Plant Leaf Classification Using Fine-Tuned Transfer Learning

Ch. Lakshmi Narayana *[1], Kondapalli Venkata Ramana [2]

[1]Research Scholar, Department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam, AP, INDIA.

[2]Associate Professor, Department of Computer Science and Systems Engineering, Andhra University, Visakhapatnam, AP, INDIA.

Email id: [1]Lakshminarayana2796@gmail.com

Article Info
Page Number: 6051-6070
Publication Issue:
Vol. 71 No. 4 (2022)

Article History
Article Received: 25 March 2022
Revised: 30 April 2022
Accepted: 15 June 2022
Publication: 19 August 2022

Abstract

Enhancing our understanding and knowledge of the plants around us is highly essential and plays a crucial role in the field of medicine, economy, and organic agriculture. As a result, researchers have done lots of research to protect the biodiversity of plant life on the planet. In general, research in this discipline focuses on identifying plant species based on leaf images. Deep learning algorithms have shown highly promising results and they are now extensively used in research in this area. In this paper, we build CNN and transfer learning models for identifying plant species. This type of network is prone to overfitting, but regularization techniques like batch-normalization and dropout can solve these problems. In addition, transfer learning and parameter fine-tuning performed well on the same dataset. The proposed approaches were tested on the Leafsnap dataset with 184 classes and achieved a peak training accuracy of 98.64 and a peak testing accuracy of 95.78 percent.

Keywords: - Leaf classification, Convolution Neural Network (CNN), Transfer learning, Fine-Tuning, Leafsnap.

1. Introduction

The identification and categorization of plants (almost 50 thousand species) are one of the most critical challenges facing in the field of agricultural research and agricultural products [1, 2]. Strong knowledge of plants is necessary for recognizing new and unique species of plants, which is important for enhancing pharmaceuticals, environmental balance, agricultural usage, and robustness. Botanists employ changes in leaf features as a comparison tool in their plant research. Furthermore, because of the rapid growth of genetic and science areas, some of the complex plants cannot even be identified by professionals [3]. As a result, we attempted to overcome this problem using computer vision or machine learning approaches [2, 4].

In machine learning, despite the various contributions that have been performed, plant identification looks like a tricky and unresolved problem. This is because, in reality, all plants look similar in terms of color and shape. Furthermore, plants leaf morphology differences in size, thickness, shape, contour, and other factors have been incorporated into this complex task [5, 6].

The features such as shape, surface, and contour are used for recognizing leaves of distinct species of plants. The two main forms of feature extraction approaches to analyze leaf pictures are handcrafted and deep learning features. The creation of handcrafted features is extracted by computer vision specialists who encode morphological features set by botanists [7]. On the other hand, the deep learning features can be learned automatically using deep learning algorithms. Deep learning is a novel and advanced technology for data analysis and image processing that gives promising outcomes. Deep learning has been effective in a variety of disciplines and recently started working in the area of agronomy and food [5]. As a result, deep learning-based leaf identification has gained popularity.

To train a deep learning algorithm, one needs a large volume of data, greater computational power, cost, and time, in some instances need days of training. However, in practical applications, gathering and classifying a huge amount of data in a certain area is time consuming and expensive challenge, and might be not possible. As a result, in various situations, the thought of acquiring a huge amount of data in certain fields may not be acceptable. This issue is one that makes using deep learning models challenging. According to the experts, people were capable of learning millions of species in some instances because they were capable to develop this skill by obtaining information over time and applying it to learn new objects. Indeed, the researchers think that identifying objects from the past improves in identifying new objects due to their similarities and relevance to new ones. In deep learning, the models that have been trained for one type of classification problem can be applied to other classification problems. Hence, after fine-tuning pre-trained deep learning models on a dataset can be applied to new problems in different domains [2]. This is referred to as transfer learning. Transfer learning has been used successfully to modify existing neural networks for image classification, object detection, translation and voice synthesis, and several other domains [9].

In reality, this is a strategy for accelerating your learning process. This technique has shown to be effective for various identification and classification applications. Our proposed model also falls into this category for plant classification. In fact, the major goal is to examine most recent transfer leaning model scores such as VGG16 [9], ResNet50 [10], Densenet201 [11], Xception [12], InceptionResNetV2 [13], and EfficientNetB7 [14]. These transfer learning techniques and Custom CNN are used in this work to detect the leaves.

2. Related Work

Recently, various experiments using machine learning techniques have been done for the recognition of plant species. Mostly identification of plant species done based on leaf image datasets such as ICL[17], Swedish[16], Flavia[15], Foliage[18], and LeafSnap[1] real-time datasets were frequently utilized, as have Flower17[19] and Flower102[20]. The CNN model and transfer learning techniques are utilized in this work to identify the leaves.

Kumar et al. [1] present LeafSnap, a smartphone app that allows people to identify plant specie based on leaf images. For 184 tree species, LeafSnap obtains a top-5 accuracy of 96.8% and top-1 accuracy of around 73 %. PlantNet is a citizen science technique proposed by Joly et al. [21] to accelerate the collecting and integrating of observed botanical picture data. PlantNet's online interface allows you to identify a plant by using images of various plant parts (such as leaves, fruits, barks, and flowers) as well as the whole plant. PlantNet was examined on nearly half of France's plant species (2200 types), with a top-5 classification accuracy up to 69 percent for individual images.

Kaya et al. [22] examined four transfer learning models for plant species categorization using DNN. Their studies showed that the analyzed models recognized the species of plants with excellent accuracy on flavia, Swedish leaf UCI Leaf and Plantvillage datasets.

Ghazi et al. [23] identified many DNN performance-impacting parameters. For this objective, they had proposed an approach for identifying plant species taken in a picture using DCNN. The aspects discussed are implemented on three prominent deep learning networks, AlexNet, GoogLeNet, and VGGNet for performance measures with an accuracy of 80% on LifeCLEF2015 data.

Fine-tuning of CNN refers to the use of pre-trained CNN models and weights, by making changes in hyperparameters and the network's final layer. Sungbin et.al [24] fine-tuned the PlantCLEF dataset using GoogleNet. The validation split of 0.2 was determined by splitting the dataset into 5 folds. For integrating the results, the author tried the Bordafuse technique and the majority voting method. The best accuracy was obtained by training 5 CNNs and applying a majority voting technique for score ranking.

However, the deep learning approaches also have certain drawbacks. If the dataset has fewer images than number of images required then it would be difficult to train the model. As previously said, Transfer Learning is an effective strategy for this case. If one's job is related to a network that is previously trained, you may use transfer learning to change the network so that it can be used in your issue with a small number of labelled pictures [2]. In reality, the transfer learning approach may address the issue of data scarcity by using pre-trained models to train desired models on selected datasets [25].

Fortunately, the "ImageNet Large Scale Visual Recognition Challenge (ILSVRC)" winner models, such as VGG16 [9], ResNet50 [10], Densenet201 [11], Xception [12], InceptionResNetV2 [13], and EfficientNetB7 [14] are pre-trained models available publicly. These models may be utilized for transfer learning. In this research, we will provide a transfer learning technique for identifying species of plant.

3. Materials and Methods

In this paper, we introduce various methods that can recognize plant species using different types of Convolution neural networks. The proposed method will take leaf images as input. Leafsnap[1], a freely available dataset is used for experimental study. The process of identification of plant species is visualized in Figure 1.

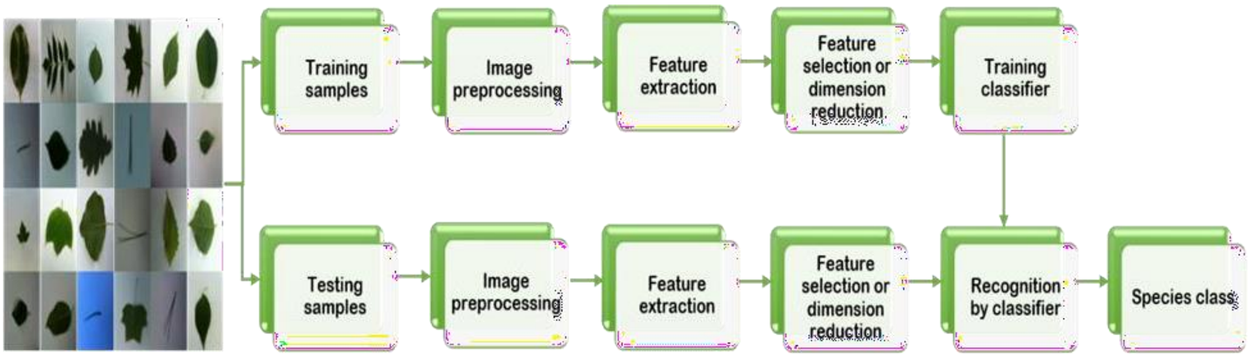


Figure 1. Plant Identification flow diagram

3.1. Leafsnap Dataset

The Leafsnap dataset is available for free on the internet at http://leafsnap.com/dataset/. It includes 185 plant species in the Northeastern-UnitedStates. The dataset comprises 30866 leaf images, with and 7719 field and 23147 lab leaf images respectively. The images in the laboratory are high quality, these images are taken in regulated back-light and front-light variants, with various samples per species. Since the images of the field are captured using a mobile phone, the quality is typically less than the quality of lab images.

