Journal of Computer Science

Original Research Paper

LeafsnapNet: An Experimentally Evolved Deep Learning Model for Recognition of Plant Species based on Leafsnap Image Dataset

[1,2,3*]Emmanuel Adetiba, [1]Oluwaseun T. Ajayi, [4]Jules R. Kala, [1,2]Joke A. Badejo, [1]Sunday Ajala and [5]Abdultaofeek Abayomi

[1]*Department of Electrical and Information Engineering, Covenant University, Canaanland, P.M.B 1023, Ota, Nigeria*
[2]*Covenant Applied Informatics and Communication Africa Center of Excellence, Covenant University, Canaanland, P.M.B 1023, Ota, Nigeria*
[3]*HRA, Institute for Systems Science, Durban University of Technology, P.O. Box 1334, Durban 4000, South Africa*
[4]*Companie d'Electricite de Cote d'Ivoire, Abidjan, Cote d'Ivoire*
[5]*Department of Information and Communication Technology, Mangosuthu University of Technology, P.O. Box 12363 Jacobs, 4026 Durban, South Africa*

Article history
Received: 11-11-2020
Revised: 23-12-2020
Accepted: 07-01-2021

Corresponding Author:
Emmanuel Adetiba,
Department of Electrical and Information Engineering,
Covenant University,
Canaanland, P.M.B 1023, Ota,
Nigeria
Email: emmanuel.adetiba@covenantuniversity.edu.ng

Abstract: Plants are very important living organisms on earth because humans and animals depend on them for nutrition, oxygen, medicine and balance in the ecosystem. Therefore, plant species recognition is critical to the improvement of agricultural productivity, mitigation of climate change and the discovery of new medicinal plants. However, species recognition has remained a difficult task even for trained botanists, because using the traditional approaches, an expert on a specie may be unfamiliar with others. Thus, researchers and practitioners are increasingly interested in the automation of species recognition problem. Recently, deep learning algorithms such as Convolutional Neural Network (CNN) have provided huge breakthroughs in various computer vision tasks compared to their shallow predecessors. Deep learning automates features extraction by learning salient representations of the data and subsequently classifies the features using a supervised learning approach. Inspired by this capability, we leveraged on five pre-trained CNN models and Leafsnap image dataset of 185 plant species to experimentally evolve an accurate species recognition model in this study. Among the pre-trained models, MobileNetV2 with ADAM optimizer gave the highest testing accuracy of 92.33%. This result provides a basis for developing a mobile app for automated species recognition on the field. This will augment existing efforts to alleviate the difficulties of manual species recognition by botanists, farmers, biologists, nature tourists as well as conservationists.

Keywords: CNN, Leafsnap, MobileNetV2, Optimizer, Plant Species

Introduction

It is estimated that there are more than 450,000 plant species on earth and approximately 385,000 of these have been identified and classified (Pimm and Joppa, 2015). Because of pollution, deforestation and climate change, many plant species have not been discovered yet and are at risk of extinction (Butler, 2020). Due to the shrinking number of botanists that can classify plants and the complexity of plant classification, it is necessary to design a computerized system for plant species identification. Many computer-based systems for plant identification have been designed using leaf analysis, DNA analysis, flower analysis, or a combination of these and other features (Kaur and Kaur, 2019). However, in current literature, leaf analysis remains the most popular approach used to identify a given plant, because leaves are visible and carry enough information to differentiate plant species. Meanwhile, feature extraction and selection are the most challenging and time-consuming steps when using leaf images to recognize plant species. Deep learning techniques came as a solution to achieve automatic feature extraction and selection in computer vision, especially with Convolutional Neural Networks

(CNN). Nonetheless, the selection of the appropriate CNN model for plant classification remains a daunting task. In this study, five state-of-the-art CNN models were employed, to automatically extract the salient features of different plant species in the Leafsnap dataset (Kumar *et al.*, 2012), hence, replacing the need for designing handcrafted features as in previous studies (Hall *et al.*, 2015). The empirical evaluation of these pre-trained CNN models presents an efficient tradeoff between accuracy and latency: MobileNetV2 (154 layers) outperformed other models-AlexNet (25 layers), GoogLeNet (144 layers), VGG-19 (47 layers) and ResNet50 (177 layers) in the classification of plants' leaf images. To build lightweight neural networks, MobileNetV2 uses depth-separable convolutions. It has also been used in past studies for object recognition with promising results thus, encouraging the development of mobile and embedded vision applications (Howard *et al.*, 2017).

The rest of this paper provides a literature review of recent and influential works on the use of machine learning to recognize different species of plant as presented in section 2. Materials and methods are contained in section 3. Section 4 discusses the findings obtained and the discussion of the proposed methods and finally, in section 5, we present the conclusion.

Related Works

The role of plants in medicine, ecosystem maintenance and sustainable agriculture cannot be over-emphasized, as it significantly fosters drug production and provides food for human consumption (Wu *et al.*, 2007). However, plant identification has been a challenging task for botanists, owing to the number of different and uncommon plant species, which exist. In modern research, automation is playing a significant role in solving plant identification problems, using different computational intelligence and deep learning techniques.

Leaf Image Recognition with Neural Networks

Research contributions in pattern recognition and image processing domain over the past decade have greatly influenced the development of models, which reduce the computational complexity and improve the accuracy of object recognition. More importantly, efficient and robust plant recognition systems have been developed to alleviate botanists' efforts in plant species identification.

The systematic review on the techniques for plant leaf recognition conducted by (Azlah *et al.*, 2019) claimed that the traditional use of Artificial Neural Networks (ANNs) in classifying medicinal plant images with 63 leaf image samples, yielded an accuracy of 94.4% (Azlah *et al.*, 2019; Janani and Gopal, 2013). However, this method poses greater computational overhead and data overfitting tendency; showing the possibility of the poor performance of ANN on out-of-sample data, while also discouraging its pragmatic deployment (Azlah *et al.*, 2019). The need for a system capable of identifying large-scale plant species spurred (Wu *et al.*, 2007) to propose an identification system for plants, which extracts twelve morphological and five geometric features based on leaf images' shape and vein structure respectively. In the study, an image dataset named *Flavia* was created, which consists of 1,907 leaf images of 32 different plant species. For the dimension scaling of the input vector, principal component analysis was used before being fed into the three-layered Probabilistic Neural Network (P-NN) for image classification. An average accuracy of 90.32% was obtained and serves as a baseline for studies in the research domain. In the same vein, (Kaur and Kaur, 2019) adopted an image segmentation procedure, using the Swedish dataset consisting of 1,125 images of 15 plant species for plant species identification. Noise handling, with image resizing and enhancement, characterized the pre-processing of the images, while the image texture and colour features were extracted to describe the image. A Multiclass Support Vector Machine (MSVM) classifier was trained for training and testing using 70% and 30% of the image dataset, respectively. Average prediction accuracy of 93.26% was obtained using the MSVM classifier, outperforming the P-NN (Wu *et al.*, 2007). Kaur and Kaur (2019) attributed the lag in the prediction accuracy to the extraction of limited image features.

Further to the counter-performance attributed to the implementation of shallow networks, some authors made significant strides in using deep machine learning for tasks of image classification. Lee *et al.* (2015), employed a CNN model to automatically learn the feature representations of 44 plant species collected at the Royal Botanic Gardens, Kew, England. A visualization technique based on deconvolutional networks was used to identify feature representation. The venation structure was chosen as the primary feature for plant species identification. Two sets of data, D1 (full leaf images) and D2 (cropped leaf images) were passed into the CNN model to check the performance and accuracy of the trained model for each dataset. The classification accuracies of 97.7% and 99.6% were obtained for D1 and D2 respectively, making D2 preferable. Similarly, a 26-layer deep CNN (i.e., ResNet26) with 8 residual building blocks for large-scale identification of plants in the wild was proposed by Sun *et al.* (2017). The model sought to address the challenges, which characterize the traditional handcrafted feature-based classification method. In the research, a total of 10,000 plant images (containing 100 ornamental plant species) of 100 image samples per species, from the Beijing Forestry University campus were collected using a mobile phone. The model parameter was trained using the Stochastic Gradient Descent (SGD) algorithm, with the categorical_crossentropy loss function as the optimization function. The recognition accuracy of ResNet26 was compared with other ResNet models (Bodhwani *et al*., 2019) with results showing 91.78%, 89.27%, 88.28% and 86.15% accuracy for ResNet26, ResNet18, ResNet34 and ResNet50 respectively. It was observed that ResNet26 provided faster and robust convergence among other ResNet models.

