Jurnal Teknologi

Full Paper

TOWARDS IMPROVED DISEASE IDENTIFICATION WITH PRETRAINED CONVOLUTIONAL NEURAL NETWORKS AS FEATURE EXTRACTORS FOR CHILI LEAF IMAGES

Nuramin Fitri Aminuddin, Herdawatie Abdul Kadir, Mohd Razali Md Tomari, Ariffuddin Joret, Zarina Tukiran*

Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, Batu Pahat, 86400, Johor, Malaysia

Article history
Received
16 February 2023
Received in revised form
4 October 2023
Accepted
19 October 2023
Published Online
18 February 2024

*Corresponding author
zarin@uthm.edu.my

Graphical abstract

Abstract

Chili is a popular crop that is widely grown due to its flavorful and spicy fruit that is nutritionally beneficial. For the benefit of economic growth, it is important to precisely assess the chili health. With the advancement of computer vision-based applications, methods such as feature descriptors have been utilized to assist farm owners in identifying chili diseases via chili leaf images. However, these feature descriptors still require the manual extraction of disease features in order to accurately identify chili diseases. In this research, pretrained Convolutional Neural Networks (CNNs) are proposed as feature extractors to identify healthy and diseased chili leaf images. Three pretrained CNN models, DenseNet-201, EfficientNet-b0, and NasNet-Mobile, are utilized for their ability to identify healthy and diseased chili leaf using five indexes: accuracy, recall, specificity, precision, and F1-score. These indexes are validated through a five-fold cross-validation method during the experiments. The experimental results show that the EfficientNet-b0 model achieved the highest identification performance, with indexes of accuracy, recall, specificity, precision, and F1-score of 97.05%, 0.97, 0.92, 0.92, and 0.94, respectively. Therefore, the use of pretrained CNNs as feature extractors has the capability to enhance the efficiency and accuracy of chili disease identification in agricultural settings.

Keywords: Diseased, chili leaf, pretrained, convolutional neural networks, cross-validation

Abstrak

Cili merupakan tanaman popular yang ditanam secara meluas kerana buahnya yang berperisa pedas dan bermanfaat dari segi pemakanan. Untuk pertumbuhan ekonomi, adalah penting untuk menilai kesihatan cili. Dengan kemajuan aplikasi berasaskan penglihatan komputer, kaedah deskriptor ciri telah digunakan untuk membantu pemilik ladang bagi mengenal pasti penyakit cili melalui imej daun cili. Namun begitu, kaedah ini masih memerlukan pengekstrakan ciri-ciri penyakit secara manual bagi mengenal pasti penyakit cili dengan tepat. Dalam penyelidikan ini, Rangkaian Neural Konvolusi (CNNs) yang terlatih dicadangkan untuk mengestrak ciri bagi mengenal pasti imej daun cili yang sihat dan berpenyakit. Tiga model CNN yang terlatih, DenseNet-201, EfficientNet-b0, dan NasNet-Mobile, digunakan bagi mengenal pasti daun cili yang sihat dan berpenyakit dengan menggunakan lima indeks: ketepatan, ingatan semula, kekhususan, keperincian dan skor F1. Indeks ini disahkan melalui kaedah pengesahan silang lima kali semasa eksperimen. Keputusan eksperimen menunjukkan bahawa model EfficientNet-b0 mencapai prestasi pengenalpastian tertinggi, dengan ketepatan, ingatan semula, kekhususan, keperincian dan skor F1 masing-masing sebanyak 97.05%, 0.97, 0.92, 0.92, dan 0.94. Oleh itu, penggunaan CNN yang terlatih sebagai pengekstrak ciri berpotensi untuk meningkatkan kecekapan dan ketepatan pengenalpastian penyakit cili dalam persekitaran pertanian.

Kata kunci: Berpenyakit, daun cili, terlatih, rangkaian neural konvolusi, pengesahan silang

1.0 INTRODUCTION

Chili, a type of spice crop from the Solanaceae family, is indigenous to South and Central America [1]. The crop is widely grown due to its flavorful and spicy fruit that is rich in nutrients [2]. These nutrients include vitamin C, fiber, phosphorus, potassium, and antioxidants, as well as flavonoids including cryptoxanthin, lutein, zeaxanthin, α-carotene, and β-carotene which have been shown to have cancer-fighting properties [3]. However, the chili is prone to various diseases caused by factors such as viruses, bacteria, fungus, pests, and the environment, which can negatively impact the chili cultivation process and reduce the fruit quality and yield. Approximately 70% of diseases as well as early disease symptoms can be identified by examining crop leaves, according to a report [4]. Therefore, it is important to accurately identify these diseases early on in the chili leaves in order to address these diseases and maintain the health as well as the productivity of the chili.

With the advancement of technology, several recent documented research studies have demonstrated the effectiveness of computer vision-based applications in the identification of diseases using crop leaf images. The computer vision-based applications can be divided into two approaches [5]: the use of conventional feature descriptors and the use of Convolutional Neural Networks (CNNs). In the feature descriptor approach, image features are manually extracted from the pixels of a leaf image and then fed into a machine learning-based classifier, which is trained to classify the features based on the input feature provided. The research in [6] entails the usage of a feature descriptor approach, utilizing the Scale-Invariant Feature Transform (SIFT) descriptor to identify unique points on the diseased chili leaf images, which are self-captured, and extract features based on those points. These extracted features are then supplied to several machine learning-based classifiers, including Decision Tree, K-Nearest Neighbors (KNN), Naive Bayes, as well as Support Vector Machine (SVM) for the purpose of disease identification. The findings of the research show that Naive Bayes classifier produces the highest accuracy, yielding an accuracy of 84.5%.

In contrast, the research in [7] entails the usage of a Gray Level Co-occurrence Matrix (GLCM) descriptor to extract six features, which are contrast, energy, entropy, dissimilarity, Inverse Different Moment (IDM), and correlation from self-captured chili leaf images with diseases. These extracted features are input into a SVM classifier that is subsequently trained to classify and identify the diseases. The authors report an overall accuracy of 88% for this method. Conversely, the research in [8] engages in a fused feature extraction technique utilizing Histograms Oriented Gradient (HOG) and Local Binary Patterns (LBP) as feature descriptors on diseased chili leaf images acquired from the PlantVillage website. Once the features have been extracted, the authors apply Principal Component Analysis (PCA), a dimensionality reduction method, in order to reduce the amount of features, which can assist to enhance the effectiveness and accuracy of the method. The extracted features are subsequently passed to several machine learning-based classifiers, including HistGradientBoosting, SVM, Naive Bayes, Logistic Regression and Decision Tree classifiers, for the purpose of disease identification. The highest accuracy is achieved through the use of the extracted features with the HistGradientBoosting classifier, resulting in an accuracy of 89.11%.