We have built a histogram to analyze how often images are presented in Leafsnap dataset. you can observe that majority of leaf species contain 100 to 200 images. A species' image count ranges from 50 to 450 as shown in Figure 2.

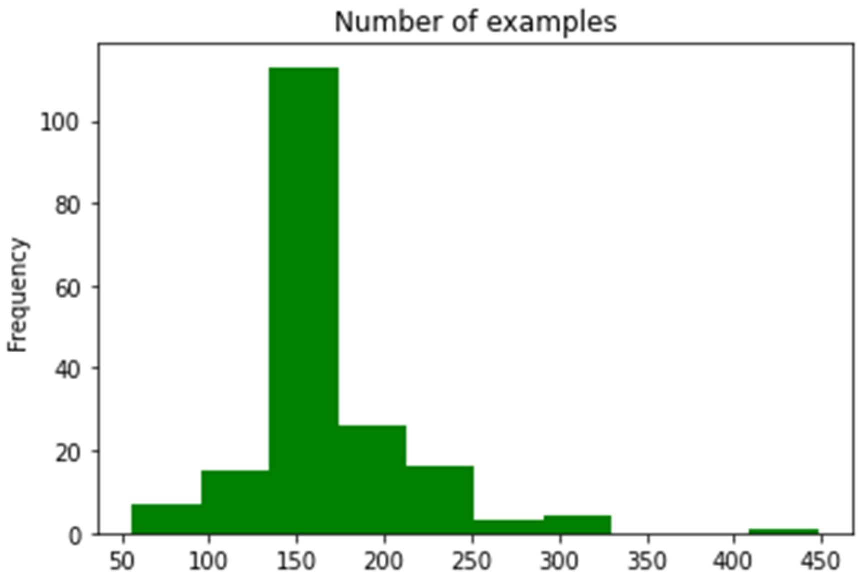


Figure 2. Distribution of Leaf images in Leafsnap dataset

3.2. Pre-processing

In order to train the data, we cannot send the leaf images directly as input into CNN, we must first pre-process the leaf images. The steps to pre-process the images are as follows:

- Read text file containing image information. Which includes information of 30866 images where each row contains image path, segmented image path, name of the species of image, and source from where the photograph was shot in field or lab.
- Instead of having a Latin name for each species, we need numerical labels for passing as input to CNN thus we add another column to the data frame containing them.
- We split our data frame into 3 parts as 80% for train, 10% for validation, 10% for the test by taking random samples from each species and saving these split data frames, which are useful for train, validate, test with more CNN models with different picture sizes. As we have 30866 images, we split them into 24694 train, 3090 validation, and 3082 for test images.
- Read the images of both lab and field and Resize the picture to desired pixels. Here we resized leaf images to 64X64 pixels for Custom CNN, VGG16, ResNet50, DenseNet201, EfficientNetB7, 71X71 for Xception, and 75X75 for InceptionResNetV2.
- Read the resized images as RGB arrays with the shape of [height, width, 3] where 3 represent Red, green and blue color channels.
- Now, the image vectors should be arranged in an array vertically such that each row signifies each image. the form of these arrays should be (24694, 64, 64, 3) for train, (3090, 64, 64, 3) for validation, and (3082, 64, 64, 3) for the test.
- Normalize images pixel values, in general, pixel values range from 0 to 255. Now we rescale the pixel values to range 0 to 1 by dividing with 255 which is preferred for CNN.
- one-hot encode the species numerical labels; we create a new category column for each categorical value and a binary value of 0 or 1 is assigned to it (for example if an image is 38th of 185 classes then we will mark the 38th column value of that image row as 1 and the remaining values as 0). Where each row in an array indicates an image and each column a class.
- Save the train, validation, test input vectors, and one-hot encoding vectors so that we can preserve the randomized vectors and we don't have to go through all of the aforementioned processes again in the future session.

3.3. Model Architectures

Computer vision is advancing at a rapid pace. One of the reasons for this is Deep learning. Various Deep learning architecture improvements have considerably benefitted a wide range of problems, such as image identification, image creation, and tagging. A convolutional neural network (CNN) is a type of Deep Learning model that is widely used to analyze images.

In this paper, we design a custom CNN model and used 6 different transfer learning models. CNN's are made up of layers that analyze visual data. A CNN receives an image as input and then processes it through various layers. There are several different sorts of layers, the most widely used layers are Convolutional, pooling, fully-connected layers as shown in Figure 3.

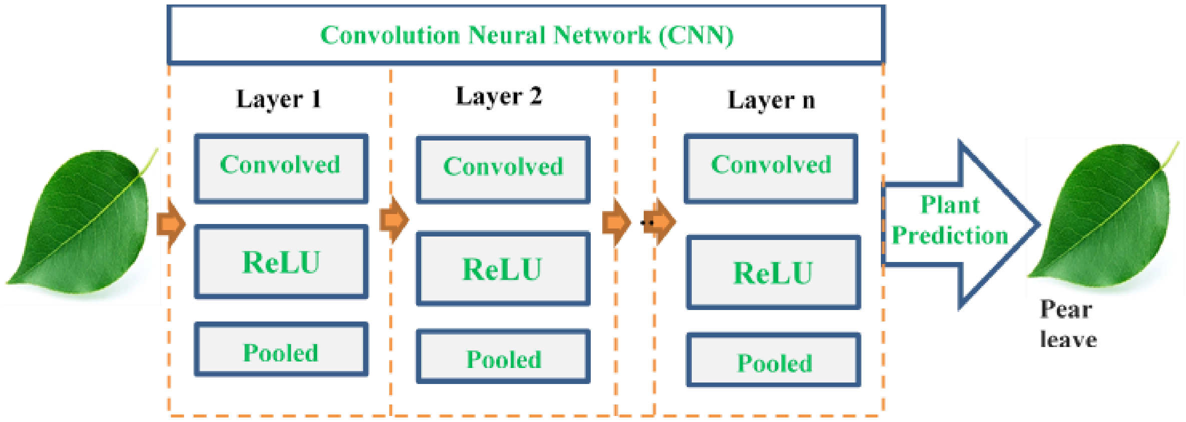


Figure 3. CNN Architecture for leaf classification

Convolutional layer:

The convolutional layer is the first layer in the network, it is also known as a feature extractor. convo layer takes a small local region of the image with the same dimensions as a filter and performs the convolution operation with filters. Then we slide the region over the same image and the process is repeated on the entire image. This Convolution layer output is passed into the next layer as input. Convo layer additionally has activation functions, that convert its inputs into outputs with a specific range. There are different types of activation functions, where ReLU is the most commonly used activation function. When it receives a negative input value, it returns zero, but when it receives a positive input value, it returns the same value. As a result, output ranges from 0 to infinity.

Pooling layer:

After Convolution, the pooling layer is applied to minimize the input dimensions. It is used as a transition layer in between two convolutional layers. If the output of the convo layer is directly passed to a fully-connected layer without using pooling then the computational power will become expensive, which is not preferred. So, the pooling is added to minimize the size of the input image.

Fully-connected layer:

In this layer, the inputs of each element in one layer are interconnected to each and every element of the subsequent layer. In most deep learning architectures, these layers are present at the end that combine the data acquired by the preceding layer to produce the final result.

3.3.1. Custom CNN

The model accepts 64x64x3 images as input, which are 64x64 colored images with RGB bands. In the first convolution layer, 32 (5x5) filters having stride 1 are used with no padding. As a result, the output of this convolution layer is 60x60x32. This is passed into a max - pooling, which slides a (2x2) window across the input with a stride of 2. As a result, the input was reduced to 30x30x32 output. The subsequent convolution layer is made up of 64 5x5 filters are used with a stride of 1. This yields a result of 26x26x64. This is passed through the max-pooling layer, which reduced the size to 13x13x64 and then the output is flattened and sent to a fully-connected layer with 1000 nodes, then forwards the output to the SoftMax layer with 185 nodes, which classifies the plant species among 185 classes. The architecture of Custom CNN is shown in Figure 4.