In an attempt to differentiate between individual plant species, Lasseck (2017) proposed a Deep CNN (DCNN) technique to classify 10,000 plant species using an image-based identification method. Plant images were retrieved from the Encyclopedia of Life dataset and PlantCLEF 2016 dataset for training. Models of three DCNN architectures - GoogLeNet, ResNet152 and ResNeXT, were used for training and classification of the plant images. To gain diversity across models, the training images were rescaled to various dimensions and random cropping was carried out. The models' performance on the test dataset was evaluated using Mean Reciprocal Rank (MRR), top-1 and top-5 accuracy, with results of 92.0%, 88.5% and 96.2% respectively. The sheer desire for an optimum performance inspired Ghazi *et al*. (2017) to employ DCNNs-AlexNet, GoogLeNet and VGGNet, with the optimization of transfer learning parameters to uniquely classify different plant species. To fine-tune the three pre-trained models, using the LifeCLEF 2015 plant image datasets, the transfer learning approach was adopted. Data augmentation techniques were used to artificially increase the datasets, thereby mitigating the overfitting of the models. An empirical evaluation was carried out on AlexNet, GoogLeNet and VGGNet based on the optimization parameters and dataset, to identify the different factors which affect the performance of each of the deep CNNs. Results from the comparative study showed that VGGNet, GoogLeNet and AlexNet ranked 1st, 2nd and 3rd respectively in the prediction accuracy, with VGGNet having an accuracy of 78.44%. The factors, which affected the performance of the models are the number of iterations used and poor data augmentation methods.

Convolutional Neural Networks

The advancement in image processing, pattern recognition and computer vision has brought about the upsurge in the use of deep neural networks in the field of image classification. Relatively, a lot of shallow neural network classifiers have been reported to pose problems of data overfitting, which greatly affects how objects or images are classified (Janani and Gopal, 2013; Sweetwilliams *et al*., 2019; Alaba *et al*., 2020; Akanle *et al*., 2020), hence, the advent of CNN to mitigate the vanishing gradient problem. A CNN is a supervised deep learning method that employs the use of convolution mathematics, which is the process of implementing a 2-D convolution with a filter on an input image, in at least one of the network layers. A deep CNN consists of an input layer that contains image data of *m* training examples, multiple hidden layers that compute features from input images and an output layer, which classifies the learned images. Deep learning models employ non-linear transformation functions to solve complex large-scale problems (Reyes *et al*., 2015; Li *et al*., 2016; Badejo *et al*., 2018). As shown in Figure 1, the hidden layers consist of stacked convolution layers that convolve using a Rectified Linear Unit (ReLU) activation (or transfer) function, as well as a pooling layer, which reduces the dimension of the convoluted image. The Fully Connected (FC) layer connects each input to the next layer (classification layer) from the previous layer. Significant achievements have been made through the application of deep learning models in image processing, natural language processing and speech recognition tasks, thereby paving the way for more predictive analysis of big data (Li *et al*., 2016). The description of the aforementioned layers can be mathematically represented as shown in Equation (1)-(7):

$$X \in \Re^{H\times W\times D} \tag{1}$$

A colour (RGB) image X is a three-dimensional matrix of H rows (height), W columns (width) and D channels (depth) denoted as $H \times W \times D$. For a colour image, $D = 3$, which stores each of the Red, Green and Blue pixel (px) value in the range $0 < px < 255$. The RGB image is fed into the first convolution layer, which extracts salient image features. The output of each convolution layer gives a feature map (G), obtained from the kernel computation:

$$H_n = \frac{H_m - F + 2P}{S+1} \tag{2}$$

$$W_n = \frac{W_m - F + 2P}{S+1} \tag{3}$$

$$D_n = K \tag{4}$$

Where:

H_m	=	Previous layer's image height
H_n	=	Current convolution layer's output image height
W_m	=	Previous layer's image width
W_n	=	Current convolution layer's output image width
F	=	Filter, K = number of filters, P = padding and S = stride.

The feature map $G = H_n \times W_n \times D_n$ passes through the Rectified Linear Unit (ReLU) activation function, which normalizes and converts all negative values to zero. It is expressed as:

$$f(Z) = \max(0, Z) = \begin{cases} Z_i & if\ Z_i \geq 0 \\ 0 & if\ Z_i < 0 \end{cases} \tag{5}$$

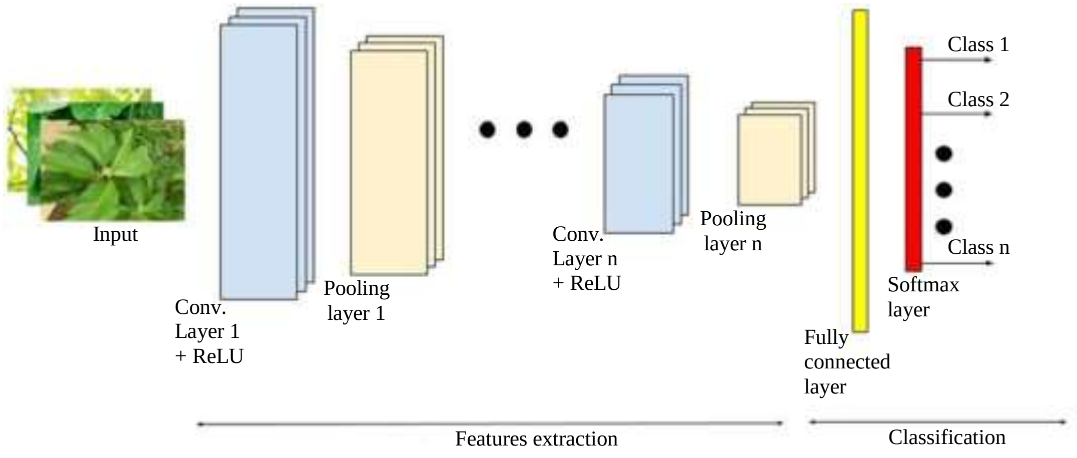


Fig. 1: Deep CNN architecture

A pooling layer, which performs non-linear down sampling of the feature map reduces the dimension of the convoluted image. There are two options of pooling - max and average pooling. Max pooling uses the maximum computed feature map value, while the average pooling uses the average. The Fully Connected (FC) layer, which is the last learning phase, maps the extracted features from the previous layers to the output labels by summing the weighted inputs of each layer:

$$G_j^{[l]} = \sum_{j=1}^{n^{[i]}} W_j^{[l]} a_j^{[l-1]} + b^{[l]} \tag{6}$$

$$a_j^{[l]} = \sigma_j^{[l]}\left(G_j^{[l]}\right) \tag{7}$$

Where:

$G_j^{[l]}$	= The sum of weighted inputs for the neurons in each layer
$a_j^{[l]}$	= The activation of the l^{th} layer's weighted inputs
$W_j^{[l]} a_j^{[l-1]}$	= The element-wise product of each layer's neuron weight and the previous layer's activation
$b^{[l]}$	= The updated bias in each layer
$\sigma_j^{[l]}$	= Represents the activation function for each of the neurons j in layer *l*

Lastly, a softmax layer, which performs multiclass classification using discrete probability distribution (Jiang *et al*., 2020) generates the predicted corresponding output label (class) for each feature map, with the highest probability being the prediction target as shown in Equation (8):

$$P\left(Y_i\right) = \frac{\exp\left(X_i\right)}{\sum_{j=1}^{K} \exp\left(X_j\right)} \tag{8}$$

Whilst most studies (Kaur and Kaur, 2019; Wu *et al*., 2007; Lee *et al*., 2015; Bodhwani *et al*., 2019) addressed plant identification from a single neural network perspective using plant datasets, other studies (Sun *et al*., 2017; Lasseck, 2017; Ghazi *et al*., 2017) carried out a comparative analysis on the performance of at most three deep neural network models. However, it is imperative to evaluate the performance of a number of the state-of-the-art deep CNN models using rich and robust datasets, while demystifying the factors affecting each model for optimum prediction accuracy. Hence, this paper addresses the problem of using heuristic methods in plant species identification through rigorous experimentation of pre-trained deep CNN models, which include AlexNet, GoogLeNet, VGG-19, ResNet50 and MobileNetV2 for the recognition of plant species based on the Leafsnap image dataset.

Pre-Trained CNN Models

The robust application of deep learning in several problem domains has yielded commercial success, owing to the paradigm shift from traditional Machine Learning (ML) methods to Transfer Learning (TL). In contrast to traditional learning where knowledge for a trained model is not retained, transfer learning allows the learning of new tasks from the knowledge (such as features, weights and hyperparameters) of previous tasks, thereby making the learning process of a model faster, efficient and accurate (Li *et al*., 2017). Figure 2 shows the difference between the traditional ML and TL.