Under the CNN approach, the first few layers of a CNN typically consist of convolutional layers, which are designed to automatically extract features originating in an input leaf image. These extracted features are prepared to be fed into the classification layer, which will be trained to classify the input leaf image based on the extracted features. The research in [9] encompasses the usage of a CNN approach, where a basic CNN model is built from scratch to identify diseased chili leaf images. The images used in the research are self-captured, rather than obtained from a website source. The CNN model includes of two convolutional layers, and a softmax-based layer for classification tasks. The authors report an overall accuracy of 81.5% for the method.

On the other hand, the research in [10] examines the use of multiple CNN models for identifying different diseases in chili. The authors obtain the chili leaf images from the PlantVillage website for use in the research. These models include LeNet, AlexNet, VGG16, VGG19, and ResNets-34. The LeNet model consists of two convolutional layers, and three fully connected layers. For classification tasks, it uses a built-in softmax function in the fully connected layers. In comparison, the AlexNet model includes five convolutional layers, three fully connected layers, and an additional softmax-based layer. As opposed to the AlexNet model, the VGG16 model has sixteen convolutional layers and three fully connected layers, while the VGG19 model has nineteen convolutional layers and three fully connected layers. Both VGG16 and VGG19 models use a built-in softmax function in the fully connected layers, similar to the LeNet model. For the ResNets-34 model, it consists of thirty-four convolutional layers, which are followed by fully connected layers and a softmax-based layer for classification tasks. It is reported that the Resnet-34 model obtains the highest validated accuracy compared to the rest of the CNN models, yielding an accuracy of 94.7%.

Overall, the use of CNNs for computer vision-based application, particularly in the identification of diseases in chili, is showing promising results and continues to remains a field of active research. Despite the emergence of various strains of chili diseases, the CNNs remain effective in identifying the diseases and can be improved to increase the accuracy of the results. In this research, more pretrained CNNs are proposed as feature extractors

for identifying diseased and healthy chili leaf images in order to improve disease identification. Three pretrained CNN models, DenseNet-201, EfficientNet-b0, and NasNet-Mobile, are utilized in the research, where their identification performances on diseased and healthy chili leaf images are compared and evaluated. In order to measure each identification performance, five performance indexes are computed from the confusion matrix generated from the model testing result that are accuracy, recall, specificity, precision as well as F1-score. These indexes are validated through a k-fold cross-validation method in order to find the model with the highest identification performance. The results of the experiments and the designated CNN models are obtained using Matlab™ software version 9.11, R2021b on a laptop with an Intel® Core™ i5 processor operated at 3.4 GHz.

The contributions of this research can be summarized as follows: (a) introduction of pretrained CNN models for identifying diseased and healthy chili leaves, (b) an analysis of comparison of the performance of pretrained CNN models as feature extractors in identifying diseased and healthy chili leaves, and (c) cross-validating the performance of the pretrained CNN models to reduce bias and generalization errors from the underlying chili leaf dataset. The rest of the paper is organized in the following manner. The description of the chili leaf dataset used in the research is explained in Section 2.0. The section also delves into the various layers of the architectures of the DenseNet-201, EfficientNet-b0, and NasNet-Mobile models. The details of the experiments performed are also presented. In Section 3.0, the evaluation results are shown, and a discussion of the results is provided. The research conclusion is presented in Section 4.0.

2.0 METHODOLOGY

2.1 Chili Leaf Dataset

To reduce the burden for farm owners and crop pathologists to document and track crop diseases, a team of researchers has created a dataset of crop diseases, including those that affect chili. This dataset is available on the PlantVillage website and is intended to help reduce the workload for farm owners and crop pathologists by providing a comprehensive resource for information on various diseases, including chili diseases.

In this research, the chili leaf dataset is acquired from the PlantVillage website, comprising a total of 3000 images, with 1000 images belonging to the healthy category, 1000 images of which are in the category of Bacterial Leaf Spot disease, and 1000 images of which are in the category of Powdery Mildew disease. Each of the images in the chili leaf dataset has an initial resolution of 256 x 256 pixels. These images are downsized to 224 x 224 pixels to fit the dimensional image input of utilized CNN models, and to reduce the processing device workload during the model training process. Figure 1 to Figure 3 present examples of the images from the utilized chili leaf dataset.

Figure 1 Healthy chili leaf

Figure 2 Bacterial Leaf Spot

Figure 3 Powdery Mildew

2.2 Developments of a CNN Model

A CNN model architecture typically consists of a series of layers that extract, process, and classify an input leaf image. These layers can be broadly classified into several categories, including convolutional, activation, pooling, and fully-connected layers [11]. In the CNN model architecture, a convolution layer is the fundamental layer [12]. The convolution layer goal is to extract features from an input leaf image. It has a set of learnable kernels or filters, f that activate when the filters discover available features in the input leaf image. The filters have two parameters: weight, W^f and bias, b^f. Each filter is employed to the image

pixel values in a sliding window fashion, taking into account the blue, red, and green color channels, and computing the dot product between the weight and the input pixel, x. This produces a 2-dimensional activation map of the filter called the feature map, h^{f}, as shown in Equation (1).

$$h^f = f(W^f * x + b^f) \quad (1)$$

Afterwards, the Rectified Linear Unit (ReLU) layer is used to apply a non-linear function to the feature map, where every negative value from the feature map is removed and replaced with a zero value. In contrast, every positive value in the feature map remains. This makes it possible for the model to use fewer computational resources from a processing device. The formula of non-linear function is given by Equation (2).

$$f(x)_{ReLU} = max(0, x) \quad (2)$$

Next, a pooling layer processes the feature map, reducing the feature map size to lessen the number of parameters that must be computed by the model network. There are various pooling types, like as average pooling and maximum pooling, which, respectively, extract the value of average or maximum from a region of the feature map. The pooled feature map comes into a vector form and is then inserted into a fully connected layer.

As a layer activation connector, the fully connected layer links all other layers together to be in a same path in order to prepare for the classification step. The pooled feature map is converted by the fully connected layer into an output, which is then sent to the softmax-based layer.

The output is compressed into a distribution of probability over the potential classes in the softmax-based layer. The softmax function can be utilized to determine the probability that a given input is associated to a specific class. This probability is calculated by applying the softmax function, as shown in Equation (3).

$$\sigma(\vec{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \quad (3)$$

where σ is the softmax, $\vec{x}$ is the input vector, and $e^{x_i}$is the standard exponential function for the input vector. Meanwhile, K is the number of classes, and e^{x_j} is the standard exponential function for the output vector. In the following section, the motivation for selecting a CNN model in this research, and its architectural details will be described and explained in more details.