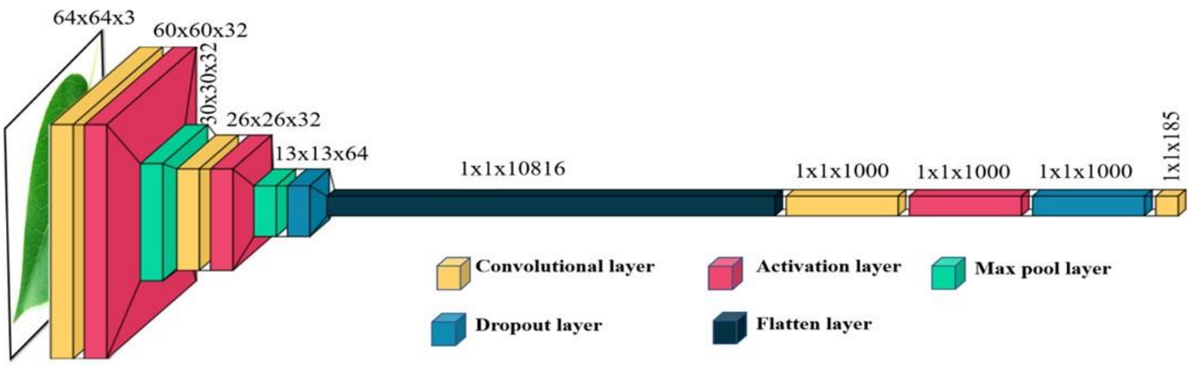


Figure 4. Architecture of Custom CNN

3.3.2. Transfer Learning models:

Transfer learning is the concept of breaking out from the isolated learning and applying the knowledge gained from one task is used to solve another. These transfer learning models are already trained on a huge dataset like ImageNet, which contains millions of images with different categories. The weights learned from this pre-training are recorded and utilized to initialize the model.

The pre-trained models used in this paper are as follows:

3.3.2.1. VGG16

The VGG16[9] CNN model topped the ILSVR contest in 2014. It is largely acknowledged as one of the best classification model designs currently available. The main aspect of VGG16 is that instead of having a lot of parameters, they focused on passing images through convolutional layer with the filter size of 3x3 and then the max-pool layer of 2x2 filter. The convolution kernels in the first four blocks are 64, 128, 256, and 512, respectively. This combination of convolutional and max-pooling layers remains consistent throughout the architecture. Finally, it has two fully connected layers and a SoftMax. The 16 in VGG16 refers to how many weighted layers it contains [9]. The architecture of VGG16 can be seen in Figure 5.

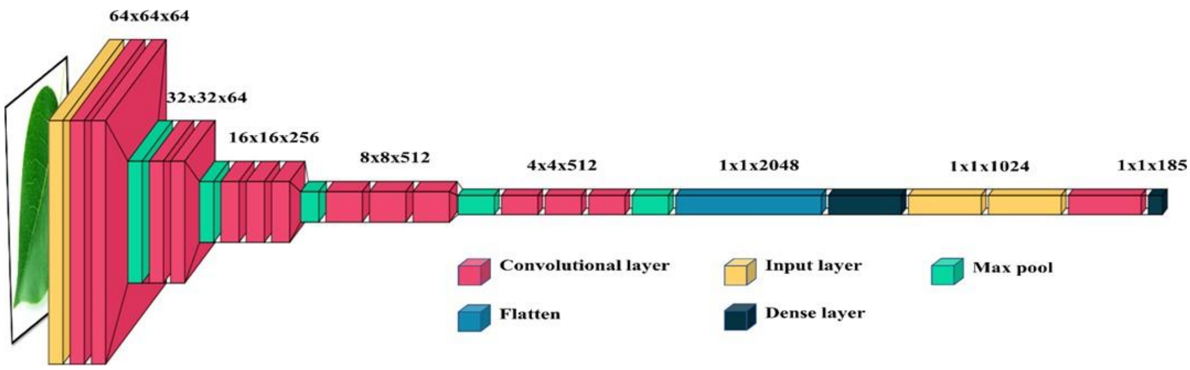


Figure 5. Architecture of VGG16

3.3.2.2. ResNet50

ResNet is a deep convolutional neural network that won the 2015 "ImageNet Large Scale Visual Recognition Challenge (ILSVRC)" by using transmission of features to avoid gradient vanishing, which allows a more deeper network than previous models and trained more effectively [10]. According to previous research that a deeper network is significantly more effective than a shorter network, our model was created from a 50-layer residual network model and has 177 layers in total. ResNet50 was trained on an ImageNet dataset that can classify photos of 1000 categories.

In the ResNet50 architecture, there are two types of modules. The first is an identity-block, in this block the size of input and output are the same. Another is a conv-block, which contains a convolution layer at the shortcut. In this block, the input size is much less than the output size. In both blocks, convolution layers are added to the network's starting and ending. This is a method known as bottleneck design, and it decreases the number of parameters that will not affect network performance much. The ResNet50 architecture can be seen in Figure 6.

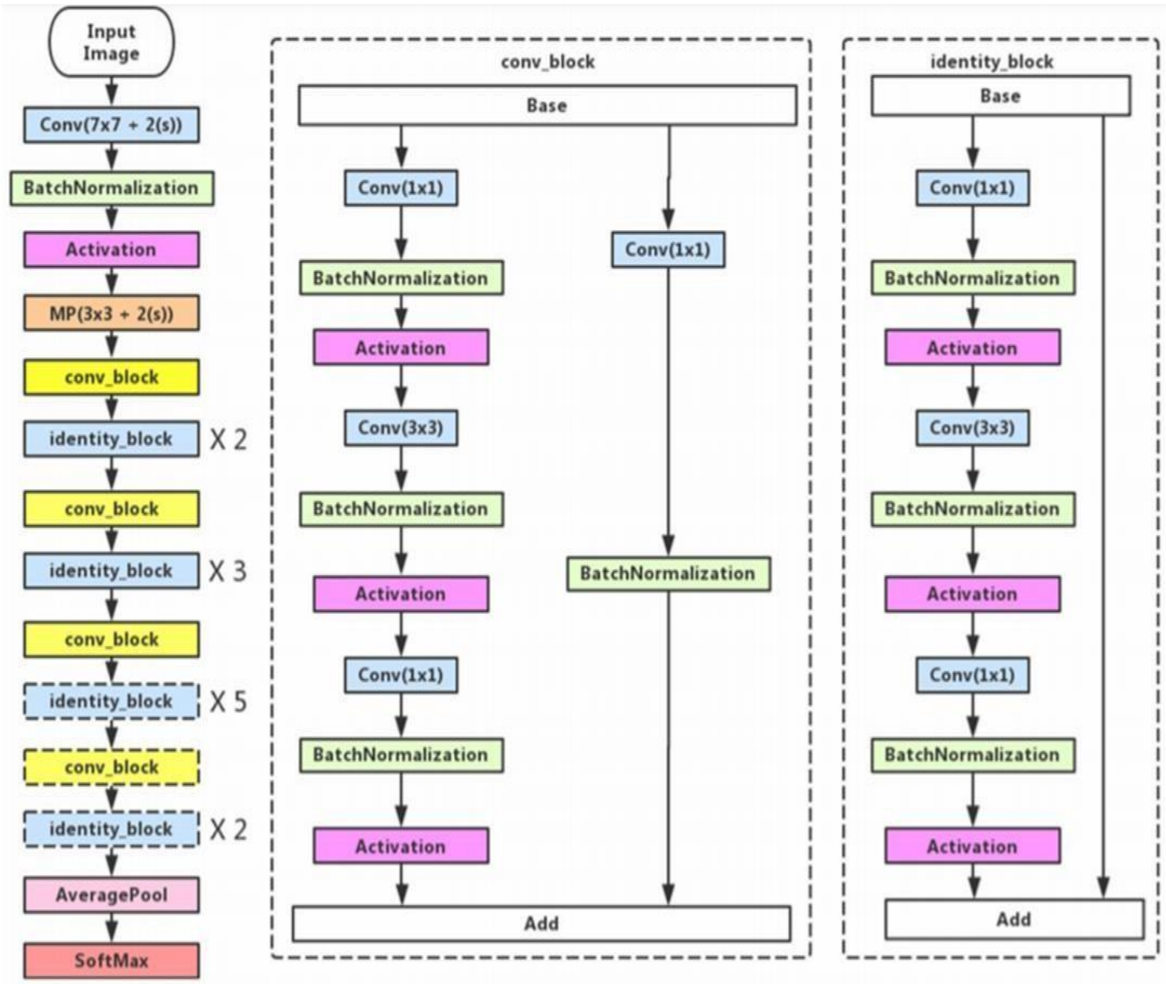


Figure 6. (Left) ResNet50 architecture. (Middle) Convolution block. (Right) Identity block.

3.3.2.3. Densenet201

Huang et al. [11] proposed the DenseNet classification model. This network model allows reusing network features for any two layers that have the same feature graph size. In order to learn a more efficient and highly accurate model Simultaneously, the neural network's feature propagation of the input picture is reinforced, while gradient diffusion is minimized. Finally, the classification results' accuracy has dramatically increased. Model overfitting is decreased to some degree in DenseNet because of the regularization impact of dense connections, and the gradient flow and information in a network is enhanced. The images are sequentially processed through the dense block and transition layer of various layers after entering the classification model. Finally, the output is passed to a fully connected (FC) layer. Figure 7 depicts the dense block and transition layer structures, as well as the dense block's nonlinear transformation. The transition layer structure is shown in Figure 8, it connects two adjacent dense blocks and minimizes the feature map size. The transition layer comprises a 1 × 1 convolution and 2 × 2 avg pooling, which are applied to reduce the model size.

