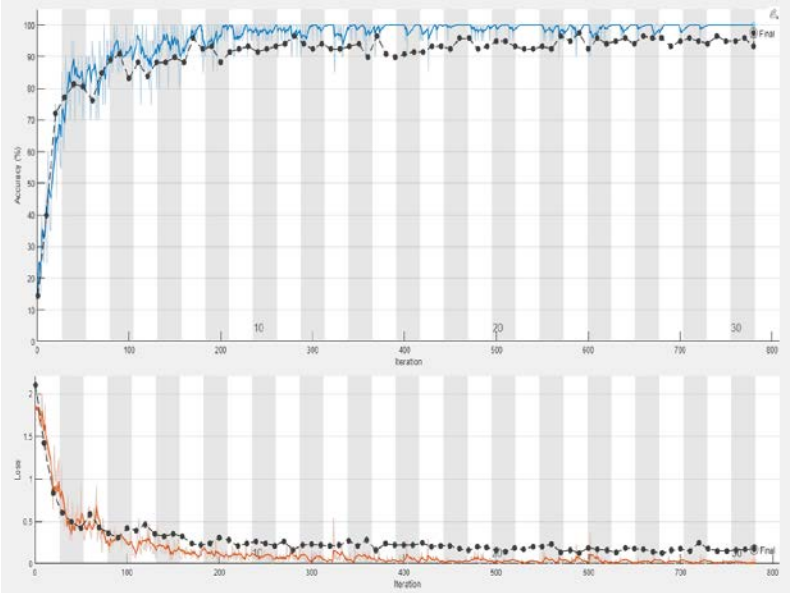
(d) MobileNetV2



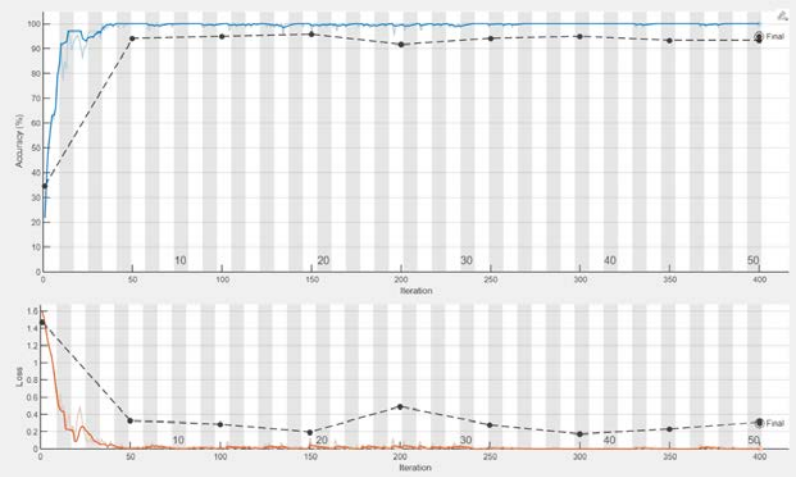
(e) RestNet18



(f) RestNet50



(g) RestNet101



(h) InceptionResNetV2

Fig. 3. Accuracy and loss of pre-trained models.

Table 4. Comparisons of training parameters

Pre-trained model	Training epoch	Training time	Validation accuracy
AlexNet	50 Epoch	3 min 22 sec	87.29%
GoogleNet	30 Epoch	32 min 6 sec	90.68%
XceptionNet	8 Epoch	127 min 23 sec	93.59%
MobileNetV2	50 Epoch	144 min 25 sec	93.22%
RestNet18	30 Epoch	419 min 35 sec	89.83%
RestNet50	8 Epoch	124 min 8 sec	88.98%
RestNet101	30 Epoch	34 min 23 sec	97.46%
InceptionResNetV2	50 Epoch	344 min 5 sec.	94.92%

We also evaluated our pre-trained models in terms of precision, recall, accuracy, and F1 score using equations (1)–(4):

$$Precision = \frac{TP}{(TP+FP)} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

$$\text{Accuracy} = \frac{\text{Correct classification}}{\text{Number of entire instance set}} \tag{3}$$

$$\text{F1 score} = \frac{2\times(\text{Precision}\times\text{Recall})}{(\text{Precision}+\text{Recall})} \tag{4}$$

where TP is true positive and defined as the correct classified type of activities. TN is true negative. FP is false positive. FN is false negative. **Table 5** shows the classification results of the proposed method.