2.3 Utilized CNN Models

As more computational resources become available due to technological advancement in processing devices, CNN models are scaled over the years to increase their learning capabilities. When scaling a CNN model, three dimensions are changed: depth, width, and resolution [13]. The depth refers to the overall number of convolutional layers, while the width refers to the number of filters in every convolutional layer. Finally, the resolution is merely the dimensions of the input image.

A CNN model can learn more complex features of an image by adding more convolutional layers, e.g., by increasing the depth, but deeper models tend to suffer from vanishing gradients [14]. Vanishing gradients refer to the information of the features disappearing before reaching their destination due to the longer path between the convolutional layer and the fully classification layer. This problem causes the learning process to degrade over time, resulting in lower accuracy. Many CNN models, such as ResNet [15], and FractalNet [16], are introduced to address the problem by using skip connections that create short paths to pass information between layers. The drawback of this solution is that it limits the information representation capabilities [17]. Therefore, the DenseNet-201 model is used in this research, which mitigates the representation problem by using multi-layer feature concatenation [18] in its network.

Respectively, by increasing the width of a CNN model, the layers can learn more fine-grained features. In fact, this approach has been used in the Wide ResNet [19] model. Nonetheless, in contrast to scaling up the depth, increasing only the width inhibits the model from learning complex features. In addition, higher image resolution allows a CNN model to extract finer features but, on its own, returns a big computational resource as well [20]. Still, balancing all three dimensions where one of the dimensions is scaled up is a difficult task. In fact, the process often requires many attempts to scale up the dimensions appropriately to satisfy the computational resource constraints. The use of the EfficientNet-b0 model in this research starts with a compound coefficient [21] that can scale all three dimensions equally, which gives the model better learning capabilities and allows the use of fewer computational resources.

The main block of the EfficientNet-b0 model, the Mobile Inverted Bottleneck Convolution (MBConv), is inspired by the construction method of its predecessor, NasNet-Mobile, whose model is developed using the neural architecture search space [22] to determine the dimensions and connections of each layer in order to minimize real-world processing latency on mobile devices. Therefore, along with the EfficientNet-b0 and DenseNet-201 models, the NasNet-Mobile model is also considered for use in this research.

2.3.1 DenseNet-201 Model Architecture

In order to accomplish the feature concatenation, a down-sampling of the feature map is required. Therefore, dense blocks are essential in order to achieve down-sampling. The DenseNet-201 model in

this research has four dense blocks, each with a growth rate of 2. Before entering the first dense block, the input leaf image is passed through a convolutional layer with a kernel size of 7 x 7 and a max pooling layer with a kernel size of 3 x 3. These layers are used to maintain the size of the feature map consistent as it is passed through the network.

Between successive dense blocks, transition layers are used that consist of a convolution layer with a kernel size of 1 x 1, and an average pooling layer with a kernel size of 2 x 2. These layers are used to reduce the size of the feature map, and to facilitate the down-sampling process. At the end of the network, there is a Global Average Pooling layer that reduces the dimensionality size of the feature map. This is followed by fully connected and softmax-based layers that are used for classification. Figure 4 illustrates the architecture of the DenseNet-201 model, and its details are presented in Table 1.

Table 1 Details of the Densenet-201 model architecture

Architecture	Kernel size and type	Resolution
Convolution	7 × 7 conv, stride 2	224 × 224 × 32
	3 × 3 max pool, stride 2	56 × 56 × 64
Dense block 1	$\begin{Bmatrix}1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv}\end{Bmatrix} \times 6$	56 × 56 × 256
Transition 1	1 × 1 conv	28 × 28 × 128
	2 × 2 average pool, stride 2	
Dense block 2	$\begin{Bmatrix}1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv}\end{Bmatrix} \times 12$	28 × 28 × 512
Transition 2	1 × 1 conv	14 × 14 × 256
	2 × 2 average pool, stride 2	
Dense block 3	$\begin{Bmatrix}1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv}\end{Bmatrix} \times 24$	14 × 14 × 1792
Transition 3	1 × 1 conv	7 × 7 × 896
	2 × 2 average pool, stride 2	
Dense block 4	$\begin{Bmatrix}1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv}\end{Bmatrix} \times 16$	7 × 7 × 1920
Global Average Pooling	7 × 7 average pool, stride 2	1 × 1 × 1920
Classification	fully connected, softmax	1 × 1 × 1000

2.3.2 EfficientNet-b0 Model Architecture

The main building block for the EfficientNet-b0 model in this research is the MBConv block. The MBConv block is made to effectively increase the network depth and width while preserving a small model size. Before entering the first MBConv block, the input leaf image is passed through a convolutional layer with a kernel size of 3 x 3 to maintain the size of the feature map consistent as it is passed through the network. The first MBConv block then conducts a 1 x 1 convolution to expand the feature map, followed by a spatial convolution with a kernel size of k3 x k3. The successive MBConv blocks apply a spatial convolution with a kernel size of either k3 x k3 or k5 x k5. The last MBConv block is followed by convolution 1 x 1 and pooling layers with a kernel size of 7 x 7 to restore the original dimensions of the feature map. At the end of the network, fully connected and softmax-based layers are attached in order to perform classification. Figure 5 illustrates the architecture of the EfficientNet-b0 model, and its details are presented in Table 2.

Table 2 Details of the Efficientnet-b0 model architecture

Architecture	Kernel size and type	Resolution
Convolution	3 × 3 conv, stride 2	224 × 224 × 32
MBConv1, block=1	1 × 1 conv, stride 2 k3 × k3, spatial	112 × 112 × 16
MBConv6, block=2	k3 × k3, spatial	112 × 112 × 24
MBConv6, block=2	k5 × k5, spatial	56 × 56 × 40
MBConv6, block=3	k3 × k3, spatial	80 × 80 × 28
MBConv6, block=3	k5 × k5, spatial	192 × 192 × 14
MBConv6, block=2	k5 × k5, spatial	112 × 112 × 14
MBConv6, block=1	k3 × k3, spatial	320 × 320 × 7
Convolution	1 × 1 conv, stride 2	7 × 7 × 1280
Pooling	7 × 7 average pool, stride 2	1 × 1 × 1280
Classification	fully connected, softmax	1 × 1 × 1000

2.3.3 NasNet-Mobile Model Architecture

The NasNet-Mobile model in this research is obtained through the neural architecture search process, which involves searching through a space of possible architectures. In this case, the search is performed on a convolutional cell, which can be classified as either a Normal Cell or a Reduction Cell. The Normal and Reduction cells are used to increase the depth of the network and reduce the spatial resolution of the input leaf image, respectively. These cells consist of multiple layers, each of which can perform different types of operations such as regular convolutions, depthwise convolutions, max pooling, and average pooling. At the end of the network, fully connected and softmax-based layers are attached in order to perform classification. Figure 6 illustrates the architecture of the NasNet-Mobile model, and its details are presented in Table 3.