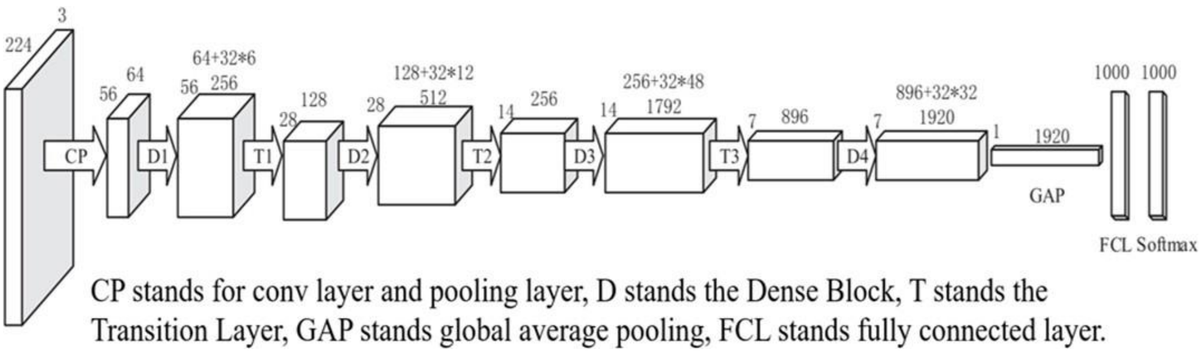


Figure 7. Densenet Architecture

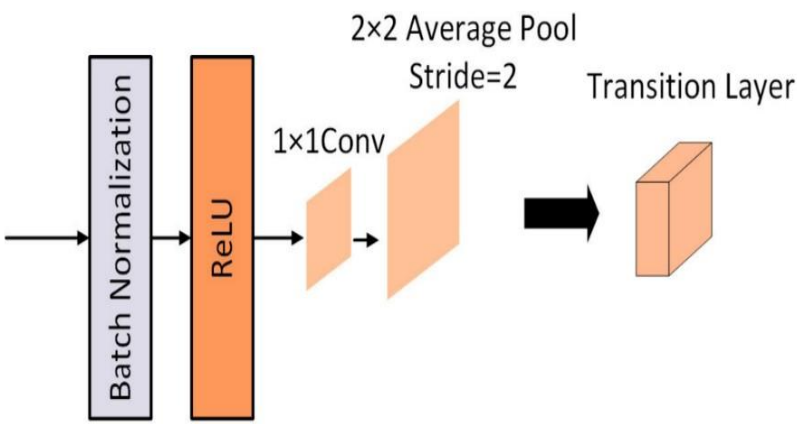


Figure 8. DenseNet’s Transition layer.

In Figure 9, H() denotes the process of a nonlinear transformation of a batch normalization, activation function (ReLU), and a 3 x 3 convolution layer. In the dense block 1x1 convolutional layer is inserted before the 3x3 convolutional layers to reduce the number of features and enhance the performance of model.

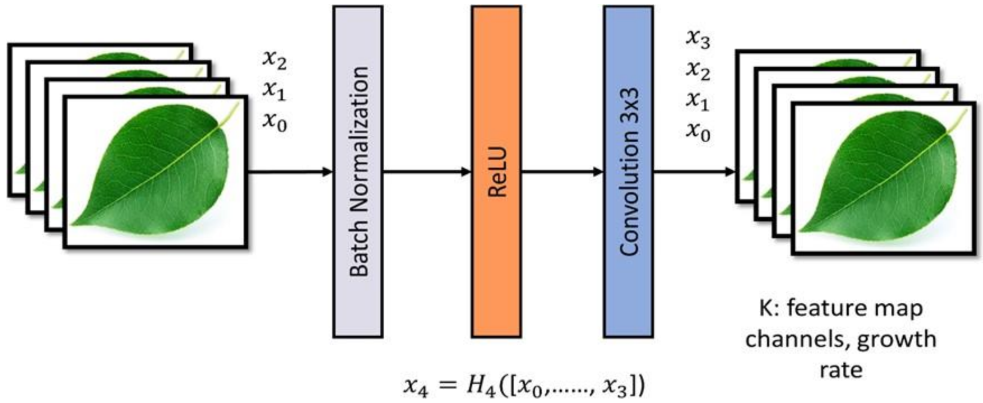


Figure 9. Process of Nonlinear transformation.

3.3.2.4. Xception

The Xception model [12] is an enhancement of the Inception model in which depthwise separable convolutions replace the normal Inception modules. Rather than dividing the input data into many compressed pieces, it maps the spatial relationship for each output channel independently, followed by a 1x1 depthwise convolution to capture cross-channel correlation.

This is substantially the same as the current procedure known as a "depthwise separable convolution," This consists of a depthwise convolution (a spatial convolution performed to each channel independently) followed by a pointwise convolution (a 1x1 convolution across channels). This may be thought of as looking for correlation in a 2D space first, then in a 1D space. This 2D+ 1D mapping is relatively simpler than 3D mapping. The architecture of Xception is shown in Figure 10.

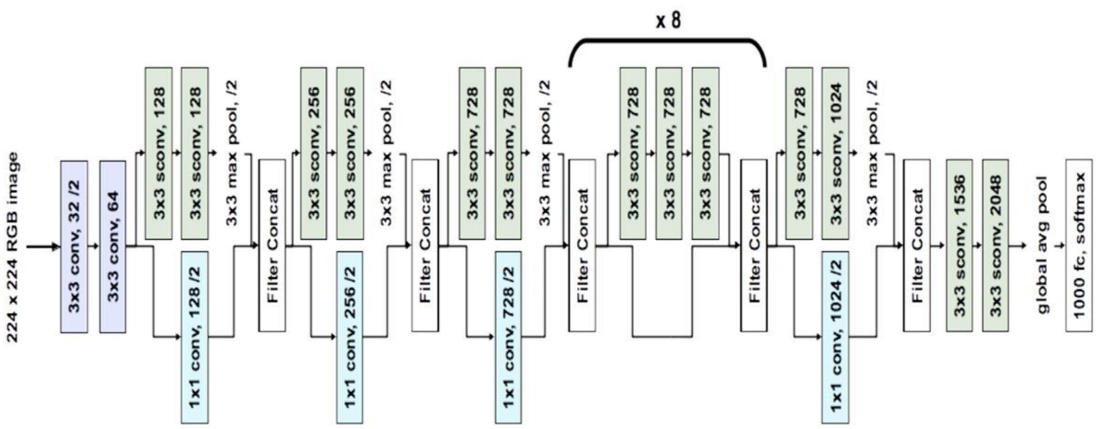


Figure 10. Xception Architecture.

3.3.2.5. InceptionResnetv2

The inceptionResNetv2 [13] model developed by GoogleAI is one of the key elements of this research. The network is made up of several different creative ways that allow it to perform so effectively and allow us to create captions. This network has 164 layers; it is commonly known that deeper networks result in a better image analysis.

The term "Inception-Resnet" refers to the use of inception convolutions in the model. Figure 11 shows the network's first layers, which include three conventional convolutional layers, a max-pooling layer, two convolutional layers, and another max-pooling layer. The network's next step is inception convolution, that involves performing convolution operation on input several times with various filter sizes for each convolution, then combining (stacking) the results and sending them on to the remaining part of the network. Figure 12 shows inception module and Figure 13 shows the Reduction.