Table 5. Classification results

Pre-trained model	Class	Accuracy	Precision	Recall	F1 Score
AlexNet	Bon Bai Kab	0.958	0.815	1.000	0.898
	Bon Bai Klom	0.958	1.000	0.808	0.894
	Bon Bai Pai	0.957	0.840	0.955	0.894
	Bon Bai Thai	0.949	0.950	0.792	0.864
	Bon Bai Yao	0.924	0.800	0.833	0.816
	Avg.	0.949	0.881	0.877	0.873
GoogleNet	Bon Bai Kab	0.968	0.815	1.000	0.898
	Bon Bai Klom	0.959	1.000	0.808	0.894
	Bon Bai Pai	0.971	0.840	0.955	0.894
	Bon Bai Thai	0.968	0.950	0.792	0.864
	Bon Bai Yao	0.961	0.800	0.833	0.816
	Avg.	0.965	0.901	0.877	0.884
XceptionNet	Bon Bai Kab	0.967	0.958	0.885	0.920
	Bon Bai Klom	0.983	1.000	0.905	0.950
	Bon Bai Pai	0.967	0.885	0.958	0.920
	Bon Bai Thai	0.970	0.964	0.931	0.947
	Bon Bai Yao	0.972	0.875	1.000	0.933
	Avg.	0.972	0.936	0.936	0.934
MobileNetV2	Bon Bai Kab	0.983	0.955	0.955	0.955
	Bon Bai Klom	0.949	0.857	0.923	0.889
	Bon Bai Pai	0.966	0.875	0.955	0.913
	Bon Bai Thai	0.975	1.000	0.875	0.933
	Bon Bai Yao	0.992	1.000	0.958	0.979
	Avg.	0.973	0.937	0.933	0.934
RestNet18	Bon Bai Kab	0.967	1.000	0.846	0.917
	Bon Bai Klom	0.951	0.900	0.818	0.857
	Bon Bai Pai	0.953	0.923	1.000	0.960
	Bon Bai Thai	0.954	0.917	1.000	0.957
	Bon Bai Yao	0.954	0.933	1.000	0.966
	Avg.	0.956	0.935	0.933	0.931
RestNet50	Bon Bai Kab	0.949	0.864	0.864	0.864
	Bon Bai Klom	0.983	0.929	1.000	0.963
	Bon Bai Pai	0.958	0.870	0.909	0.889
	Bon Bai Thai	0.941	0.947	0.750	0.837
	Bon Bai Yao	0.934	0.808	0.875	0.840
	Avg.	0.953	0.883	0.880	0.879

RestNet101	Bon Bai Kab	1.000	1.000	1.000	1.000
	Bon Bai Klom	1.000	1.000	1.000	1.000
	Bon Bai Pai	0.983	0.917	1.000	0.957
	Bon Bai Thai	0.992	1.000	0.958	0.979
	Bon Bai Yao	0.975	0.957	0.917	0.936
	Avg.	0.990	0.975	0.975	0.974
InceptionResNetV2	Bon Bai Kab	0.983	1.000	0.905	0.950
	Bon Bai Klom	0.967	0.958	0.885	0.920
	Bon Bai Pai	0.975	0.964	0.931	0.947
	Bon Bai Thai	0.967	0.885	0.958	0.920
	Bon Bai Yao	0.975	0.875	1.000	0.933
	Avg.	0.974	0.936	0.936	0.934

The confusion charts for RestNet101 and InceptionResNetV2 are also shown in **Fig. 4** and **Fig. 5**, respectively. These charts show the classes that networks were not able to classify accurately; therefore, we can recheck our data and retrain networks with improved data to see whether they can be corrected. We observe that in InceptionResNetV2, the class Bon Bai Klom is the most misclassified class (11.5% of misclassification). In this case, we can improve the mini-batch accuracy of InceptionResNetV2 using the MATLAB command `knnsearch`, which finds the nearest neighbors of Bon Bai Klom images using the K-D Tree search algorithm. The K-D Tree algorithm is a simple but efficient search algorithm that browses split nodes in two directions (same as binary search Tree). Using the median as the split criterion, the search result can be obtained quickly according to a balanced structure [19] and [20]. **Fig. 6** shows the modified layer and corrected classification of the class Bon Bai Klom.