The pre-trained CNN models used in this study have been trained on a large dataset containing thousands of classes and millions of samples and are hereafter succinctly described.

a) AlexNet

The AlexNet, which was trained to identify about 1.2 million high-resolution images of 1,000 classes from the

ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) dataset, was developed by (Krizhevsky *et al*., 2017). The neural network consists of 650,000 neurons and 60 million parameters, including five convolutional layers, two normalization layers, three max-pooling layers, three fully connected layers and one softmax layer, with the ReLU activation function applied between the convolutional layers and fully connected layers. AlexNet (Krizhevsky *et al*., 2017) uses a dropout regularization method (Srivastava *et al*., 2014) to reduce overfitting of the model and it is renowned for winning the ILSVRC 2012. The image input size, $H \times W \times D$, of the network is 227×227×3.

b) GoogLeNet

Szegedy *et al*. (2015) developed the GoogLeNet - an inception architecture, which combines multi-scale processing and dimension reduction. GoogLeNet, because of its reputation, won the ILSVRC 2014 competition, as it enables an increment in the depth and width of the network, towards an improved generalization. It possesses 6.8 million parameters from nine inception modules, two convolutional layers, four max-pooling layers, one average pooling layer, two fully connected layers and a softmax layer, with the ReLU activation function applied in all the convolutional layers. The image input size, $H \times W \times D$, of the network is 224×224×3.

c) VGG-19

The ILSVRC 2014 challenge paved the way for several deep learning models to emerge, such as the VGGNet, which investigates the effects of depth increment on the performance of a convolutional network, using a homogeneous architecture. Developed by (Simonyan and Zisserman, 2014), VGG-19, a variant of the VGGNet, consists of 144 million parameters from 16 convolutional layers, five max-pooling layers, three fully connected layers and a softmax layer in the output. The image input size, $H \times W \times D$ of the network is 224×224×3. According to Ghazi *et al*. (2017), the VGGNet provides optimum performance in transfer learning tasks but poses computational complexities owing to a large number of inherent parameters compared to AlexNet (Krizhevsky *et al*., 2017) and GoogLeNet (Srivastava *et al*., 2014).

d) ResNet50

He *et al*. (2016) presented a ResNet learning model, referred to as *deep residual nets*, to address the degradation problem with deeper networks. ResNet50, a residual network with 50 layers employs the *"shortcut connection concept"* by explicitly letting the stacked layers fit a residual mapping $\mathcal{F}\left(x,\{W_i\}\right)$, rather than stacking each layer directly. This is shown in Equation (9), which describes the stacked layers' approximation of a residual function. Equation (10) and (11) show the shortcut connections, where x, y and (a) are the input vectors, output vectors and ReLU function respectively of the layers considered (Sun *et al*., 2017). The residual network with 50 layers, consists of 48 convolutional layers, a max-pooling layer and a fully connected layer and has an image input size $H \times W \times D$ of 224×224×3:

$$\mathcal{F}(x) := \mathcal{H}(x) - x \tag{9}$$

$$y = \mathcal{F}\left(x,\{W_i\}\right) + x \tag{10}$$

$$\begin{aligned} F &= W_2\,\sigma(W_i\,x), \\ \sigma(a) &= \max(0,a) \end{aligned} \tag{11}$$

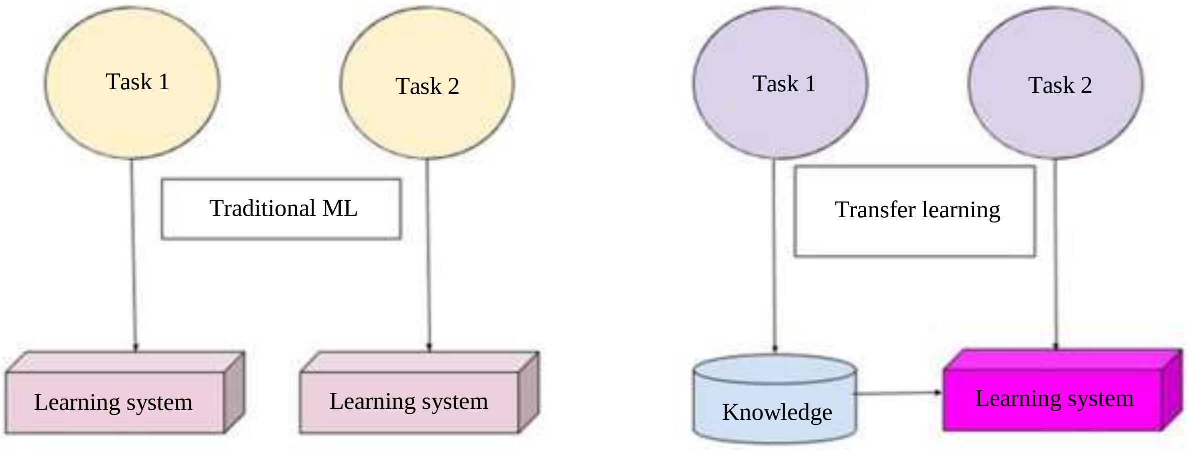


Fig. 2: Traditional ML vs Transfer Learning (Li *et al*., 2017)

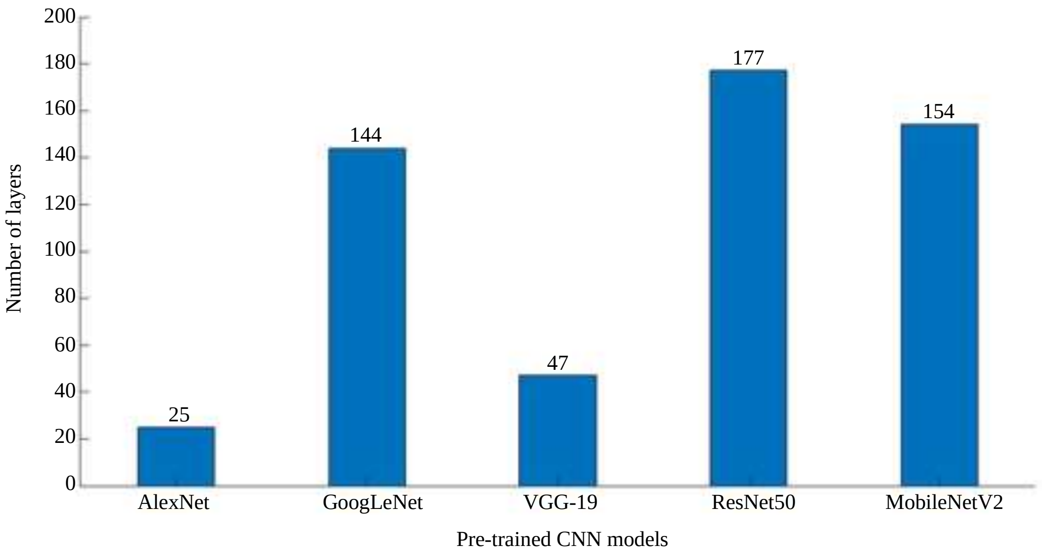


Fig. 3: Visualization of the pre-trained deep CNN models

e) MobileNetV2

Sandler *et al.* (2018) developed the MobileNetV2 architecture for object detection. MobileNetV2 is an improvement over MobileNetV1 (Howard *et al*., 2017), as it uses depthwise separable convolution as efficient building blocks. For mobile applications, the model focuses on multiple image classification tasks and contains the initial convolution layer with 32 filters and 19 residual bottleneck layers. The model has an image input size $H \times W \times D$ of 224×224×3 and uses 3.4 million parameters for training. Figure 3 illustrates all the CNN models evaluated in this study and their corresponding depths (layers).

Research Methodology

CNNs have been known for successfully demystifying image classification and object identification problems. They are widely utilized for training on extensive datasets while using high-performance computing hardware including the Graphical Processing Unit (GPU) and/or Tensor Processing Unit (TPU). This study uses the Leafsnap dataset in the training of five state-of-the-art deep CNNs-AlexNet, GoogLeNet, VGG-19, ResNet50 and MobileNetV2 for image classification. This section discusses image acquisition, pre-processing and how transfer learning, using pre-trained CNN models, was employed for plant species recognition. The pattern recognition and image classification techniques employed by the CNN models are also explained. All the experiments were conducted using the MATLAB R2020a software on an Ubuntu 18.04.4 Linux server with a 2.00 GHz Intel Xeon(R) CPU (7.7 GB memory) and a GeForce GT 650 M GPU (12 GB memory). Figure 4 shows the block diagram that graphically illustrates the methodology in this study.