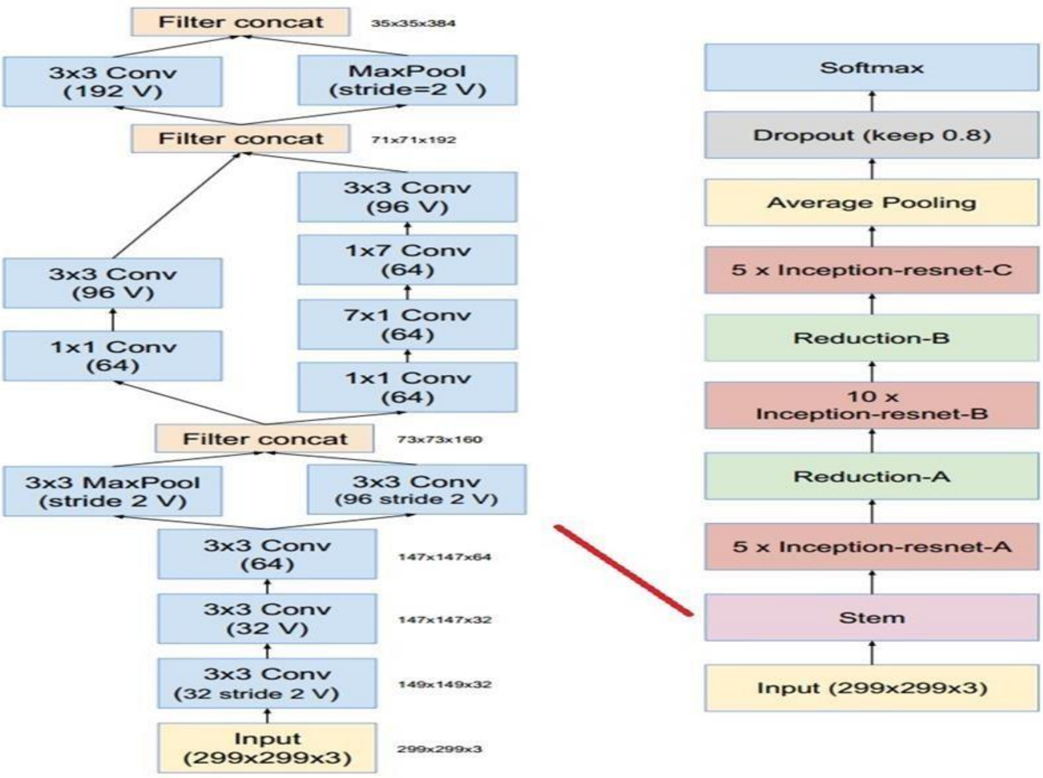


Figure 11. InceptionResnetv2 Architecture.

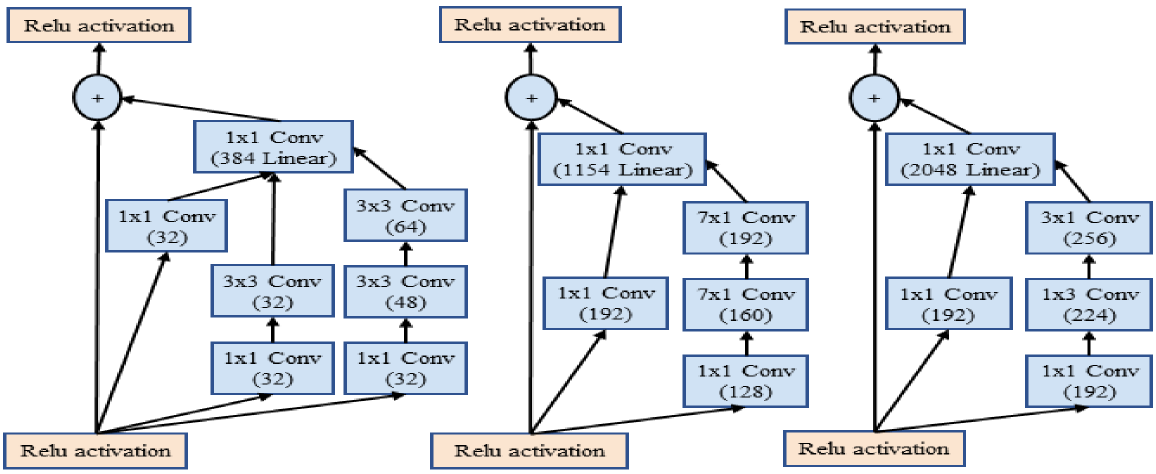


Figure 12. Inception-ResNet-v2: Inception-A (Left), Inception-B (Middle), Inception-C (Right).

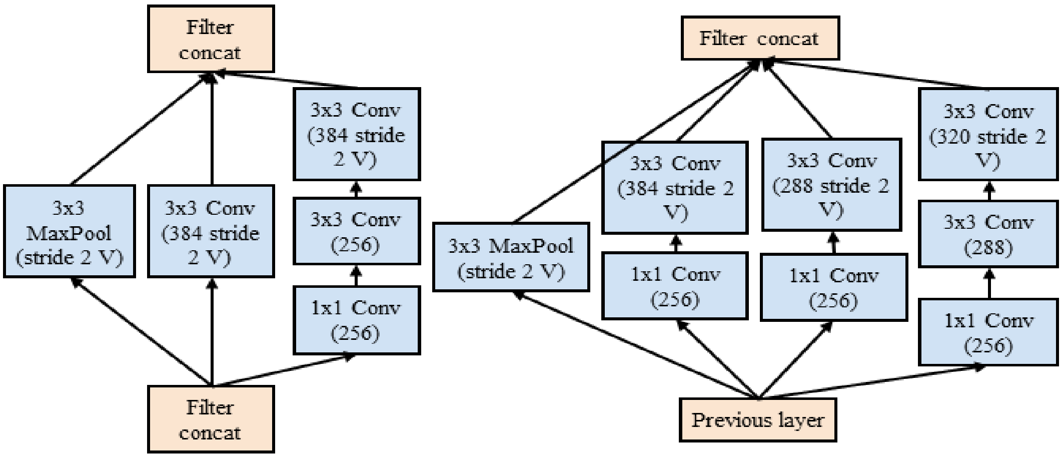


Figure 13. InceptionResnetV2: Reduction-A(Leftside), Reduction-B(Rightside).

The network then repeats inceptions and residuals several times, this repetition of certain portions may lead to overfitting to avoid this dropout layers are used to drop weights randomly. In addition, the last two layers are fully connected layer and SoftMax layer which used to distribute probability scores to 185 neurons.

Because the InceptionResnetv2 employs a complex network to extract required features from pictures, we chose CNN as the Image encoder. Szegedy et al. [13] discuss a more detailed explanation of the network.

3.3.2.6 EfficientNetB7

EfficientNet CNN architectures are developed according to the resources available, and scaling happens to obtain better performance as resources grow. ResNet18, for example, maybe scaled to ResNet101 by increasing the number of layers. The conventional method for scaling the model has been to randomly increase the CNN width or depth or the input picture quality. This technique requires time- consuming manual tweaking and sometimes yields sub-optimal results. Tan et al. [14] presented a new compound scaling strategy that uses a fixed set of scaling parameters to uniformly scale network depth, breadth, and resolution for improved performance. Initially, a new baseline network known as EfficientNetB0 was developed, which was then built up to create a family of EffcientNets using the compound scaling approach. This method has resulted in eight EfficientNets variations, notably EfficientNetB0 through EfficientNetB7. Scaling the network incrementally increases model performance by balancing the architecture's breadth, depth, and image resolution compound coefficients, While the accuracy improves. Mobile inverted bottleneck convolution (MBConv) [14] with a squeeze and excitation optimization is the foundation of the EfficientNet architecture. The MBConv concept is seen in Figure 14. The EfficientNet network family has a variable number of MBConv blocks. The top-performing model, EfficientNetB7, surpasses previous state-of-the-art CNNs on ImageNet in terms of accuracy [14]. Figure 14 depicts the EfficientNetB7 network design. It is split into seven sections depending on filter size, striding, and channel count.

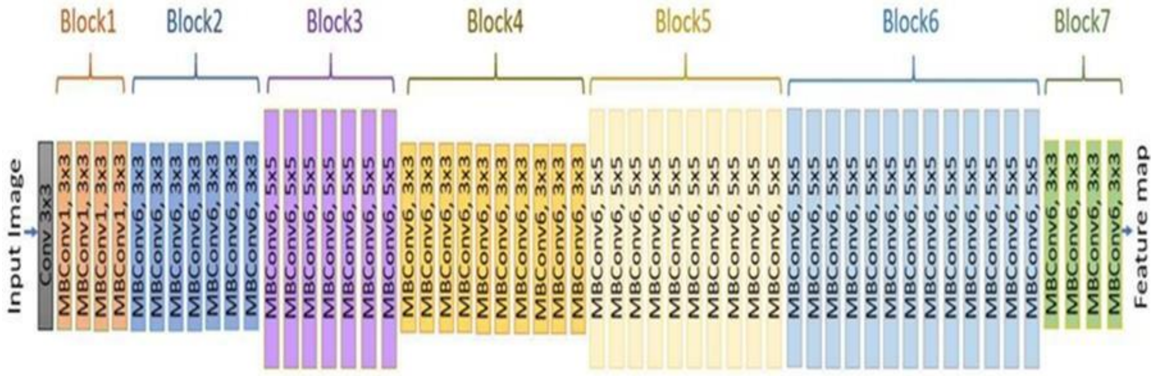


Figure 14. EfficientNetB7 architecture using MBConv as fundamental building pieces.

The overall architecture may be split into seven blocks, depending on filter size, striding, and channel count. MBConv is the network's fundamental building component. Each MBConvR block is labeled with the filter size it corresponds to, and the R=1 and R=6 indicate the conventional ReLU and ReLU6 activation functions, respectively.