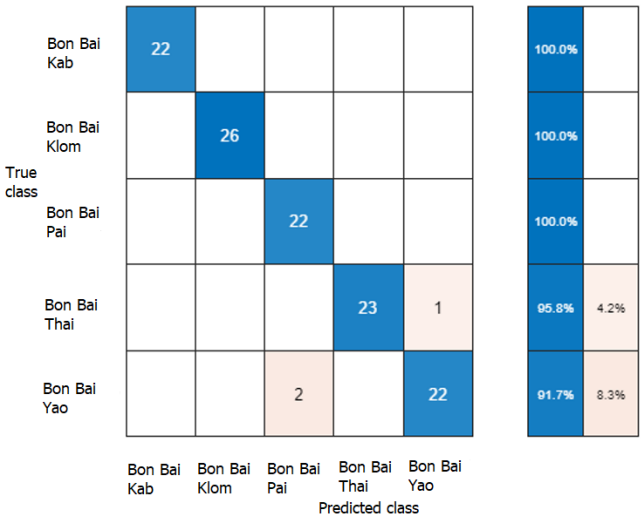


Fig. 4. Confusion chart of RestNet101.

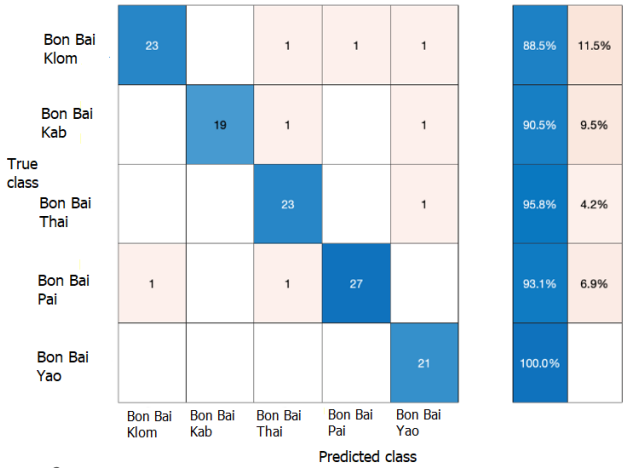


Fig. 5. Confusion chart of InceptionResNetV2.