Data Acquisition

The first step in the process of plant species recognition research is the collection of plant images. Intuitively for plant species recognition, images of leaf, flower, stem, or fruits can be used to distinguish each plant (Kaur and Kaur, 2019). However, in most studies, leaf images have been used for plant species recognition, as deep neural network models uniquely extract salient features from each leaf image. For the large-scale recognition of plants, the Leafsnap dataset, which consists of 185 plant species, was used in this study to train the CNN models. The plant species are mainly found in the Northeastern United States and acquired in Columbia University, the University of Maryland and the Smithsonian Institution (Kumar *et al*., 2012). The dataset contains at least 7,500 field images captured by mobile devices (iPads and iPhones) in the outdoor environment, with varying levels of noise, blur, shadows and illumination patterns. Figure 5 shows some of the samples from the dataset. To build a visual recognition system for automatic plant species identification, the robust Leafsnap dataset was used (Kumar *et al*., 2012).

Pre-Processing

Data pre-processing in deep learning, as elicited in this study, enhances the Leafsnap dataset before each model is trained on it. First, the images were resized to match the input layer dimension for each model, after which training and validation were carried out on the image datasets. Data augmentation, which is a technique for creating diverse image variations to increase the size of the datasets, was employed to artificially create variations in the existing image dataset. This was carried out by configuring the image augmentation options for deep learning. Specifically, the properties of the image data augmenter object were set to randomly translate the images up to 30 pixels horizontally and vertically. Other image transformation methods include scaling, flipping, cropping, rotation and padding. These allow each model to train on a larger image dataset.

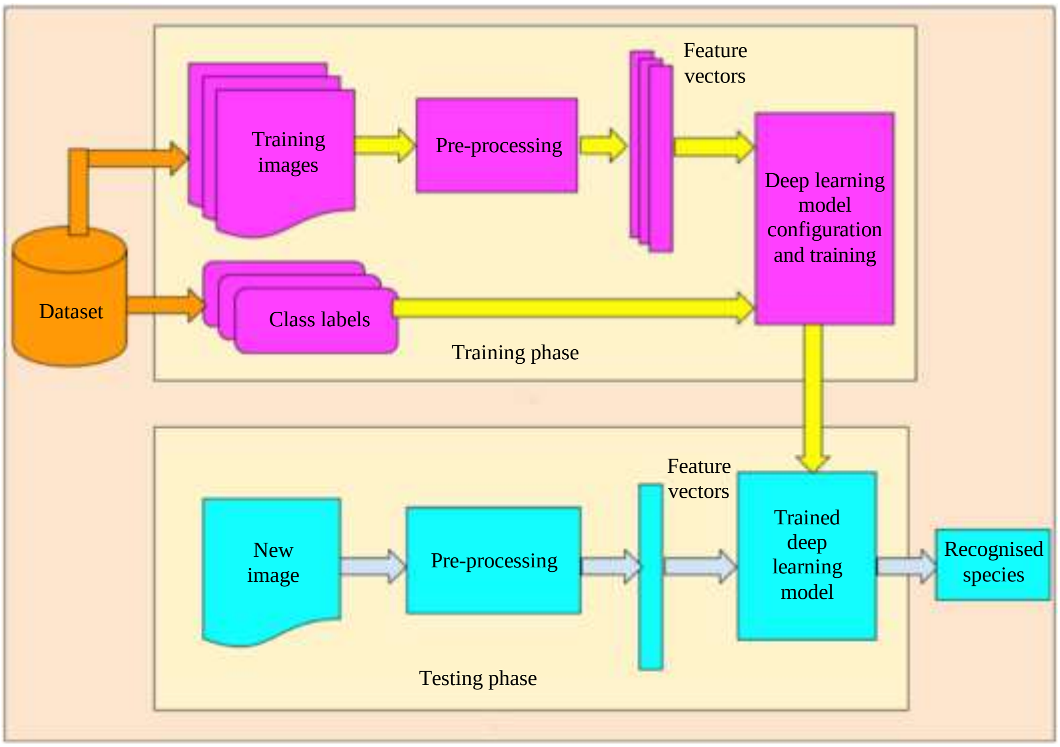


Fig. 4: Block diagram of leaf images recognition using deep learning (CNN)

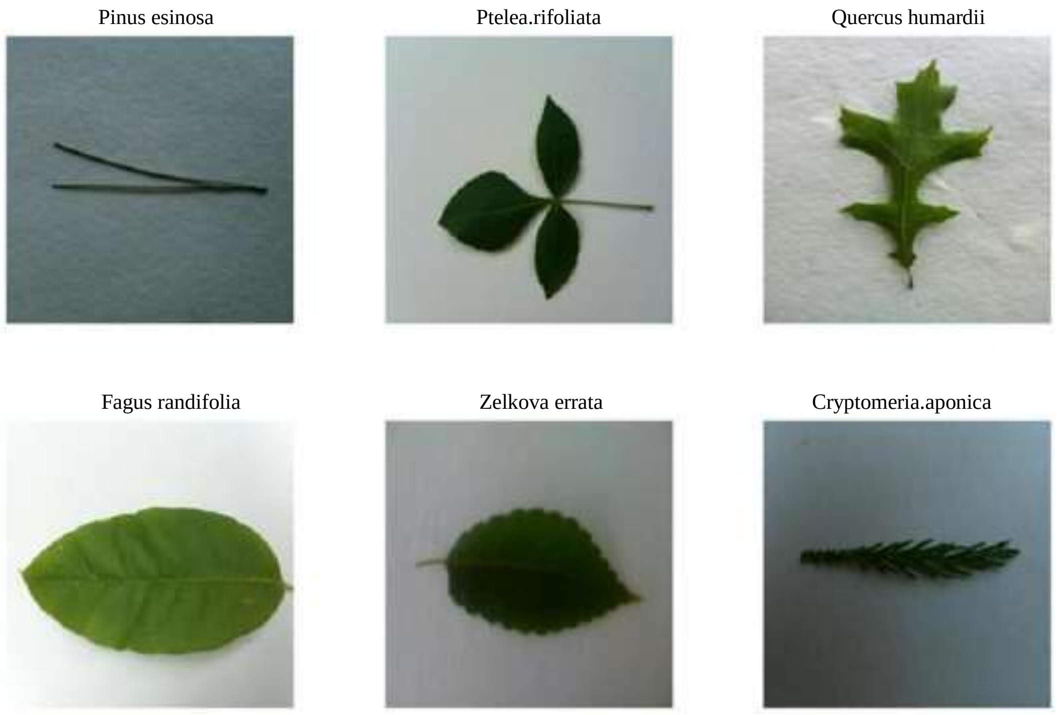


Fig. 5: Sample images of the LeafSnap dataset

CNN Model Development

Training a machine learning model (Fig. 4) requires bridging the gap between the observed training data labels and the prediction of the model. Through interactive visualizations, the model parameters (weights and biases) were defined to minimize the cost function J over the entire training dataset for optimum model performance as shown in Equation (12):

$$\begin{aligned} \mathcal{L}^{(i)} &= \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^z \\ J &= \frac{1}{m_b}\sum_{i=1}^{m_b}\mathcal{L}^{(i)} \end{aligned} \tag{12}$$

Where:

m_b = The size of the training data referred to as the mini-batch size for each iteration,
$L^{(i)}$ = The loss obtained for a single training example $x^{(i)}$ labeled $y^{(i)}$.

However, it is expedient to choose appropriate values of hyperparameters such as the learning rate α and batch size m_b, to reduce the cost function and accelerate optimization while enabling model convergence to the global minima. This is critical as the selection of a small or large learning rate leads to slow convergence or overshooting respectively (Katanforoosh *et al*., 2020). Notably, the batch size, which is the number of samples used to train the models in each iteration, influences the convergence of the cost function. For improved training accuracy and quicker updates, the mini-batch size was set to a small value, as larger values could cause the models to overfit instead of generalizing. Also, the training was terminated at 100 epochs, when the validation loss stops decreasing and before the models lose generalization capacity. Table 1 shows the configuration parameters and network training options used for the experiments in this study.

Whilst the learning rate and mini-batch size significantly influence the efficiency of CNN models, the choice of the optimization algorithms is also crucial as they make it possible for the deep neural network models to learn or train with the dataset. However, the speed of convergence given an optimization algorithm is a factor, which determines the performance of models. To evaluate the performance of the CNN models selected for this study, three optimizers, namely-Stochastic Gradient Descent with Momentum (SGDM), Adaptive Moment (ADAM) estimation and Root Mean Square Propagation (RMSProp) were used to compute the neurons' activation (Katanforoosh *et al*., 2020). The update rule, as shown in Table 2, shows how the gradients are calculated for each model.