3.4 Finetuning Transfer learning models:

In Finetuning models, we freeze all layers in the Pre-trained model to avoid changing of layer weights. Then we unfreeze batch normalization since it contains non-trainable weights that get updated during training and these are connected to a Dense layer with Relu activation function as shown in Figure 15. we applied batch normalization, which has the effect of stabilizing the learning process and significantly minimizing the number of training epochs necessary to train deep networks. A dropout layer is added, which is a technique for preventing overfitting in a model. Because there are 185 categories to classify, a 185-unit dense layer with SoftMax activation is used at the end. The SoftMax layer will give an output value that the images belong to a specific class.

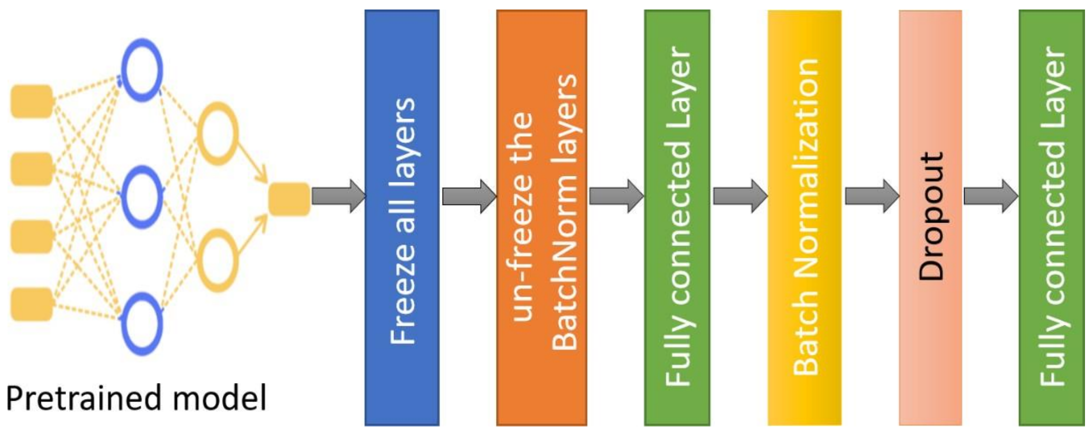


Figure 15. Finetuned Transfer Learning model

4. Experimentation

The multi-class classification was utilized since the Leafsnap dataset comprises 185 categories. The metrics are computed utilizing indices like "True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN), which are based on the values in the confusion matrix obtained in such classifications. TP is the number of properly classified leaf images in the respective class, while TN denotes the total number of other categories of images classified correctly. FN denotes the number of predictions in which the classifier incorrectly classifies images of positive class as negative. FP shows the number of predictions in which the classifier incorrectly classifies images of negative class as positive" [26][27].

The accuracy of cutting-edge CNN models described in this research are evaluated using Accuracy. Accuracy is the proportion of properly categorized samples among all samples.

$$\text{Acc} = \frac{TP+TN}{TP+FN+TN+FP} \tag{1}$$

The categorical cross-entropy loss function computes an example's loss by adding the following sums:

$$\text{Loss} = -\sum_{i=1}^{\text{output}} \text{size}(x_i) \cdot \log(\hat{x}_i) \tag{2}$$

where $\hat{x}$ represents softmax probability in the i^{th} class, x_i represents the matching target value. This loss is a great measure of how far apart two different probabilities are. The negative sign ensures that the loss reduces as the distributions get closer to one other.

5. Results

In this section, we analyze our CNN model's performance. After several experiments, the parameters used for these models gives good results for predicting multiple classes. Table 1 depicts the parameters used for our models.

Table 1. Parameter configuration for training.

Parameter	**Value**
Image size	64x64 (CNN, VGG16, ResNet50, Densenet201, EfficientNetB7), 71x71(Xception), 75x75(InceptionResNetV2)
Optimizer	Adam
Learning rate	0.0003
Batch Size	64
Epochs	100

On the leafsnap dataset, we analyzed and compared the performance of CNN and transfer learning-based methods. We applied a fine-tuned (FT) approach to six architectures of pre-trained CNN and custom CNN. The comparison of accuracies of all proposed models and previous studies as shown in Table 2.

Table 2. Comparision of models.

Year	Model	Train Accuracy	Test Accuracy
2017	LeafNet CNN[30]	-	86.3
2018	MSF-CNN [29]	-	85.28
2019	MobileNet[2]	-	90.54
-	FT_Custom CNN	95.66	84.43
-	FT_VGG16	91.6	89.29
-	FT_ResNet50	**98.64**	93.35
-	FT_DenseNet201	98.57	**95.78**
-	FT_Xception	98.59	92.05
-	FT_InceptionResNetV2	95.62	90.3
-	FT_EfficientNetB7	97.69	93.33

The Figures (17-30) below show the accuracy and loss curves produced by the Fine-tuned custom CNN, VGG16, ResNet50, DenseNet201, Xception, InceptionResNet v2, and EfficientNetB7 models on training and test sets over 100 epochs. The Accuracies of all these Fine-tuned models can be seen in Figure 16.

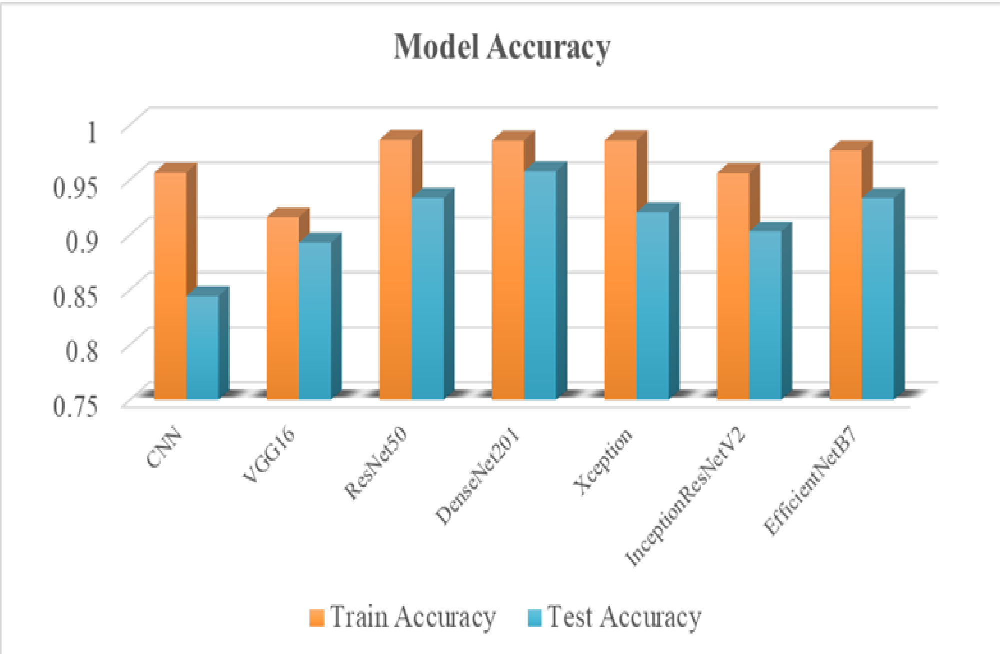
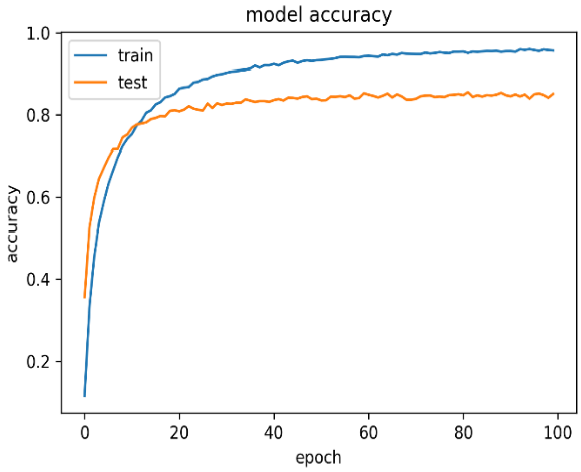


Figure 16. Models Accuracy.