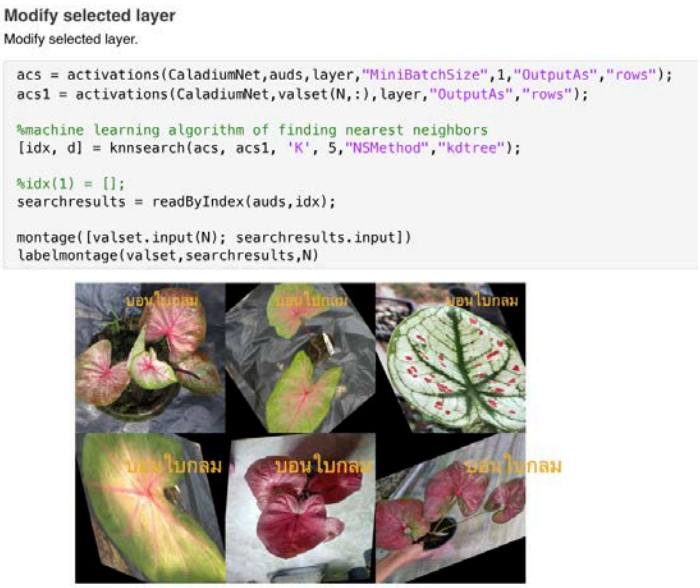
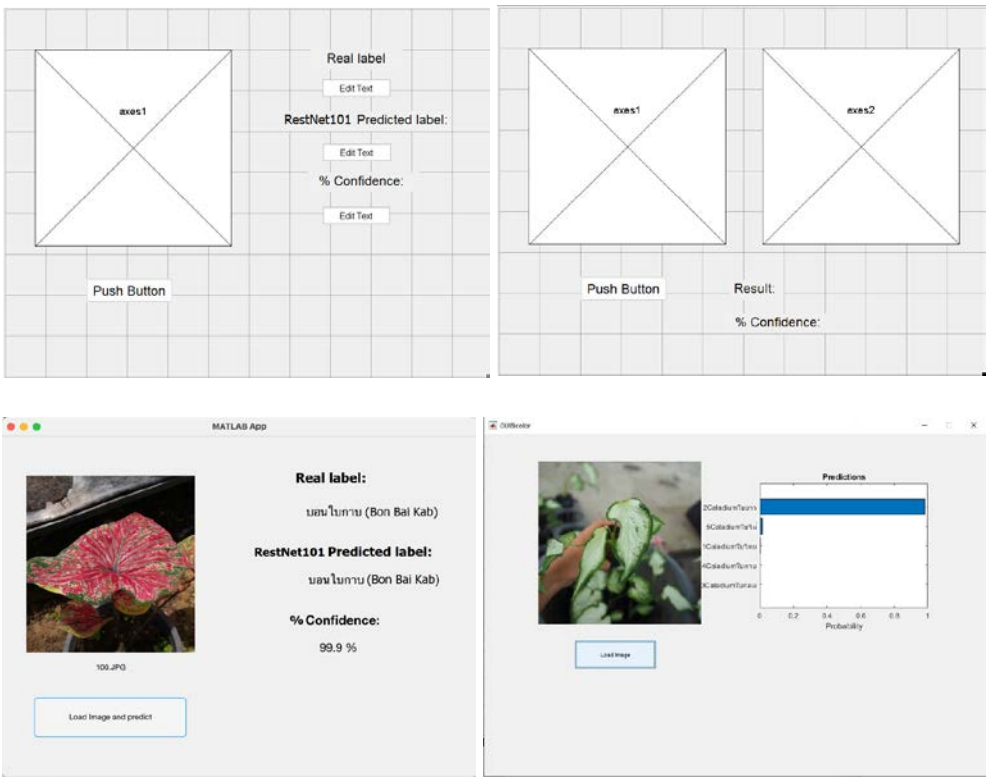


Fig. 6. Result of the MATLAB command `knnsearch` to modify the selected layer.

As seen in **Fig. 4** and **Fig. 5**, RestNet101 gave us better classification results compared to InceptionResNetV2. Therefore, we chose to deploy RestNet101 for our application. In the next step, we deployed RestNet101 to the Caladium bicolor application. To gather the user requirements, we listed some MATAB GUI (e.g., panel, button, axes, edit text, etc.) that were required for designing the application as choices for selection. After the layout design, we made two variants of the application GUI design and conducted A/B testing to figure out which GUI was the most appropriate and met to the user need [21]. **Fig. 7** shows the two variants used for A/B testing. In variant no.1, the GUI consists of axis no.1 for loading an image to be classified by pressing a push button, the results of classification will be shown in the application by appearing in three edit text objects. In variant no.2, the GUI consists of two

axes: the first one is used to load an image for classification, and the second one is used to show the result in a graph-like format. Both of the two GUIs also provide the results of the class label and % confidence level.



(a) Variant A (b) Variant B

Fig. 7. Two variants GUI for running A/B testing.

A/B testing was conducted in an HCI course (040613349), which included 24 students in this class, and 98% of the students reported that they were familiar with the concept of UX/UI and the user testing process. To run the test without any bias, we did this test as a partial of the user-testing class. After a class lecture, we gave all the information necessary for performing A/B testing, together with instructions to do the user testing and the expectations of the experiment. To evaluate A/B testing, all students were asked to complete a questionnaire "System Usability & UX Questionnaire." We designed this questionnaire using five questions, which were adopted based on the usability criteria defined by ISO 9241-11 and the UX criteria [22]. All the questions were designed to cover the necessary topics of the UX/UI concept (utility, functionality, ease of use, consistency, and satisfaction). We designed a questionnaire with five response options ranging from strongly disagree to strongly agree (1–5). The questionnaire also included an open question to provide more specific feedback.