Table 1: Network configuration parameters

Parameter	Value
Batch size	10
Learning rate	0.0001
Epochs	100
Validation frequency	30
Execution environment	GPU

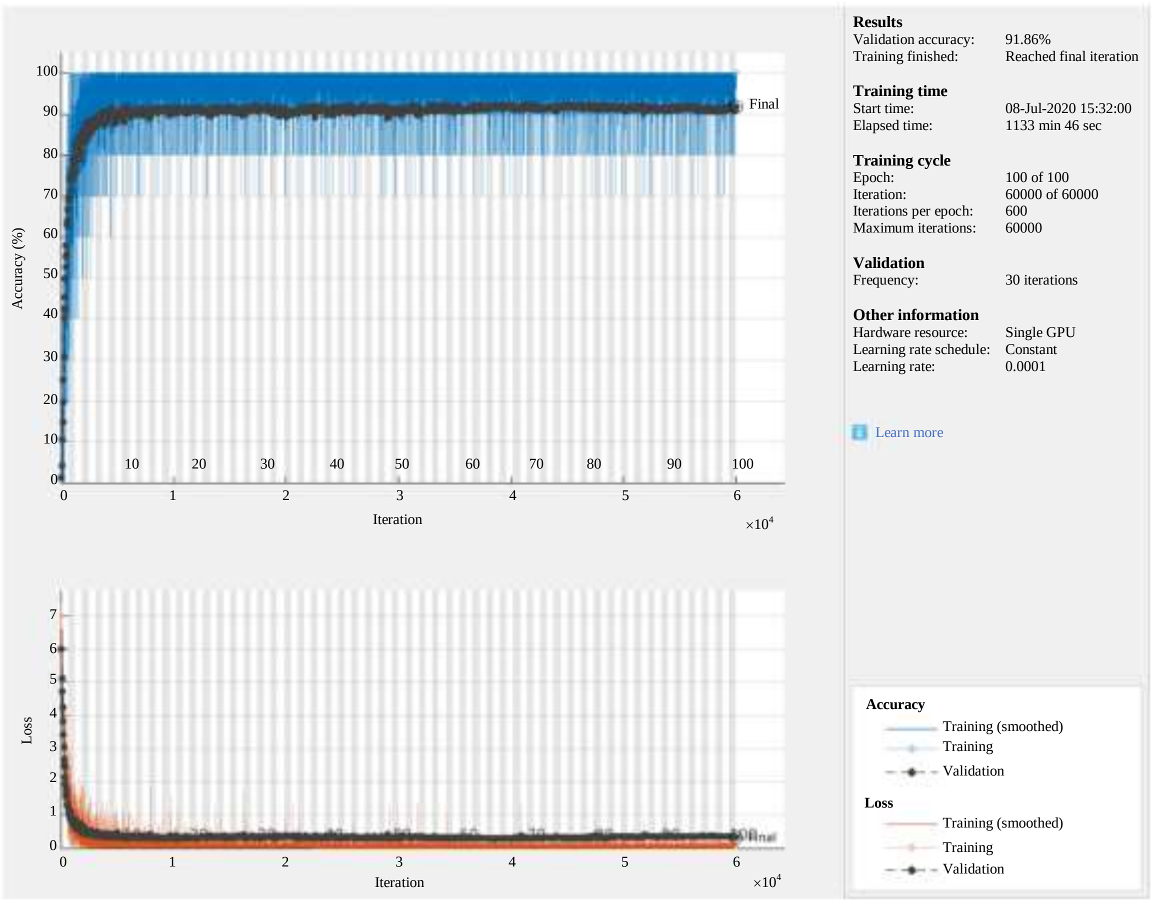
Table 2: Definition and characteristics of optimization algorithms

Algorithm	Update rule	Characteristics
SGDM	$S_{dW} = \beta S_{dW} + (1-\beta)dW$	a) It requires applying exponential smoothing to the computed gradient.
	$W = W - \alpha S_{dW}$	b) It performs better than conventional gradient descent.
		c) It uses more memory than ADAM and RMSProp and requires hyperparameter tuning.
ADAM	$S_{dW} = \beta_1 S_{dW} + (1-\beta_1)dW$	a) The hyperparameters are predefined and require no tuning.
	$V_{dW} = \beta_2 S_{dW} + (1-\beta_2)dW^2$	b) It uses more memory for specified batch size.
	$Scorr_{dW} = \frac{S_{dW}}{(1-\beta_1)^t}$	c) It is mostly the default optimizer used in machine learning.
	$Vcorr_{dW} = \frac{V_{dW}}{(1-\beta_2)^t}$	
	$W = W - \alpha\frac{Scorr_{dW}}{\sqrt{Vcorr_{dW} + \varepsilon}}$	
RMSPROP	$V_{dW} = \beta V_{dW} + (1-\beta)dW^2$	a) It uses more memory for a given batch size than SGDM, but less than ADAM.
	$W = W - \alpha\frac{dW}{\sqrt{V_{dW} + \varepsilon}}$	b) It normalizes the impact of the learning rate decay.
		c) It maintains per-parameter learning rates.

IN this research, the first experiment was carried out to train the five deep CNN models (AlexNet, GoogLeNet, VGGNet, ResNet50 and MobileNetV2) using the original Leafsnap dataset (non-augmented), while the second experiment was carried out using the augmented Leafsnap dataset for the training of the models for improved generalization. The dataset was split into 8:2 ratios for training (i.e., 6000 images) and testing (i.e., 1500 images) respectively.

Results and Discussion

In this section, the performance assessment of the CNN models trained on the Leafsnap dataset is presented, while identifying the model with the best tradeoff between the validation accuracy and training time. Each model was trained at 100 epochs with 600 iterations per epoch. Table 3 shows the performance comparison between the pre-trained CNN models examined in this study for the first experiment using the non-augmented Leafsnap dataset. Results from the experiment show that VGG-19 with SGDM and MobileNetV2 with ADAM gave the best performance, each with a validation accuracy of 91.86%, while GoogLeNet with RMSProp gave the worst performance with a validation accuracy of 4.00%. However, the training time shows that VGG-19 trains faster on the Leafsnap dataset than MobileNetV2. Figures 6 and 7 show the evolution of the validation accuracy for each of the best performing models on the non-augmented dataset respectively. For the second experiment, Table 4 shows the performance result of the pre-trained CNN models using the augmented Leafsnap dataset. Results from the experiment show that MobileNetV2 with ADAM and ResNet50 with SGDM gave the best performance and second-best validation accuracies of 92.33% and 92.13% respectively, while GoogLeNet with RMSProp gave the poorest performance with a validation accuracy of 1.73%. Figures 8 and 9 show the evolution of the validation accuracy for MobileNetV2 and ResNet50 on the augmented dataset respectively. Also, it is noteworthy that the depth of the CNN models, as shown in Fig. 3 determines the overall training time as illustrated further in Fig. 10 and 11. This observation is critical because it will help machine learning experts to make informed decisions on model selection for classification tasks while considering the computation time as a significant factor.



Results
Validation accuracy: 91.86%
Training finished: Reached final iteration

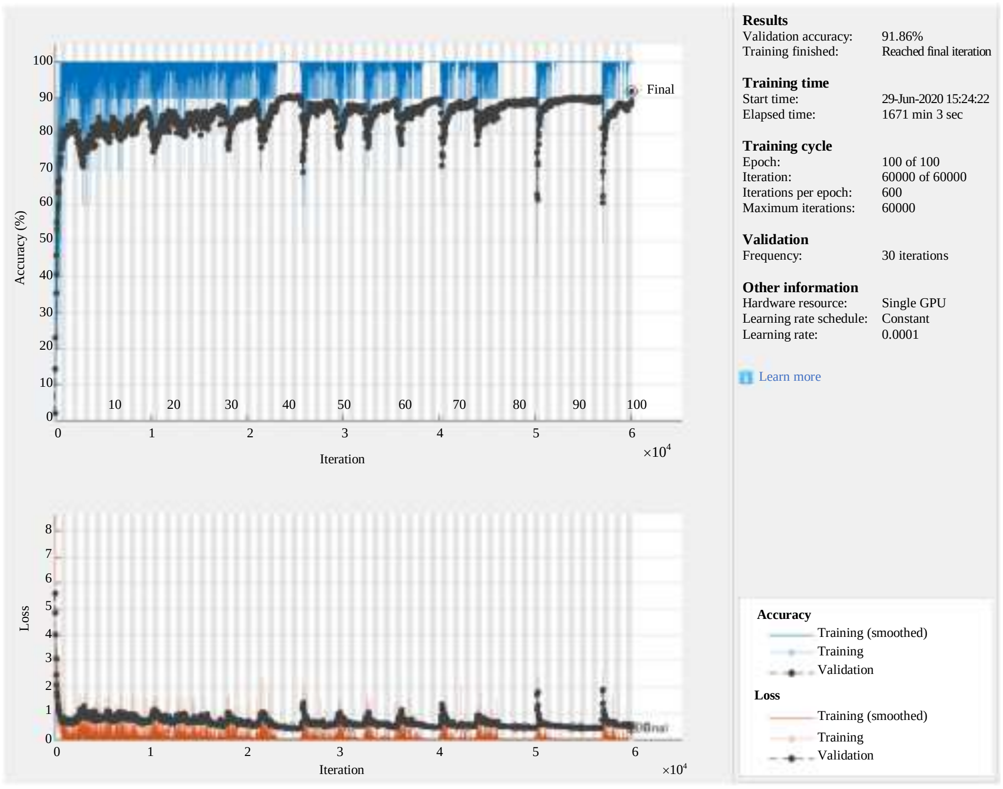
Training time
Start time: 08-Jul-2020 15:32:00
Elapsed time: 1133 min 46 sec

