Implementation of CNNs for Crop Diseases Classification: A Comparison of Pre-trained Model and Training from Scratch

Priyanka Sahu[1†], Anuradha Chug[1†], Amit Prakash Singh[1†], Dinesh Singh[2††] and Ravinder Pal Singh[2††],

[1†]University School of Information, Communication, and Technology
Guru Gobind Singh Indraprastha University, New Delhi, India
[2††]Division of Plant Pathology, Indian Agricultural Research Institute, New Delhi

Abstract
In recent times, the machine learning field has become very progressive and has given impressive results, through the development of advanced Convolutional Neural Networks (CNN). These models simulate the biological behavior of the human-beings, especially, in dealing with the image recognition tasks. To train a deep CNN from scratch, a massive amount of labeled training samples are required that make the training process difficult. Also, the results have not converged properly and become overfitted in nature. Data augmentation is applied to overcome this issue of overfitting. An optimistic substitution is to use the pre-trained networks, prepared previously over a large dataset such as ImageNet. This kind of CNNs can also be deployed using their two alternative approaches, namely, feature extraction and fine-tuning of the pre-trained model. In this paper, 1296 leaf images of bean crops were used to perform the experiment that classifies the diseased or healthy leaf. In order to show the variation in the performance of the CNNs, the network has trained from scratch as well as using pre-trained networks. Experimental results showed that training from scratch performs worst (70% accuracy) over small training data and pre-trained networks (97.06%) gave far better results compared to the previous one. It is observed that the use of pre-trained networks with the tuning of hyperparameters is an optimal choice for training, for small training data set.

Key words:
Deep learning, crop disease, classification, Convolutional Neural Network, VGG16.

1. Introduction

Convolutional Neural Networks (CNNs) have been deployed in the domain of computer vision for a long time [1]–[3]. AlexNet [4] has observed as a benchmark in the area of CNN, after winning the ImageNet challenge (ILSVRC) in 2012. It has efficiently made use of dropout, regularization, and rectified linear unit (ReLU). After this, various architectures were proposed to overcome the research gaps seen previously. Moreover, the outstanding performance of CNNs is observed because of its deep learning architectures [5]–[7], which allows us to extract a discriminate set of features at a higher level of abstraction.

1.1 Convolutional Neural Networks (CNNs)

CNNs are similar to the traditional artificial neural networks (ANNs), comprised of neurons that performed self-optimization using the learning process. As the traditional ANNs have the limitation in terms of computational complexity, especially for image data computations, that's why CNNs are primarily designed to deal with pattern-recognition within the images. Technically, CNNs are used for training and testing of each inserted input image and flow it in the sequence of convolutional layers with kernels (filters), pooling, flatten, dense layers, and finally, use the "softmax function" for classification (output) of some objects in the image with probabilistic values in a range (0-1) [4].

The basic operations in the CNN are as follows [4]:

Convolutional operation:
Convolution operation performs in a combination of convolutional layer + ReLU. The convolutional layer takes input in the form of pixels of an image and regulates the output of neurons that are associated with local regions of a given input. This layer computes the scalar product of the weights of neurons and the region associated with the input. An activation function ReLU is applied after each convolution in order to carry the element-wise activation for the output generated by the convolutional layer.

Pooling operation:
The process of down sampling of the spatial dimensions of the fed input is termed as pooling. It also reduces the number of parameters within the activation.

Fully connected operation:
A fully connected layer performs this operation by generating class scores from the feature maps (activations) that are required for classification.

However, training of a deep CNN from scratch needs to

Manuscript received October 5, 2020
Manuscript revised October 20, 2020
https://doi.org/10.22937/IJCSNS.2020.20.10.26

meet with the following requirements: 1) A bulk of labeled data with expert annotations is required, 2) Extensive memory and computational resources in order to make the training process fast and efficient, 3) Repetitive adjustment in learning parameters to ensure that all layers learned at an equivalent speed.

Furthermore, training a CNN from scratch would be a monotonous and time-consuming task. It also faces the overfitting issues. To overcome this issue, the use of the pre-trained network is another alternative that can be used instead of training from scratch. These pre-trained networks have been trained over the large labeled dataset like ImageNet and then successfully applied to several computer vision works for feature extraction [4], [8], [9]. In this paper, a performance comparison has been made between the two: training a CNN from scratch and using pre-trained models VGG16. This is an important aspect as training from scratch is not practical with limited annotated image data from the agricultural domain. This study conducted a range of experiments for bean leaf disease detection in crops.

The rest of the paper is framed into the following sections. Section 2, gives a brief of literature works. Section 3, discusses the Materials used for crop disease detection. Section 4, shows the training of a CNN from scratch using a small data sample. Section 5 shows training using a pre-trained network, and section 6 shows the comparison of results of both pre-trained and training from scratch CNN. Finally, section 7 concludes the study.