The evaluation results of the two variants are shown in **Table 6** (average and standard deviation). The responses were statistically significant for one variable: ease of use ($p<0.05$). Whereas four variables were exceptional: utility, functionality, consistency, and satisfaction ($p>0.05$). The p-values for the five variables are also shown in **Table 6**. Therefore, we chose a GUI design for variant B to deploy the RestNet101 model. **Fig. 8** shows the final GUI of the Caladium bicolor classification application, where an input image can be loaded on the left

side of the panel, and the result of the classification is shown on the right side of the panel. The predicted label is shown together with the confidence scores of the RestNet101 prediction.

Table 6. Average rating of two variants and p-value (*sig)

	Utility	Functionality	Ease of use*	Consistency	Satisfaction
Variant A	4.29 (std. 0.79)	4.33 (std. 0.75)	4.25 (std. 1.05)	4.63 (std. 0.56)	4.38 (std. 0.63)
Variant B	4.25 (std. 0.78)	4.50 (std. 0.65)	4.54 (std. 0.91)	4.67 (std. 0.55)	4.50 (std. 0.87)
p-value	0.227	0.366	0.011	0.334	0.5

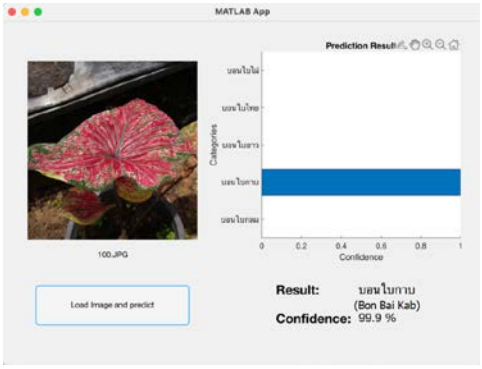


Fig. 8. Screen capture of the application.

5. Conclusion

The contribution of this work is the classification of Caladium bicolor. There are more than 500 breeds of Caladium bicolor in Thailand. Caladium bicolor is one of the most popular plants used for gardening and planting during the COVID-19 pandemic. In this study, we proposed a transfer learning technique to classify Caladium bicolor using the shape features of Caladium bicolor. We gathered Caladium bicolor images from Google images using Google custom image search and the JavaScript console in Chrome, and we also increased the number of images using the MATLAB augmentation process to generate more images for training the model. Eight efficient and well-known pre-trained models were conducted in our experiments, which were AlexNet, GoogleNet, XceptionNet, MobileNetV2, RestNet18, RestNet50, RestNet101, and InceptionResNetV2. The experimental results showed that RestNet101 and InceptionNetV2 yielded the best results among the eight pre-trained models. RestNet101 gained a validation accuracy of 97.46% with an overall running time of 34 minutes and 23 seconds, whereas InceptionResNetV2 gained a validation accuracy of 94.92% with an overall running time of 344 minutes and 5 seconds. Therefore, we selected RestNet101 as the model to deploy into the Caladium bicolor classification application. The GUI of the application was designed based on the UX/UI concept; we conducted an A/B testing process to search for the best appropriate GUI design for the application using the questionnaire. The questionnaire was designed based on ISO 9241-11 and UX criteria (utility, functionality, ease of use, consistency, and satisfaction). The statistical responses showed that there was a statistical difference in one UX criterion: ease of use ($p<0.05$). The screen captured for the classification application is shown in **Fig. 8**, where the classification of the Caladium bicolor image gives the correct label (Bon Bai Kab) with 99.99% confidence scores.

Acknowledgment

This research was funded by the Faculty of Applied Science, King Mongkut's University of Technology, North Bangkok, Thailand (Contract No. 662145).