Training cycle
Epoch: 100 of 100
Iteration: 60000 of 60000
Iterations per epoch: 600
Maximum iterations: 60000

Validation
Frequency: 30 iterations

Other information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0001

Fig. 6: Validation Accuracy using VGG-19 with SGDM for Non-Augmented Dataset



Results
Validation accuracy: 91.86%
Training finished: Reached final iteration

Training time
Start time: 29-Jun-2020 15:24:22
Elapsed time: 1671 min 3 sec

Training cycle
Epoch: 100 of 100
Iteration: 60000 of 60000
Iterations per epoch: 600
Maximum iterations: 60000

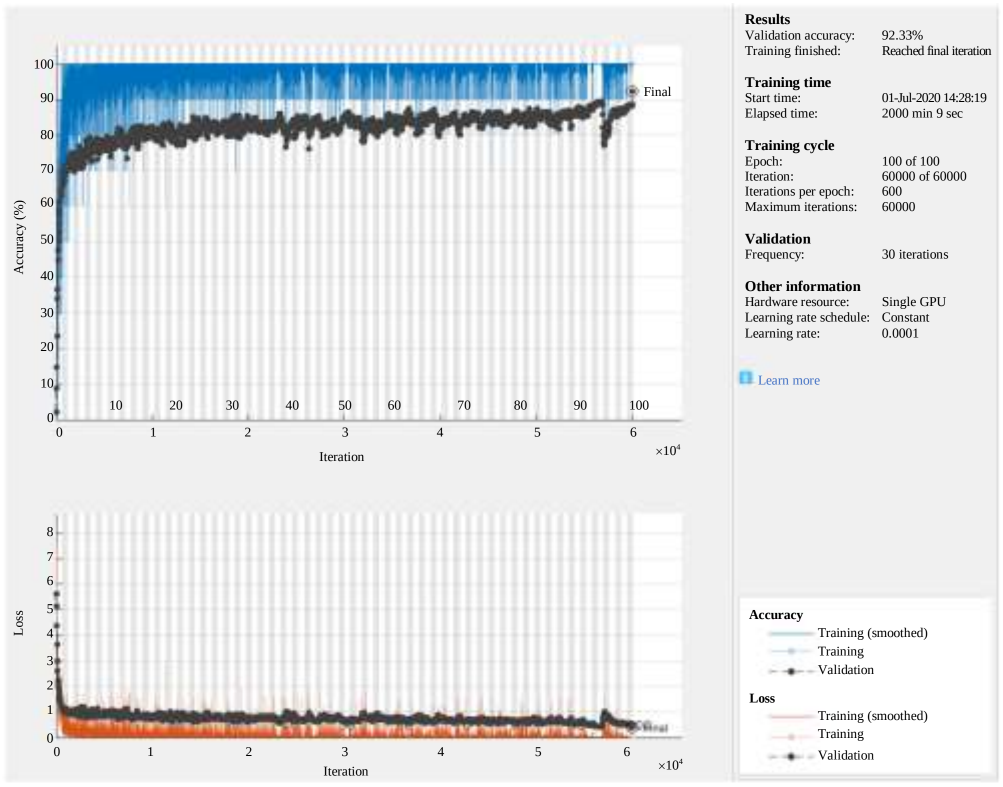
Validation
Frequency: 30 iterations

Other information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0001

Accuracy
Training (smoothed)
Training
Validation

Loss
Training (smoothed)
Training
Validation

Fig. 7: Validation accuracy using MobileNetV2 with ADAM for non-augmented dataset



Results
Validation accuracy: 92.33%
Training finished: Reached final iteration

Training time
Start time: 01-Jul-2020 14:28:19
Elapsed time: 2000 min 9 sec

Training cycle
Epoch: 100 of 100
Iteration: 60000 of 60000
Iterations per epoch: 600
Maximum iterations: 60000

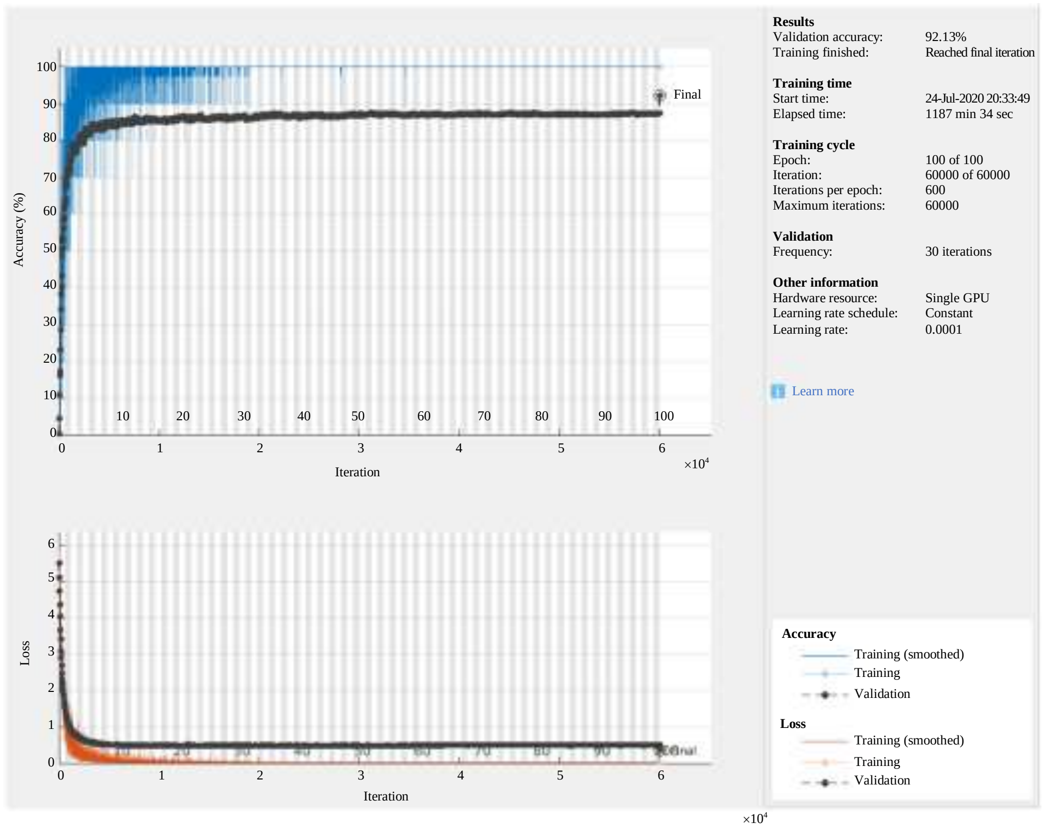
Validation
Frequency: 30 iterations

Other information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.0001

Accuracy
Training (smoothed)
Training
Validation

Loss
Training (smoothed)
Training
Validation

Fig. 8: Validation accuracy using MobileNetV2 with ADAM for augmented dataset



Results

Validation accuracy:	92.13%
Training finished:	Reached final iteration

Training time

Start time:	24-Jul-2020 20:33:49
Elapsed time:	1187 min 34 sec

Training cycle

Epoch:	100 of 100
Iteration:	60000 of 60000
Iterations per epoch:	600
Maximum iterations:	60000

Validation

Frequency:	30 iterations

Other information

Hardware resource:	Single GPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Fig. 9: Validation accuracy using ResNet50 with SGDM for augmented dataset