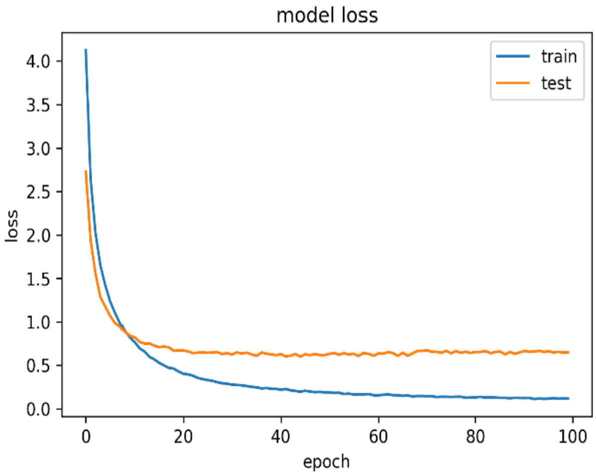
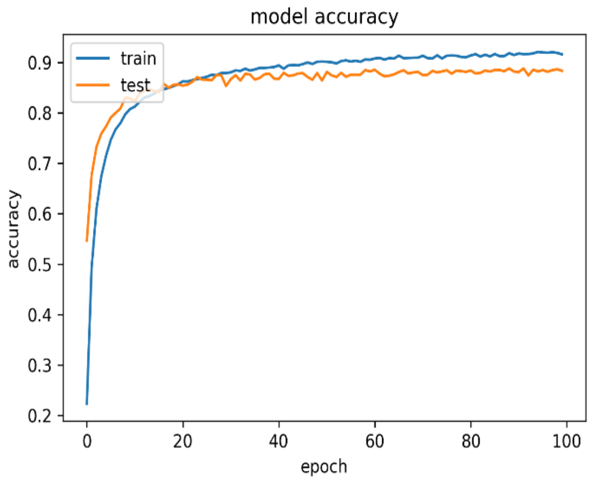


Figure 17. Custom CNN accuracy

Figure 18. Custom CNN loss



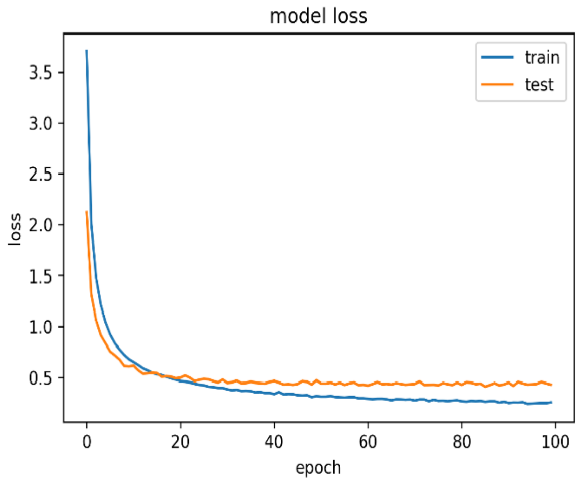
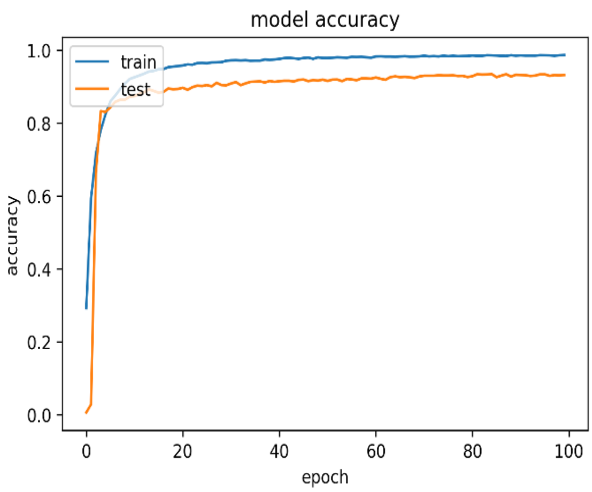