This research partly used computational resources from the China Scholarship Council (CSC) for the Senior Visiting Scholarship Program provided by the School of Computer Science and Technology, Beijing Institute of Technology (BIT).

References

[1] A. C. Maia and C. Schlindwein, "Caladium bicolor (Araceae) and Cyclocephala celata (Coleoptera, Dynastinae): A Well-established Pollination System in the Northern Atlantic Rainforest of Pernambuco, Brazil," *Plant Biology*, 8(4), pp. 529-534, 2006. Article (CrossRef Link).

[2] E. U. Ahmed, T. Hayashi, Y. Zhu, M. Hosokawa and S. Yazawa, "Lower Incidence of Variants in Caladium bicolor Ait. Plants Propagated by Culture of Explants from Younger Tissue," *Scientia Horticulturae*, 96(1–4), pp. 187-194, 2002. Article (CrossRef Link)

[3] W. Hetterscheid, J. Bogner and J. Boos, "Two New Caladium Species. Aroideana," *Journal of the International Aroid Society*, 32, pp. 126-131, 2009.

[4] C. Ekeke and I. O. Agbagwa, "Anatomical Characteristics of Nigerian Variants of Caladium bicolor (Aiton) Vent. (Araceae)," *African Journal of Plant Science*, 10(7), pp. 121-129, 2016. Article (CrossRef Link).

[5] P. Visutsak, T. Kuarkamphun and N. Samleepant, "Thai Buddha Amulet Classification Using Discrete Wavelet Transform and Transfer Learning," *ICIC Express Letter*, 16(11), pp. 1205-1214, 2022. Article (CrossRef Link).

[6] A. D. J. V. Dijk, G. Kootstra, W. Kruijer and D. D. Ridder, "Machine Learning in Plant Science and Plant Breeding," *iScience*, 24(1), 2021. Article (CrossRef Link).

[7] Y. Sun, Y. Liu, G. Wang and H. Zhang, "Deep Learning for Plant Identification in Natural Environment," *Computational Intelligence and Neuroscience*, vol. 2017, 6 pages, 2017. Article (CrossRef Link).

[8] J. Huixian, "The Analysis of Plants Image Recognition Based on Deep Learning and Artificial Neural Network," *IEEE Access*, 8, pp. 68828-68841, 2020. Article (CrossRef Link).

[9] G. Valarmathi, S.U. Suganthi, V. Subashini, R. Janaki, R. Sivasankari and S. Dhanasekar, "CNN Algorithm for Plant Classification in Deep Learning," *Materials Today: Proceedings*, 46(9), pp. 3684-3689, 2021. Article (CrossRef Link).

[10] K. Shobana and P. Perumal, "Plants Classification Using Machine Learning Algorithm," in *Proc. of 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 96-100, 2020. Article (CrossRef Link).

[11] S. Ghosh and A. Singh, "The Analysis of Plants Image Classification Based on Machine Learning Approaches," in *Emergent Converging Technologies and Biomedical Systems*, pp. 133-148, 2022. Article (CrossRef Link).

[12] D. Gosai, B. Kaka, D. Garg, R. Patel and A. Ganatra, "Plant Disease Detection and Classification Using Machine Learning Algorithm," in *Proc. of 2022 International Conference for Advancement in Technology (ICONAT)*, Goa, India, pp. 1-6, 2022. Article (CrossRef Link).

[13] T. S. Xian and R. Ngadiran, "Plant Diseases Classification Using Machine Learning," *Journal of Physics: Conference Series*, 1962(1), p. 012024, 2021. Article (CrossRef Link).

[14] R. Sharma, A. Singh, N. Z. Jhanjhi, M. Masud, E. S. Jaha and S. Verma, "Plant Disease Diagnosis and Image Classification Using Deep Learning," *Computers, Materials & Continua*, 71(2), 2022. Article (CrossRef Link).

[15] N. Alipour, O. Tarkhaneh, M. Awrangjeb and H. Tian, "Flower Image Classification Using Deep Convolutional Neural Network," in *Proc. of 2021 7th International Conference on Web Research (ICWR)*, pp. 1-4, 2021. Article (CrossRef Link).