Table 3 Details of the NasNet-Mobile model architecture

Architecture	Kernel size and type	Resolution
Normal cell	7 × 1 conv, stride 2	224 × 224 × 32
	3 × 3 average pool, stride 2	72 × 72 × 64
	5 × 5 max pool, stride 2	56 × 56 × 59
	1 × 1 conv, stride 2	112 × 112 × 56
Reduction cell 1	3 × 3 conv, depthwise	357 × 357 × 48
	7 × 7 conv, depthwise	752 × 752 × 40
	3 × 3 conv, stride 2	437 × 437 × 35
	3 × 3 conv, dilated	264 × 264 × 28
Reduction cell 2	3 × 3 max pool, stride 2	153 × 153 × 14
	7 × 7 conv, stride 2	364 × 364 × 7
	1 × 1 conv, stride 2	7 × 7 × 1280
	7 × 7 average pool, stride 2	1 × 1 × 1280
Classification	fully connected, softmax	1 × 1 × 1000

2.4 Experimental Setup

The experiments begin with three CNN models, namely DenseNet-201, EfficientNet-b0, and NasNet-Mobile are used to identify healthy and diseased chili leaf images from the utilized chili leaf dataset. These models are trained on a total of 2100 images from the dataset, and the remaining 900 images are utilized to test the models. After the training and testing process of each model is completed, the confusion matrix generated from a model testing result is used to measure the identification performance, while a k-fold cross-validation method is used to validate the identification performance. The validated identification performance of each model is then compared to determine which model has the highest identification performance. The setup of the experiments will be described further in the following section.

2.4.1 Hyperparameter Setting

Hyperparameters are parameters whose values are set before training a CNN model in order to control the learning process of the model. These hyperparameters are the learning rate, epoch, testing interval, momentum, batch size, and optimizer, which values in this research are fixed according to [23]. The learning rate is applied to control the speed at which a CNN model learns. In contrast, the epoch indicates the frequency at which a training set is sent to train the model. When it comes to the testing interval, the interval refers to the number of testing attempts conducted during the training process. Next, momentum accelerates the learning rate, and the batch size indicates the training image amount that propagates along the model network. Conversely, the optimizer adjusts the model weight during training to minimize error. The fixed values of these hyperparameters are shown in Table 4.

Table 4 Hyperparameter values for CNN models

Hyperparameter	Value
Learning rate	0.1
Epoch	30
Testing interval	20
Momentum	1.0
Batch size	64
Optimizer	Stochastic Gradient Descent with Momentum (SGDM)

2.4.2 K-fold Cross-validation

For identification performance, there is a need to validate how accurately a CNN model can perform in practice. A k-fold cross-validation method is used to validate each identification performance index from the result of the testing set of a CNN model. This research uses the value of k = 5, which is the standard k value for cross-validation [24, 25]. In the experiments, both the training and testing sets are partitioned into five folds and randomly shuffled. There are five iterations, and in the first iteration, the first fold is set aside to test the model, while the remaining four folds are composed to train the model.

After the testing result is generated, the second fold is set aside to test the model, while the remaining four folds are composed again to train the model. This process continues until all five folds are used to test the model. At the end of all the iterations, each identification performance index is obtained by averaging the accumulated index scores derived from the testing result through all the iterations.

2.4.3 Index of Identification Performance

The accuracy, recall, specificity, precision, and F1-score are the indexes used to assess the identification performance of the utilized CNN models in the research. These indexes are computed from the confusion matrix generated according to the testing result produced by the models, which are True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) [26].