Figure 19. VGG16 accuracy

Figure 20. VGG16 CNN loss



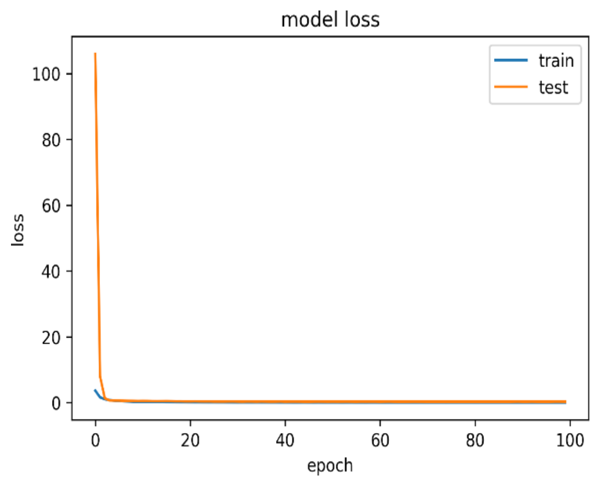


Figure 21. ResNet50 accuracy

Figure 22. ResNet50 loss

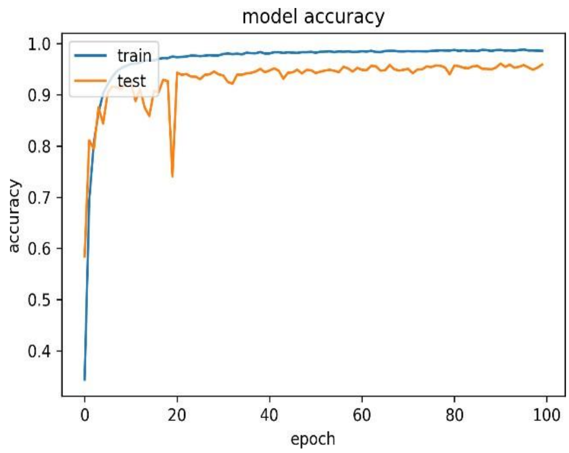


Figure 23. DenseNet201 accuracy

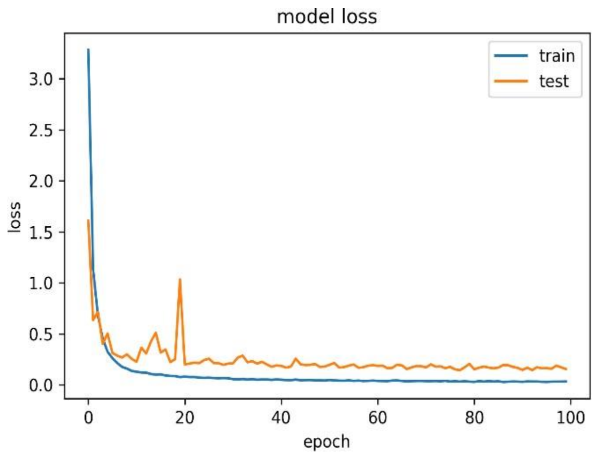


Figure 24. DenseNet201loss

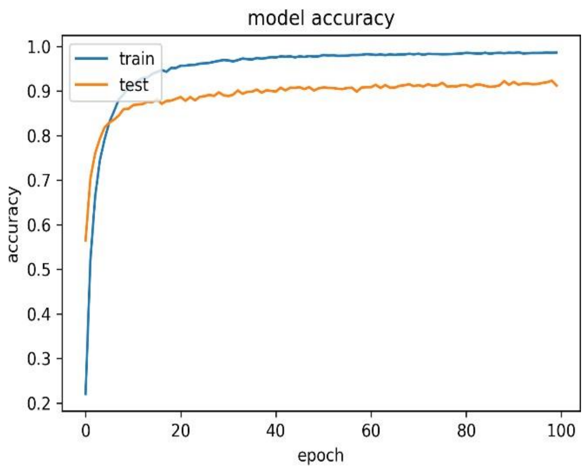


Figure 25. Xception accuracy

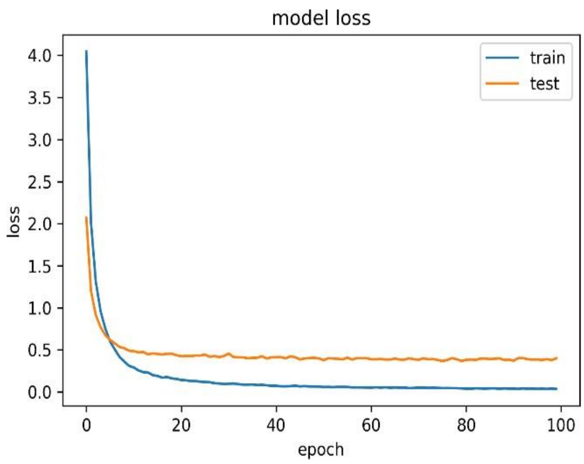


Figure 26. Xception loss

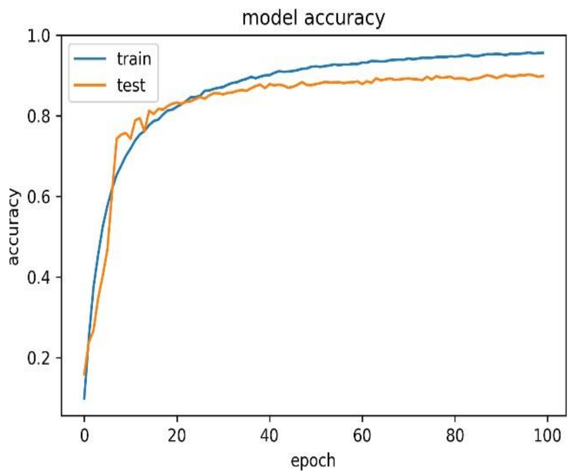


Figure 27. InceptionResNetV2 accuracy

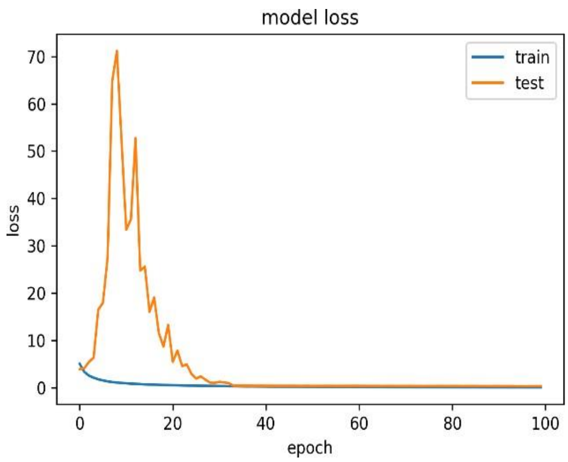


Figure 28. InceptionResNetV2 loss

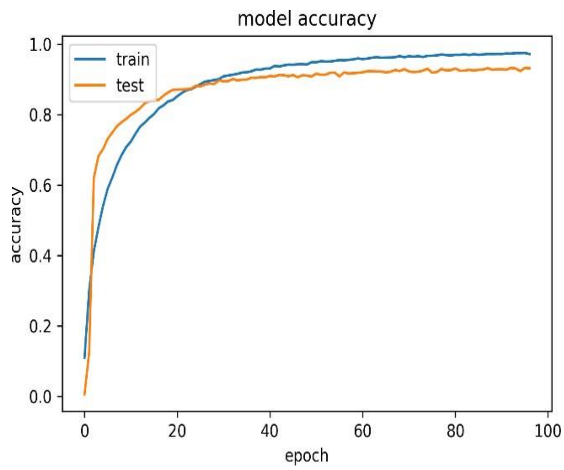


Figure 29. EfficientNetB7 accuracy

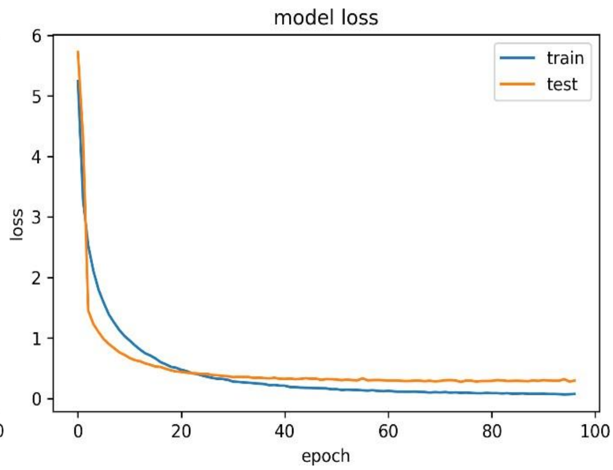


Figure 30. EfficientNetB7 loss

6. Conclusion and Future Scope

In this research, we proposed a fine-tuned deep convolutional network for plant species identification that feeds Leafsnap dataset, resulting in a model with more efficient image classification than classic CNN. When compared to custom CNN and fine-tuned transfer learning models of VGG16, ResNet50, DenseNet201, Xception, InceptionResNetV2, and EfficientNetB7, the Densenet201 network performed much better. A thorough experimental assessment on a benchmark dataset revealed that the suggested network outperforms state-of-the-artwork, with an overall train accuracy of 98.57% and test accuracy of 95.78% for the Leafsnap dataset. We observe that accuracy reduces as a result of the regular occurrence of identical leaf contours, particularly in closely related species, and that this may be addressed by integrating more leaf features.

References

[1] Kumar N, Belhumeur PN, Biswas A, Jacobs DW, Kress WJ, Lopez IC, Soares JV. Leafsnap: A computer vision system for automatic plant species identification. InEuropean conference on computer vision 2012 Oct 7 (pp. 502-516). Springer, Berlin, Heidelberg.

[2] Beikmohammadi A, Faez K. Leaf classification for plant recognition with deep transfer learning. In2018 4th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS) 2018 Dec 25 (pp. 21-26). IEEE.

[3] Ferentinos KP. Deep learning models for plant disease detection and diagnosis. Computers and electronics in agriculture. 2018 Feb 1;145:311-8.

[4] Lee SH, Chan CS, Wilkin P, Remagnino P. Deep-plant: Plant identification with convolutional neural networks. In2015 IEEE international conference on image processing (ICIP) 2015 Sep 27 (pp. 452-456). IEEE.

[5] Kamilaris A, Prenafeta-Boldú FX. Deep learning in agriculture: A survey. Computers andelectronics in agriculture. 2018 Apr 1;147:70-90.

[6] Kadir A, Nugroho LE, Susanto A, Santosa PI. Leaf classification using shape, color, and texture features. arXiv preprint arXiv:1401.4447. 2013 Nov 20.
[7] Lee SH, Chan CS, Mayo SJ, Remagnino P. How deep learning extracts and learns leaf features for plant classification. Pattern Recognition. 2017 Nov 1;71:1-3.
[8] Nam H, Han B. Learning multi-domain convolutional neural networks for visual tracking. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 4293-4302).
[9] Nam H, Han B. Learning multi-domain convolutional neural networks for visual tracking. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 4293-4302).
[10] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. InProceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 770-778).
[11] Huang G, Liu Z, Pleiss G, Van Der Maaten L, Weinberger K. Convolutional networks with dense connectivity. IEEE transactions on pattern analysis and machine intelligence. 2019 May 23.
[12] Chollet F. Xception: Deep learning with depthwise separable convolutions. InProceedings of the IEEE conference on computer vision and pattern recognition 2017 (pp. 1251-1258).
[13] Szegedy C, Ioffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning. InThirty-first AAAI conference on artificial intelligence 2017 Feb 12.
[14] Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks. InInternational conference on machine learning 2019 May 24 (pp. 6105-6114). PMLR.
[15] Wu SG, Bao FS, Xu EY, Wang YX, Chang YF, Xiang QL. A leaf recognition algorithm for plant classification using probabilistic neural network. In2007 IEEE international symposium on signal processing and information technology 2007 Dec 15 (pp. 11-16). IEEE.
[16] Söderkvist O. Computer vision classification of leaves from swedish trees. Master’s Thesis, Linköping University,2001.
[17] Silva PF, Marcal AR, Silva RM. Evaluation of features for leaf discrimination. InInternational Conference Image Analysis and Recognition 2013 Jun 26 (pp. 197-204). Springer, Berlin, Heidelberg.
[18] Kadir A, Nugroho LE, Susanto A, Santosa PI. Neural network application on foliage plantidentification. arXiv preprint arXiv:1311.5829. 2013 Nov 20.
[19] Nilsback ME, Zisserman A. Delving into the Whorl of Flower Segmentation. InBMVC 2007 Sep (Vol. 2007, pp. 1-10).
[20] Nilsback ME, Zisserman A. Automated flower classification over a large number of classes. In2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing 2008 Dec 16 (pp. 722-729). IEEE.
[21] Joly A, Goëau H, Bonnet P, Bakić V, Barbe J, Selmi S, Yahiaoui I, Carré J, Mouysset E, Molino JF, Boujemaa N. Interactive plant identification based on social image data. Ecological Informatics. 2014 Sep 1;23:22-34.

[22] Kaya A, Keceli AS, Catal C, Yalic HY, Temucin H, Tekinerdogan B. Analysis of transfer learning for deep neural network based plant classification models. Computers and electronics in agriculture. 2019 Mar 1;158:20-9.
[23] Ghazi MM, Yanikoglu B, Aptoula E. Plant identification using deep neural networks via optimization of transfer learning parameters. Neurocomputing. 2017 Apr 26;235:228-35.
[24] Choi S. Plant Identification with Deep Convolutional Neural Network: SNUMedinfo at LifeCLEF Plant Identification Task 2015. CLEF (Working Notes). 2015 Sep;2015.
[25] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. InEuropeanconference on computer vision 2014 Sep 6 (pp. 818-833). Springer, Cham.
[26] Atila Ü, Uçar M, Akyol K, Uçar E. Plant leaf disease classification using EfficientNet deep learning model. Ecological Informatics. 2021 Mar 1;61:101182.
[27] Hridoy RH, Rakshit A. BGCNN: A Computer Vision Approach to Recognize of Yellow Mosaic Disease for Black Gram. InComputer Networks and Inventive Communication Technologies 2022 (pp. 189-202). Springer, Singapore.
[28] Turkoglu M, Aslan M, Arı A, Alçin ZM, Hanbay D. A multi-division convolutional neural network-based plant identification system. PeerJ Computer Science. 2021 May 28;7:e572.
[29] Hu J, Chen Z, Yang M, Zhang R, Cui Y. A multiscale fusion convolutional neural network for plant leaf recognition. IEEE Signal Processing Letters. 2018 Feb 26;25(6):853-7.
[30] Barré P, Stöver BC, Müller KF, Steinhage V. LeafNet: A computer vision system for automatic plant species identification. Ecological Informatics. 2017 Jul 1;40:50-6.