[16] I. Patel and S. Patel, "An Optimized Deep Learning Model for Flower Classification Using NAS-FPN and Faster R-CNN," *International Journal of Scientific & Technology Research*, 9(03), pp. 5308-5318, 2020.
[17] K. Yang, Z. Weizhen and L. Fengguo, "Leaf Segmentation and Classification with a Complicated Background Using Deep Learning," *Agronomy*, 10(11), p. 1721, 2020. Article (CrossRef Link).
[18] S. Giraddi, S. Seeri, P. S. Hiremath and J. G. N, "Flower Classification Using Deep Learning models," in *Proc. of 2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, Bengaluru, India, pp. 130-133, 2020. Article (CrossRef Link).
[19] M. A. Jabbar, B. L. Deekshatulu and P. Chandra, "Classification of Heart Disease Using K-Nearest Neighbor and Genetic Algorithm," *Procedia Technology*, 10, pp. 85-94, 2013. Article (CrossRef Link).
[20] P. Thanh Noi and M. Kappas, "Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery," *Sensors (Basel)*, 18(1), p. 18, 2018. Article (CrossRef Link).
[21] C. Kamolsin, F. Pensiri, K. H. Ryu and P. Visutsak, "The Evaluation of GUI Design Using Questionnaire and Multivariate Testing," in *Proc. of 2022 Research, Invention, and Innovation Congress: Innovative Electricals and Electronics (RI2C)*, pp. 191-195, 2022. Article (CrossRef Link).
[22] M. Speicher, "What is Usability? A Characterization Based on ISO 9241-11 and ISO/IEC 25010," *arXiv preprint arXiv:1502.06792*, 2015. Article (CrossRef Link).

Porawat Visutsak received PhD in Computer Science from KMITL. He was a Senior Visiting Scholar in the Computer Science and Technology Program, School of Computer Science and Technology, Beijing Institute of Technology, under the China Scholarship Council (CSC). He is now Associate Professor in the Department of Computer and Information Science, Faculty of Applied Science, KMUTNB.

Professor Xiabi Liu is with School of Computer Science and Technology, Beijing Institute of Technology (BIT), Beijing, China.

Professor Keun Ho Ryu is with Chungbuk National University, Chungbuk, Republic of Korea. He is Adjunct Professor of Faculty of Information Technology, Co-Director of Research Group, Data Science Laboratory, Ton Duc Thang University, Vietnam.

Naphat Bussabong received BSc in Computer Science from KMUTNB and he is now Master student in Computer Science Program, KMUTNB.

Nicha Sirikong is with Sriayudhya School, she is now Master student in Computer Science Program, KMUTNB.

Preeyaphorn Intamong received BSc in Computer Science, Chiang Mai Rajabhat University. She is now Master student in Computer Science Program, KMUTNB.

Warakorn Sonnui received BEng in Computer Engineering from Walailak University and he is now Master student in Computer Science Program, KMUTNB.

Siriwan Boonkerd received BSc in Computer Science, Kamphaeng Phet Rajabhat University. She is now Master student in Computer Science Program, KMUTNB.

Jirawat Thongpiem received BSc in Computer Science from Rajapruk University. He is Full Stack Developer at Ayudhya Capital Services Company Limited, and he is now Master student in Computer Science Program, KMUTNB.

Maythar Poonpanit received BEng in Computer Engineering from Kasetsart University and he is now Master student in Computer Science Program, KMUTNB.

Akarasate Homwiseswongsa is Senior Software Developer at MOHARA. He is now Master student in Computer Science Program, KMUTNB.

Kittipot Hirunwannapong received BEng in Computer Engineering from Kasetsart University Sriracha. He is now Master student in Computer Science Program, KMUTNB.

Chaimongkol Suksomsong received BSc in Computer Science from Rajamangala University of Technology Phra Nakhon. He is with KASIKORN Business-Technology Group (KBTG). He is now Master student in Computer Science Program, KMUTNB.

Rittikait Budrit received BEng in Computer Engineering from Suranaree University of Technology. He is now Master student in Computer Science Program, KMUTNB.