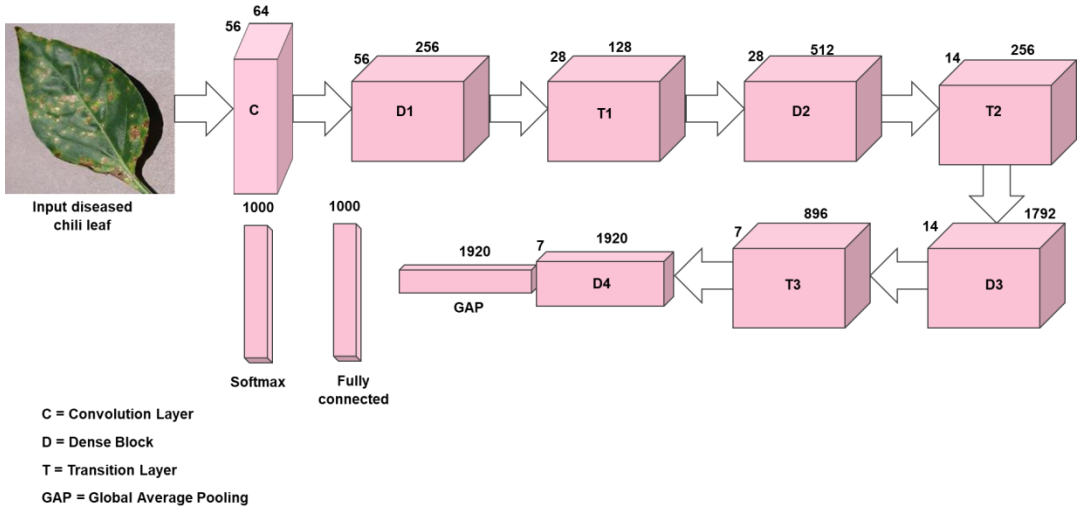

Figure 4 DenseNet-201 model architecture

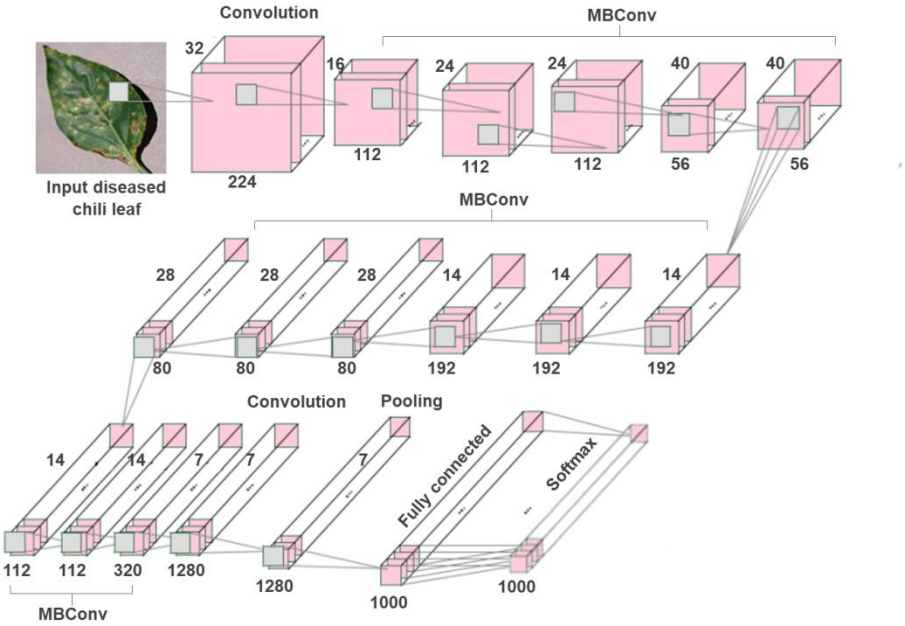

Figure 5 EfficientNet-b0 model architecture

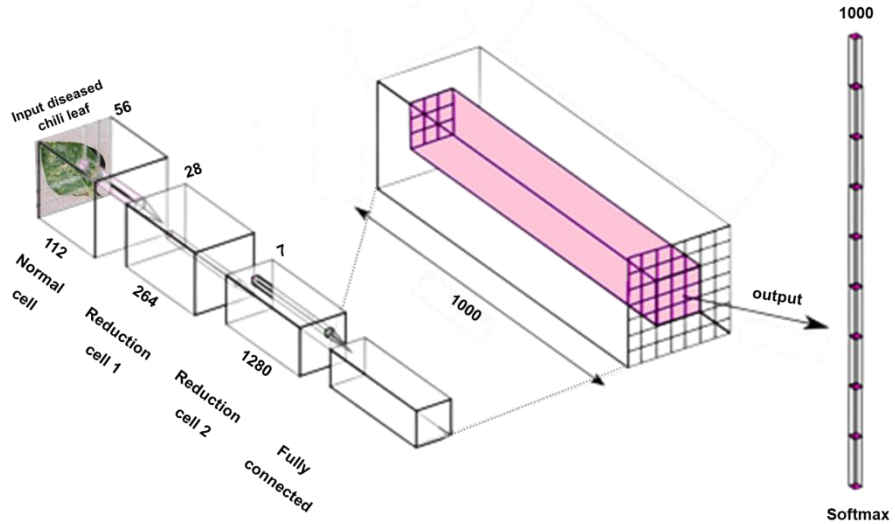

Figure 6 NasNet-Mobile model architecture

Accuracy is the measure of a likelihood of correct prediction made by a CNN model. This index is the ratio of true positive and true negative predictions to the total number of predictions. The index is mostly used when evaluating the stability of a CNN model. Theoretically, accuracy can be calculated by using Equation (4) as follows:

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} x100\% \quad (4)$$

In contrast, recall is the percentage of a true positive prediction of the total frequency of true positive and false negative predictions. The index measures the proportion of true prediction on the diseased leaf case. The index can be computed using Equation (5) as follows:

$$Recall = \frac{TP}{TP+FN} x100 \quad (5)$$

For specificity, it is the vice versa of recall, where the index represents the percentage of a true prediction on a negative diseased leaf case. The index is expressed as given in Equation (6) as follows:

$$Specificity = \frac{TN}{FP+TN} x100\% \quad (6)$$

Whereas precision gives feedback on the capability of a CNN model to give a positive result on a real diseased leaf image from all the positively predicted diseased leaf images. The index can be obtained with the following formula given in Equation (7) as follows:

$$Precision = \frac{TP}{TP+FP} x100\% \quad (7)$$

On the other hand, F1-score conveys the balance of the prediction in terms of precision and recall. The value of the F1-score can be obtained with the following Equation (8) as follows:

$$F1 = 2x \frac{Precision*Recall}{Precision+Recall} \quad (8)$$

3.0 RESULTS AND DISCUSSION

Following the implementation of the experimental setup in Section 2.4, the experimental results in this research are divided into two parts: an analysis of the model training progress, and an analysis of the model identification performance.

3.1 Model Training Progress

Softmax-based layers enable CNN models to be applied to the visualization of training progress, where their training accuracy and learning loss are displayed as these models are trained to completion. Figure 7 to Figure 9 depict the training progress of the CNN models when fed with the chili leaf datasets.

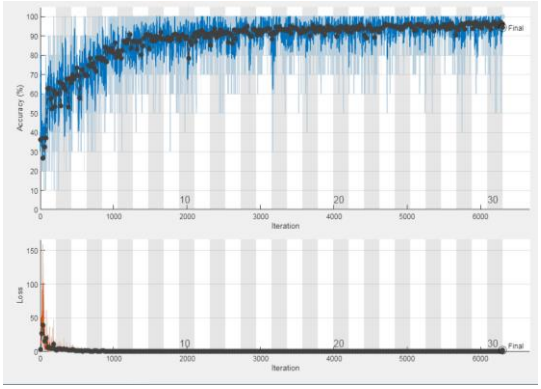

Figure 7 Training progress of DenseNet-201 model

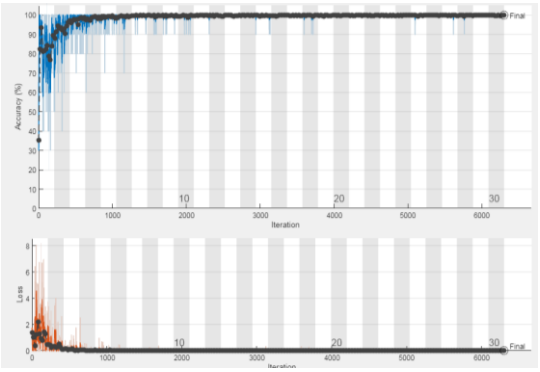

Figure 8 Training progress of EfficientNet-b0 model

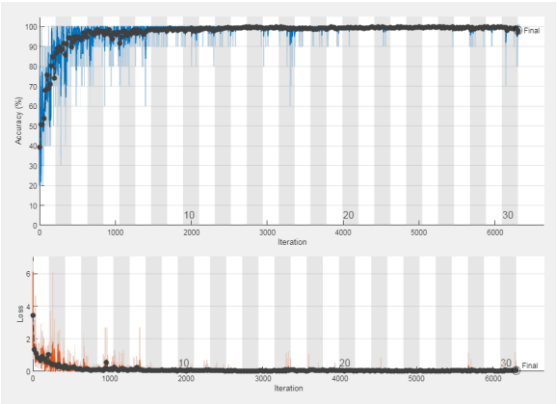

Figure 9 Training progress of NasNet-Mobile model

Firstly, in Figure 7, it can be seen that the training progress of DenseNet-201 model shows a substantial training undershoot at the beginning of epoch 10, before stabilizing at epoch 30 until training is completed. The learning loss of the model is also very high at the beginning of epoch 0, before it gradually decreases after epoch 4 until training is completed.