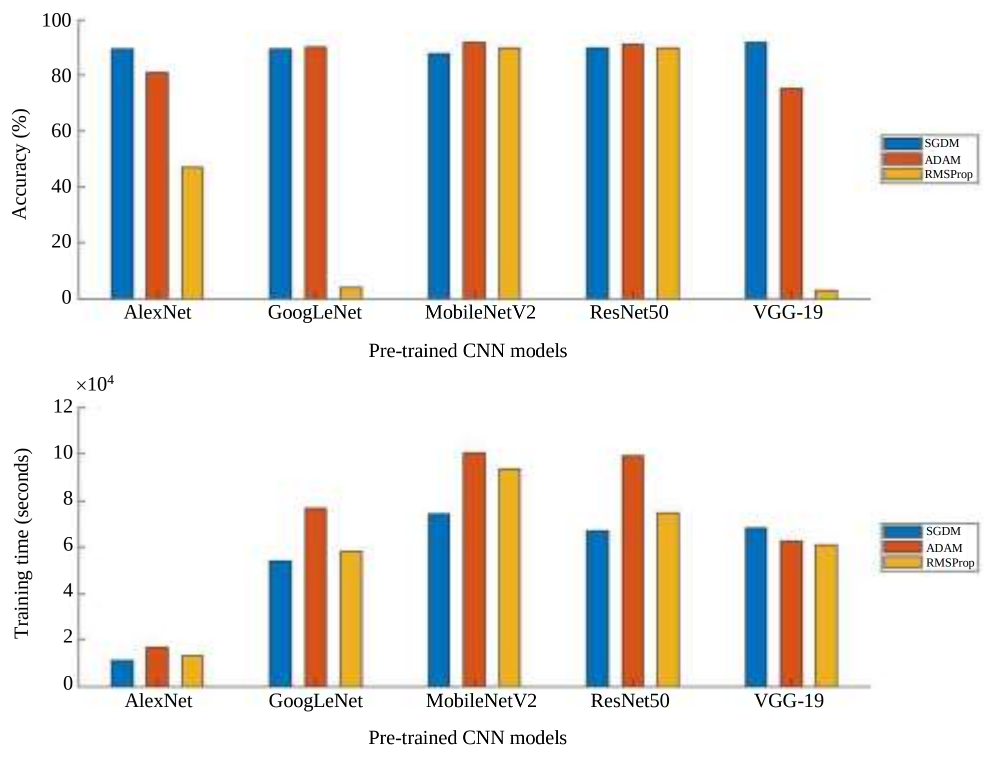


Fig. 10: First experiment without data augmentation

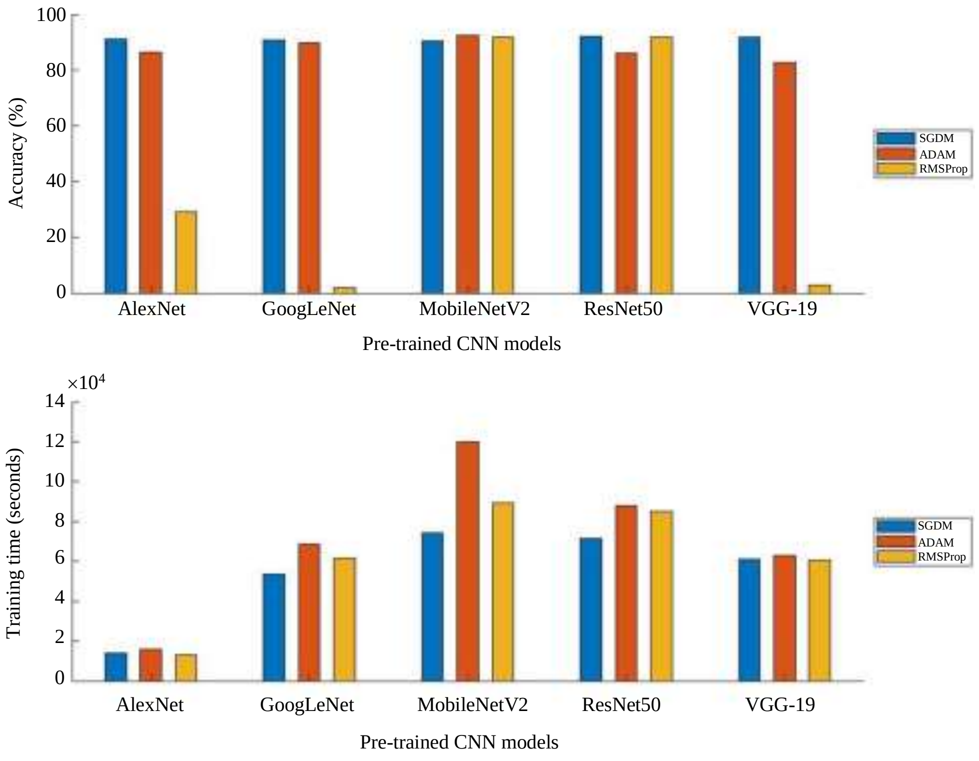


Fig. 11: Second experiment with data augmentation

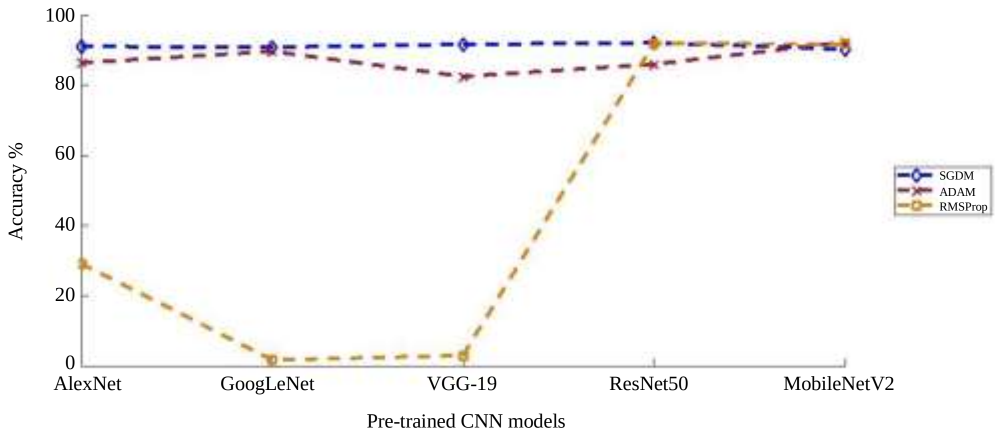


Fig. 12: Comparison plot of ADAM, RMSProp and SGDM

Notably, in this study, the intrinsic characteristics of the optimizers greatly influenced the models' performance. Comparing the training algorithms give a holistic perception of the strengths and weaknesses of each, using the Leafsnap dataset and serves as a precursor in choosing the best optimizer for a similar dataset. As shown in Fig. 12, it can be observed that the SGDM algorithm gave the highest accuracy for all, but one of the models, while the RMSProp optimizer performed poorly on three of the models. Also, SGDM possesses excellent stability for training with the dataset. However, in this study, the overall best performing model - MobileNetV2, gave the best accuracy of 92.33% using ADAM as the training algorithm and differs from using SGDM by a margin of 2.07%.

Our experimental exploration of pre-trained CNN models for recognition of plant species using the Leafsnap image dataset has strengthened the viability of employing deep machine learning techniques in several biological domains ranging from botany to medicine. Also, in this study, we have compared various pre-trained CNN models using the transfer learning method and different optimizers. Notably, a huge number of images (7,500) in the Leafsnap dataset was captured on the field with mobile devices. Thus, it became apparent while MobileNetV2 gave the overall best accuracy since it was originally optimised for multiple image classification tasks on mobile applications (Kumar *et al*., 2012; Sandler *et al*., 2018). To corroborate the findings in this study, previous studies have also reported that MobileNetV2 is efficient on various recognition tasks such as object detection, landmark recognition and facial recognition (Howard *et al*., 2017; Uchida, 2020).