Secondly, for Figure 8, EfficientNet-b0 model records a substantial training undershoot at epoch 1. Then, in epoch 4, the training begins to stabilize and reaches its optimum starting from epoch 6 till the completion of training. For the learning loss, the model exhibits an unstable learning loss between epochs 1 and 2, before stabilizing at epoch 6 till the

completion of training. Finally, it can be seen in Figure 9 that the NasNet-Mobile model has no substantial training fluctuations. The model manages to reach its optimum training starting at epoch 8 until epoch 30. Meanwhile, the model learning loss remains relatively stable, starting from epoch 2 to the end of the training, with a small overshoot recorded at epoch 5.

3.2 Model Identification Performance

The testing process is then carried out to verify each of the trained models, where the resulting confusion matrix is accustomed to measure the model identification performance. The derivation of TP, TN, FP, and FN, can be made based on the matrix. The matrix basically has a table structure, with each row representing instances of an actual class and each column representing instances of a predicted class.

There are four possible outcomes that arise in each column. Firstly, suppose the actual identification is positive, and the projected identification is positive (1,1). In this case, this is referred to as a TP result, as the model accurately identifies the positive image.

Secondly, suppose the actual identification is positive, but the projected identification is negative (1,0). In this case, this is referred to as a FN result, as the model wrongly labels the positive image as negative.

Thirdly, suppose the actual identification is negative, and the projected identification is positive (0,1). In this case, this is referred to as a FP result, as the model wrongly identifies the negative image as positive.

Lastly, if the actual identification is negative and the projected identification is negative (0,0), this is referred to as a TN result since the model accurately identifies the negative image. The confusion matrix resulting from the testing process of each CNN model is shown in Figure 10 to Figure 12. The identification performance index of each pretrained CNN-based model is shown in Table 5.

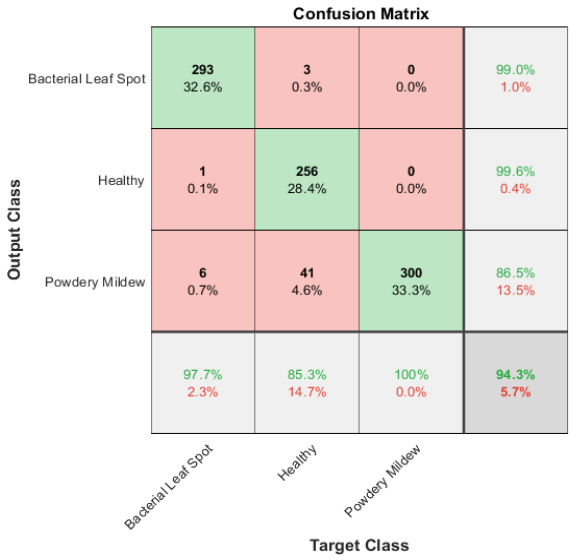

Figure 10 DenseNet-201 model confusion matrix

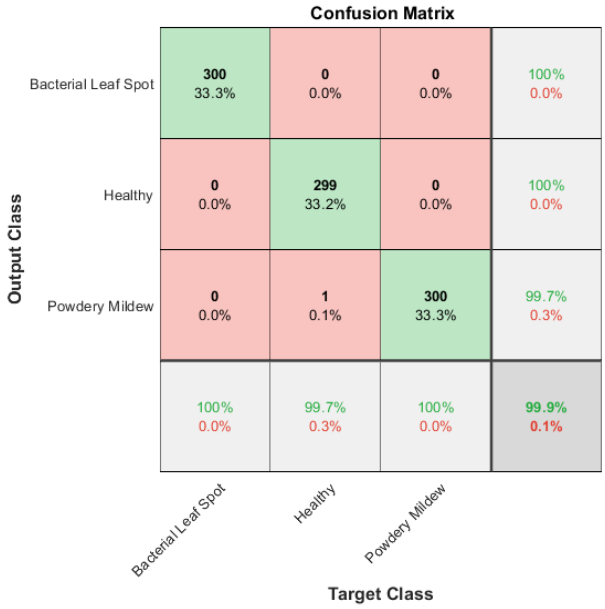

Figure 11 EfficientNet-b0 model confusion matrix

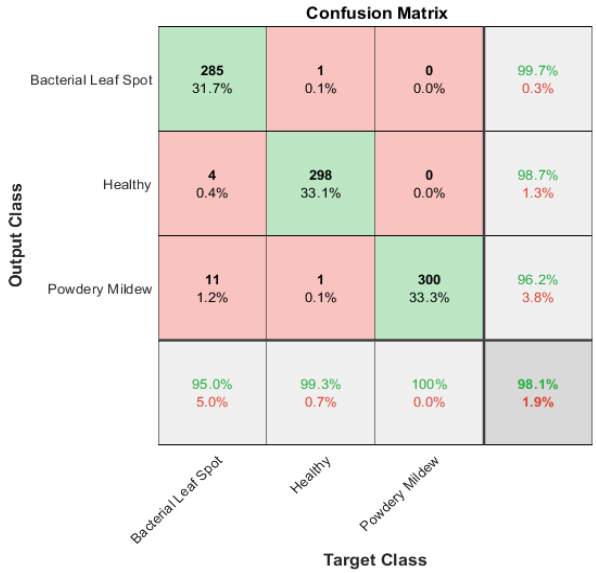

Figure 12 NasNet-Mobile model confusion matrix

Table 5 Overall performance index for each CNN model

No.	Performance Index	CNN Model		
		DenseNet-201	EfficientNet-b0	NasNet-Mobile
1.	Accuracy	94.33	99.89	98.12
2.	Recall	1.00	1.00	1.00
3.	Specificity	0.92	1.00	0.98
4.	Precision	0.86	1.00	0.96
5.	F1-score	0.93	1.00	0.98

The validated performance of each CNN model using a 5-fold cross-validation method is shown in Table 6 to Table 8, with the overall validated performance is summarized in Table 9.

Table 6 Densenet-201 model across 5-fold cross-validation

Fold	Identification Performance Index				
	Accuracy	Recall	Specificity	Precision	F1-score
1	92.39	1.00	0.94	0.89	0.94
2	93.48	0.98	0.87	0.77	0.86
3	91.27	0.95	0.88	0.84	0.89
4	94.83	1.00	0.91	0.81	0.90
5	93.08	0.92	0.90	0.83	0.87
Mean	93.01	0.97	0.90	0.83	0.89

Table 7 EfficientNet-b0 model across 5-fold cross-validation

Fold	Identification Performance Index				
	Accuracy	Recall	Specificity	Precision	F1-score
1	98.98	0.99	0.93	0.93	0.96
2	98.98	0.98	0.96	0.91	0.94
3	97.90	0.98	0.91	0.89	0.93
4	96.62	0.96	0.89	0.94	0.95
5	92.79	0.94	0.92	0.94	0.94
Mean	97.05	0.97	0.92	0.92	0.94

Table 8 NasNet-Mobile model across 5-fold cross-validation

Fold	Identification Performance Index				
	Accuracy	Recall	Specificity	Precision	F1-score
1	98.98	0.99	0.93	0.93	0.96
2	98.98	0.98	0.96	0.91	0.94
3	97.90	0.98	0.91	0.89	0.93
4	96.62	0.96	0.89	0.94	0.95
5	92.79	0.94	0.92	0.94	0.94
Mean	97.05	0.97	0.92	0.92	0.94

Table 9 Overall validated performance index for each CNN model

No.	Validated Performance Index	CNN Model		
		DenseNet-201	EfficientNet-b0	NasNet-Mobile
1.	Accuracy	93.01	97.05	96.50
2.	Recall	0.97	0.97	0.96
3.	Specificity	0.90	0.92	0.90
4.	Precision	0.83	0.92	0.91
5.	F1-score	0.89	0.94	0.93

As shown in Table 9, the Efficientnet-b0 model obtains the highest index of accuracy at 97.05%. This indicates that the model is able to correctly identify the majority of the chili leaf images in the dataset. In addition to its high accuracy, the Efficientnet-b0 model also achieves the highest scores in other performance indexes, including recall, specificity, and F1-score, which are 0.97, 0.92, 0.92, and 0.94, respectively. These results indicate that EfficientNet-b0 is the most effective model in terms of minimizing FP (incorrectly classifying healthy leaves as diseased) and FN (incorrectly classifying diseased leaves as healthy).

On the other hand, the NasNet-Mobile model is also performing well in this research, with an accuracy index of 96.50% and validated values for the index of recall, specificity, precision, and F1-score of 0.96, 0.90, 0.91, and 0.93, respectively. While its performance is not as strong as EfficientNet-b0 in some indexes, the model still demonstrates good overall performance in the task of identifying healthy and diseased chili leaf images. In contrast, the Densenet-201 model achieves the lowest accuracy index of 93.01% and the lowest values for the index of recall, specificity, precision, and F1-score of 0.97, 0.90, 0.83, and 0.89, respectively. These results suggest that Densenet-201 may not be as effective as the other models in this particular identification task.

3.3 Accuracy in Identifying Chili Diseases

Accurate disease identification plays a crucial role in chili cultivation, enabling early intervention and preventing the spread of diseases that can adversely affect chili yields. The overall accuracy in identifying chili diseases using pretrained CNN models is shown

in Table 10. Based on Table 10, the EfficientNet-b0 model demonstrates the highest accuracy index for each type of chili leaf: 98.07% for healthy leaves, 97.05% for leaves with Bacterial Leaf Spot, and 96.03% for leaves with Powdery Mildew. The consistent attainment of accuracy levels exceeding 90% across all pretrained CNN models for each chili leaf type underscores the potential for their integration into practical agricultural disease management applications.

Table 10 Overall accuracy of chili diseases for each CNN model

No.	Type of Chili Leaf	Overall CNN Model Accuracy		
		DenseNet-201	EfficientNet-b0	NasNet-Mobile
1.	Healthy	95.44	98.07	97.48
2.	Bacterial Leaf Spot	93.32	97.05	96.39
3.	Powdery Mildew	90.27	96.03	95.48

4.0 CONCLUSION

In this research, the use of pretrained CNN models is proposed as a feature extraction method for identifying diseased and healthy chili leaf images in order to improve disease identification. Three pretrained CNN models, DenseNet-201, EfficientNet-b0, and NasNet-Mobile, are utilized for their ability to identify chili diseases using five indexes: accuracy, recall, specificity, precision, and F1-score. These indexes are validated through a five-fold cross-validation method during the experiments. The experimental results have shown that the EfficientNet-b0 model achieved the highest identification performance, with indexes of accuracy, recall, specificity, precision, and F1-score of 97.05%, 0.97, 0.92, 0.92, and 0.94, respectively.

There are several challenges and limitations encountered in this research. One major challenge is the limited availability of high-quality chili leaf images for training and testing the CNN models. Accurate identification of chili diseases requires a sufficient amount of representative data, and the availability of such data may have been a challenge. Additionally, the process of training and testing the CNN models on the chili leaf dataset requires significant computational resources, which may have been a challenge as well.

Given these challenges and limitations, there are several potential directions for future work. One possibility is to improve the performance of the CNN models through boosting the size and diversity of the chili leaf dataset. This could involve collecting additional chili leaf images or expanding the size of dataset artificially by using data augmentation methods. Another possibility is to improve the efficiency of the CNN models by using more efficient training algorithms that require fewer computational resources. Techniques such as model parallelism or distributed training could be used to train the models using multiple processing devices, which could significantly reduce the time required to train the models.

Conflicts of Interest

The author(s) declare(s) that there is no conflict of interest regarding the publication of this paper.

Acknowledgement

This research is supported by Universiti Tun Hussein Onn Malaysia (UTHM) through Tier1 (Vot 916). Special appreciation to the IoT Focus Group, FKEE, UTHM for providing the related facilities needed for this research.

References

[1] Thakur, H., Jindal, S., Sharma, A., and Dhaliwal, M. 2019. A Monogenic Dominant Resistance for Leaf Curl Virus Disease in Chilli Pepper (Capsicum annuum L.). *Crop Protection*. 116: 115-120. Doi: https://doi.org/10.1016/j.cropro.2018.10.007.

[2] Colney, L., Tyagi, W., and Rai, M. 2018. Morphological and Molecular Characterization of Two Distinct Chilli Cultivars from North Eastern India with Special Reference to Pungency Related Genes. *Scientia Horticulturae*. 240: 1-10. Doi: https://doi.org/10.1016/j.scia.2018.05.045.

[3] Lu, M., Chen, C., Lan, Y., Xiao, J., Li, R., Huang, J., Huang, Q., Cao, Y., and Ho, C. T. 2020. Capsaicin - The Major Bioactive Ingredient of Chili Peppers: Bio-Efficacy and Delivery Systems. *Food & Function*. 11(4): 2848-2860. Doi: https://doi.org/10.1039/d0fo00351d.