Table 3: Performance result of the original Leafsnap dataset (without augmentation) using pre-trained CNN models

CNN model	Optimizer	Accuracy (%)	Training time (seconds)	Rank
AlexNet	SGDM	89.33	11,111	8th
	ADAM	80.92	16,826	11th
	RMSProp	46.96	13,112	13th
GoogLeNet	SGDM	89.33	53,886	8th
	ADAM	89.99	76,620	4th
	RMSProp	4.00	57,953	14th
VGG-19	**SGDM**	**91.86**	**68,026**	**1st**
	ADAM	75.38	62,290	12th
	RMSProp	2.94	60,929	15th
ResNet50	SGDM	89.59	66,786	7th
	ADAM	90.99	99,055	3rd
	RMSProp	89.73	74,414	6th
MobileNetV2	SGDM	87.79	74,164	10th
	ADAM	**91.86**	**100,263**	**2nd**
	RMSProp	89.79	93,338	5th

Table 4: Performance result of the augmented Leafsnap dataset using pre-trained CNN models

CNN model	Optimizer	Accuracy (%)	Training time (seconds)	Rank
AlexNet	SGDM	91.06	13,708	6th
	ADAM	86.39	15,868	10th
	RMSProp	29.22	12,771	13th
GoogLeNet	SGDM	90.86	53,304	7th
	ADAM	89.66	68,764	9th
	RMSProp	1.73	61,602	15th
VGG-19	SGDM	91.66	60,878	5th
	ADAM	82.52	62,790	12th
	RMSProp	2.94	60,613	14th
ResNet50	**SGDM**	**92.13**	**71,254**	**2nd**
	ADAM	85.99	87,778	11th
	RMSProp	91.86	84,879	3rd
MobileNetV2	SGDM	90.26	74,403	8th
	ADAM	**92.33**	**120,009**	**1st**
	RMSProp	91.79	89,221	4th

Conclusion

In this study, we have presented the development of automated plant species identification model through rigorous experimentations with five state-of-the-art pre-trained CNN models (i.e., AlexNet, GoogLeNet, VGG-19, ResNet50 and MobileNetV2) and the Leafsnap plant image dataset. Our experimental results showed that MobileNetV2 gave the best-fit model when tuned with ADAM optimizer by posting an accuracy of 92.33%. This provides a baseline for further studies in the application of deep learning and leaf images for plant species identification. In the future, we plan to improve the accuracy of the model and adapt it to indigenous plant species in African countries. Ultimately, we hope to develop a mobile app or customized portable digital device based on this effort, to ease the task of plant species identification among botanical and allied professionals.

Acknowledgment

This work was carried out at the Advanced Signal Processing and Machine Intelligence Research (ASPMIR) laboratory, IoT Enabled Smart and Connected Community Research Cluster, Covenant University. The High-Performance Computing node of the Covenant Applied Informatics and Communication African Centre of Excellence (CApIC-ACE) FEDGEN Testbed was utilized for experimentations. The Covenant University Centre for Research, Innovation and Development (CUCRID), Covenant University, Ota, Nigeria provided full sponsorship for the publication of this work.

Author's Contributions

All the authors have equal contributions to the completion of the manuscript.

Ethics

The authors confirm that this article is original and has not been published in any other journal. This article has no ethical issues and no conflict of interest to disclose.

References

Akanle, M., Adetiba, E., Akande, V., Akinrinmade, A., Ajala, S., Moninuola, F., Badejo, J. & Adebiyi, E. (2020, August). Experimentations with OpenStack System Logs and Support Vector Machine for an Anomaly Detection Model in a Private Cloud Infrastructure. In 2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD) (pp. 1-7). IEEE. https://doi.org/10.1109/icABCD49160.2020.9183878

Alaba, P. A., Popoola, S. I., Abnisal, F., Lee, C. S., Ohunakin, O. S., Adetiba, E., Akanle, M. B., Patah, M. F. A., Atayero, A. A. & Daud, W. M. A. W. (2020). Thermal decomposition of rice husk: a comprehensive artificial intelligence predictive model. Journal of Thermal Analysis and Calorimetry, 140(4), 1811-1823. https://doi.org/10.1007/s10973-019-08915-0

Azlah, M. A. F., Chua, L. S., Rahmad, F. R., Abdullah, F. I., & Wan Alwi, S. R. (2019). Review on Techniques for Plant Leaf Classification and Recognition. Computers, 8(4), 77. https://doi.org/10.3390/computers8040077

Badejo, J. A., Adetiba, E., Akinrinmade, A., & Akanle, M. B. (2018, April). Medical image classification with hand-designed or machine-designed texture descriptors: a performance evaluation. In International Conference on Bioinformatics and Biomedical Engineering (pp. 266-275). Springer, Cham. https://doi.org/10.1007/978-3-319-78759-6_25

Bodhwani, V., Acharjya, D. P., & Bodhwani, U. (2019). Deep residual networks for plant identification. Procedia Computer Science, 152, 186-194. https://doi.org/10.1016/j.procs.2019.05.042

Butler, R. A. (2020). Consequences of Deforestation. https://rainforests.mongabay.com/09-consequences-of-deforestation.html

Ghazi, M. M., Yanikoglu, B., & Aptoula, E. (2017). Plant identification using deep neural networks via optimization of transfer learning parameters. Neurocomputing, 235, 228-235. https://doi.org/10.1016/j.neucom.2017.01.018

Hall, D., McCool, C., Dayoub, F., Sunderhauf, N., & Upcroft, B. (2015, January). Evaluation of features for leaf classification in challenging conditions. In 2015 IEEE Winter Conference on Applications of Computer Vision (pp. 797-804). IEEE. https://doi.org/10.1109/WACV.2015.111

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778). https://doi.org/10.1109/CVPR.2016.90

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861. https://arxiv.org/abs/1704.04861

Janani, R., & Gopal, A. (2013, September). Identification of selected medicinal plant leaves using image features and ANN. In 2013 international conference on advanced electronic systems (ICAES) (pp. 238-242). IEEE. https://doi.org/10.1109/ICAES.2013.6659400

Jiang, X., Hu, B., Chandra Satapathy, S., Wang, S. H., & Zhang, Y. D. (2020). Fingerspelling Identification for Chinese Sign Language via AlexNet-Based Transfer Learning and Adam Optimizer. Scientific Programming, 2020. https://doi.org/10.1155/2020/3291426

Katanforoosh, K., Kunin, D. & Ma, J. (2020). Parameter optimization in neural networks. https://www.deeplearning.ai/ai-notes/optimization/

Kaur, S., & Kaur, P. (2019). Plant species identification based on plant leaf using computer vision and machine learning techniques. Journal of Multimedia Information System, 6(2), 49-60. https://doi.org/10.33851/JMIS.2019.6.2.49

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90. https://doi.org/10.1145/3065386

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I. C., & Soares, J. V. (2012, October). Leafsnap: A computer vision system for automatic plant species identification. In European conference on computer vision (pp. 502-516). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-33709-3_36

Lasseck, M. (2017, September). Image-based Plant Species Identification with Deep Convolutional Neural Networks. In CLEF (Working Notes).

Lee, S. H., Chan, C. S., Wilkin, P., & Remagnino, P. (2015, September). Deep-plant: Plant identification with convolutional neural networks. In 2015 IEEE international conference on image processing (ICIP) (pp. 452-456). IEEE. https://doi.org/10.1109/ICIP.2015.7350839

Li, X., Pang, T., Xiong, B., Liu, W., Liang, P., & Wang, T. (2017, October). Convolutional neural networks based transfer learning for diabetic retinopathy fundus image classification. In 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI) (pp. 1-11). IEEE. https://doi.org/10.1109/CISP-BMEI.2017.8301998

Li, X. K., Zhang, G. & Zheng, W. (2016). Deep learning and its parallelization. Big Data 12, 2016. https://doi.org/10.1016/B978-0-12-805394-2.00004-0

Pimm, S. L., & Joppa, L. N. (2015). How many plant species are there, where are they and at what rate are they going extinct?. Annals of the Missouri Botanical Garden, 100(3), 170-176. https://doi.org/10.3417/2012018

Reyes, A. K., Caicedo, J. C., & Camargo, J. E. (2015). Fine-tuning Deep Convolutional Networks for Plant Recognition. CLEF (Working Notes), 1391, 467-475. http://ceur-ws.org/Vol-1391/121-CR.pdf

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4510-4520). https://doi.org/10.1109/CVPR.2018.00474

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. https://arxiv.org/abs/1409.1556

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958. https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_campaign=buffer&utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com

Sun, Y., Liu, Y., Wang, G., & Zhang, H. (2017). Deep learning for plant identification in natural environment. Computational Intelligence and Neuroscience, 2017. https://doi.org/10.1155/2017/7361042

Sweetwilliams, F. O., Matthews, V. O., Adetiba, E., Babalola, D. T., & Akande, V. (2019, December). Detection of Sigatoka Disease in Plantain Using IoT and Machine Learning Techniques. In Journal of Physics: Conference Series (Vol. 1378, No. 2, p. 022004). IOP Publishing. https://doi.org/10.1088/1742-6596/1378/2/022004

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9). https://doi.org/10.1109/CVPR.2015.7298594

Uchida, U. (2020). Why MobileNet and Its Variants (e.g., ShuffleNet) Are Fast. https://medium.com/@yu4u/why-mobilenet-and-its-variants-e-g-shufflenet-are-fast-1c7048b9618d.

Wu, S. G., Bao, F. S., Xu, E. Y., Wang, Y. X., Chang, Y. F., & Xiang, Q. L. (2007, December). A leaf recognition algorithm for plant classification using probabilistic neural network. In 2007 IEEE international symposium on signal processing and information technology (pp. 11-16). IEEE. https://doi.org/10.1109/ISSPIT.2007.4458016