[4] Dhingra, G., Kumar, V., and Joshi, H. D. 2017. Study of Digital Image Processing Techniques for Leaf Disease Detection and Classification. *Multimedia Tools and Applications*. 77(15): 19951-20000. Doi: https://doi.org/10.1007/s11042-017-5445-8.

[5] Özaydın, U., Georgiou, T., and Lew, M. 2019. A Comparison of CNN and Classic Features for Image Retrieval. *Contemporary Multimedia Index*. 1-4. Doi: https://doi.org/10.48550/arXiv.1908.09300.

[6] Sufola, D., Malemathh, V., and Sundaram, K. M. 2021. Affine Invariant Approach for Disease Detection on Chili Plant. *Electrical, Computer and Energy Technologies*. Doi: https://doi.org/10.1109/icecet52533.2021.9698472.

[7] Sari, Y., Baskara, A. R., and Wahyuni, R. 2021. Classification of Chili Leaf Disease Using the Gray Level Co-occurrence Matrix (GLCM) and the Support Vector Machine (SVM) Methods. *Informatics and Computing*. Doi: https://doi.org/10.1109/icic54025.2021.9632920.

[8] Devi, B. M., and Amarendra, K. 2020. Machine Learning Based Application to Detect Pepper Leaf Diseases Using HistGradientBoosting Classifier with Fused HOG and LBP Features. *Engineering Research*. 8(10): 7371-7376. Doi: https://doi.org/10.30534/ijeter/2020/1148102020.

[9] Sahuri, G., and Rosalina. 2020. Implementation of Deep Learning Methods in Detecting Disease on Chilli Leaf. *Advances in Studies in Computation*. 9(6): 10-15.

[10] Deeba, K., and Amutha, B. 2020. ResNet - Deep Neural Network Architecture for Leaf Disease Classification.

Microprocessors and Microsystems. Doi: https://doi.org/10.1016/j.micpro.2020.103364.
[11] Francis, M., and Deisy, C. 2019. Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks - A Visual Understanding. *Signal Processing and Integrated Networks*. Doi: https://doi.org/10.1109/spin.2019.8711701.
[12] Alzubaidi, L., Zhang, J., Humaidi, A., J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M., A., Al-Amidie, M., and Farhan, L. 2021. Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions. *Journal of Big Data*. 8(1). Doi: https://doi.org/10.1186/s40537-021-00444-8.
[13] Colaco, S., J., and Han, D., S. 2022. Deep Learning-based Facial Landmarks Localization using Compound Scaling. Access. IEEE. 10: 7653-7663. Doi: https://doi.org/10.1109/access.2022.3141791.
[14] Jha, D., Gupta, V., Ward, L., Yang, Z., Wolverton, C., Foster, I., Liao, W. K., Choudhary, A., and Agrawal, A. 2021. Enabling Deeper Learning on Big Data for Materials Informatics Applications. *Scientific Reports*. 11(1). Doi: https://doi.org/10.1038/s41598-021-83193-1.
[15] Kumar, V., Arora, H., Harsh, and Sisodia, J. 2020. ResNet-Based Approach for Detection and Classification of Plant Leaf Diseases. *Electronics and Sustainable Communication Systems*. Doi: https://doi.org/10.1109/icesc48915.2020.9155585.
[16] Oyedotun, O., K., Ismaeil, K., A., and Aouada, D. 2021. Training Very Deep Neural Networks: Rethinking the Role of Skip Connections. *Neurocomputing*. 441: 105-117. Doi: https://doi.org/10.1016/j.neucom.2021.02.004
[17] Jiang, X., Xiao, Z., Zhang, B., Zhen, X., Cao, X., Doermann, D., and Shao, L. 2019. Crowd Counting and Density Estimation by Trellis Encoder-Decoder Networks. *Computer Vision and Pattern Recognition*. 6133-6142.
[18] Demir, F., Turkoglu, M., Aslan, M., and Sengur, A. 2020. A New Pyramidal Concatenated CNN Approach for Environmental Sound Classification. *Applied Acoustics*. DOI: https://doi.org/10.1016/j.apacoust.2020.107520
[19] Acharya, A., Muvvala, A., Gawali, S., Dhopavkar, R., Kadam, R., and Harsola, A. 2020. Plant Disease Detection for Paddy Crop Using Ensemble of CNNs. *Innovation in Technology*.
Doi: https://doi.org/10.1109/inocon50539.2020.9298295.
[20] Thambawita, V., Strümke, I., Hicks, S. A., Halvorsen, P., Parasa, S., and Riegler, M. A. 2021. Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images. *Diagnostics*. 11(12): 2183. Doi: https://doi.org/10.3390/diagnostics11122183.
[21] Tan, M., and Le, Q. 2019. Efficientnet: Rethinking Model Scaling for Convolutional Neural Networks. *International Conference on Machine Learning*. 6105-6114. Doi: https://doi.org/10.48550/arXiv.1905.11946.
[22] Adedoja, A. O., Owolawi, P. A., Mapayi, T., and Tu, C. 2022. Intelligent Mobile Plant Disease Diagnostic System Using NASNet-Mobile Deep Learning. *Computer Science*. 1-16.
[23] Setyanto, A., Kusrini, K., Sasongko, T. B., Permana, A. B., and Saputra, A. P. 2021. Efficient Deep Learning Architecture for Facemask Detection. Information and Communications Technology.
Doi: https://doi.org/10.1109/icoiact53268.2021.9564011.
[24] Baron, G., and Stańczyk, U. 2021. Standard vs. Non-Standard Cross-Validation: Evaluation of Performance in a Space with Structured Distribution of Datapoints. *Procedia Computer Science*. 192: 1245-1254.
Doi: https://doi.org/10.1016/j.procs.2021.08.128.
[25] Ghaffari, M., Monneret, A., Hammon, H., Post, C., Müller, U., Frieten, D., Gerbert, C., Dusel, G., and Koch, C. 2022. Deep Convolutional Neural Networks for the Detection of Diarrhea and Respiratory Disease in Preweaning Dairy Calves Using Data from Automated Milk Feeders. *Dairy Science*. 105(12): 9882-9895. Doi: https://doi.org/10.3168/jds.2021-21547.
[26] Yu, X., Gong, Q., Chen, C., and Lu, L. 2022. Automatic Diagnosis of Soybean Leaf Disease by Transfer Learning. *American Journal of Biochemistry and Biotechnology*. 18(2): 252-260.
Doi: https://doi.org/10.3844/ajbbsp.2022.252.260