2. Related work

CNNs have been applied to different agricultural domains like crops disease detection, weeds detection, shelf-life prediction, field monitoring, etc. Ferentinos et al. [10] compared five CNN models, namely, GoogLeNet, AlexNet, Overfeat, VGG, and AlexNetOWTbn in which VGG performed best. This experiment was conducted to detect plant diseases for 25 distinct crops in a set of 58 different classes. Fuentes et al. [11] have implemented seven well- known DL models, namely, AlexNet, ResNetXt101, ResNet50, GoogLeNet, ZFNet, ResNet101, and VGG for crop diseases prediction. To compare this study, three detectors were used, namely, SSD, Faster-RCNN, and RFCN. The experimental results showed that ResNet50 with a combination of RFCN performed best with the highest prediction percentage. As technology develops, CNNs have revived with the GPU computational resources and the agricultural domain has seen a new generation of computer vision through its superior performance.

Sa et al. [12] implemented SegNet for weed classification using multispectral imaging captured with the help of micro aerial vehicle (MAV). Six models were trained with a distinct number of input channels and fine-tuning was applied to reach nearly 0.78 area under the curve and 0.8 F1 Score. Jetson TX2 system and Graphics Processing Unit (GPU) were deployed for amalgamation of data gathered using MAV. To predict the shelf-life of fruit with distinct browning levels, Wang et al [13] evaluated the performance of PCA, Threshold algorithm, and Backpropagation (BP) algorithm. Fruit color and reflectance curves were the major extracted features from 2000 samples of hyperspectral images.

In Order to enhance the accuracy to diagnose the results, several researchers have performed the plant disease detection that relies on machine learning and pattern recognition. They made use of pattern recognition, Image processing algorithms, Deep learning techniques, and computer vision. The last decade has observed various applications of deep learning, especially CNNs in different areas of agriculture. Kawasaki et al. [14] implemented a novel architecture relying on CNNs to identify the disease or Infection present in the leaves of cucumber crop. It obtained 94.9% accuracy that marks distinction among zucchini yellow mosaic virus, melon yellow spot virus, and non-diseased category (class).

Liang et al. [15] deployed a multitasking CAD system that is capable to diagnose the disease, identifying the crop species, and there disease severity estimation. This study was performed over the PlantVillage dataset. Stress severity estimation was categorized into three classes: serious, general, and healthy. Outcomes of this experiment gave an overall accuracy of 98% and 91% for plant disease classification and severity estimation of leaf disease.

To detect the diseases in plants at the early stage, various imaging techniques such as thermal imaging [16], multispectral imaging [17], fluorescence, and hyperspectral imaging (HSI) [16] were used. Xie et al. [18] have used HSI for tomato disease detection by identifying the regions of interest. A feature ranking – KNN has been used to give significant results for the classification of healthy and gray mold diseased leave of tomato plants. Chen et al. [19], carried out the leaf spot disease detection for peanuts using canopy hyperspectral reflectance by detecting the sensitive bands. The authors have also used the hyperspectral vegetation indices for the same purpose. Some more related works in detail are shown in Table 1.

Table 1: Literature review of a few DL approaches used along with the crop names, datasets, visualization, and performance metrics.

Deep Learning Architectures	***Datasets***	***Visualization Techniques***	***Metrics***
AlexNet and GoogLeNet[2	Plant-Village	Visualization of neuron activations	CA

0]		in the initial convolutional layer	
CaffeNet [21]	Internet	Visualization of activation maps with filter from initial to final layer	Precision
ZFNet, AlexNet, R-FCN, GoogleNet, SSD, VGG-16, Faster RCNN, ResNet-50, ResNetXt-101[11]	Image captured from real fields	Bounding boxes were used for localization and classification of diseases.	Precision
DCNN[22]	Images were Captured from real field	Feature map to detect the diseases in rice plants	Accuracy
GoogLeNet, AlexNet[23]	PlantVillage	Symptoms visualization approach	Accuracy
CNN, VGG-A[24]	Images Captured from actual field	HSV with K-means clustering	Accuracy
AlexNet[25]	Images were Captured from real field	Feature map to identify the marks of diseases	CA
Student/Teacher architecture[26]	Plant Village	Images with discriminant regions, heat map formation and image segmentation using binary threshold logic	Validation accuracy and loss, Training accuracy and loss
3D- CNN[27]	Real environment	Saliency feature map visualization	F1-score, CA
AlexNet, VGG-16[28]	CASC-IFW	Saliency feature map, mesh graphical image, 2D and 3D contour	CA
Modified LeNet[29]	Plant Village	Segmentation and edge feature map	True positive rate

3. Materials

3.1 Dataset description

The dataset has been taken from GitHub (https://github.com/AI-Lab-Makerere/ibean/). It includes the bean crop leaves images captured from the real field using a smartphone. Examples of the leaf images as per the categorized classes are shown in figure 1. The image annotation was done by the experts (NaCRRI) in order to determine the disease manifestation. Table 2 shows a description of the used dataset. This dataset contains 1296 images split into 3 classes; 80:20 splitting is performed. We have used 3 classes (labels) for the identification of diseases in crops.

Table 2: Bean dataset summarization

Disease Name	***Images count***
Bean Rust	436
Angular Leaf Spot	432
Healthy	428
Total	1296

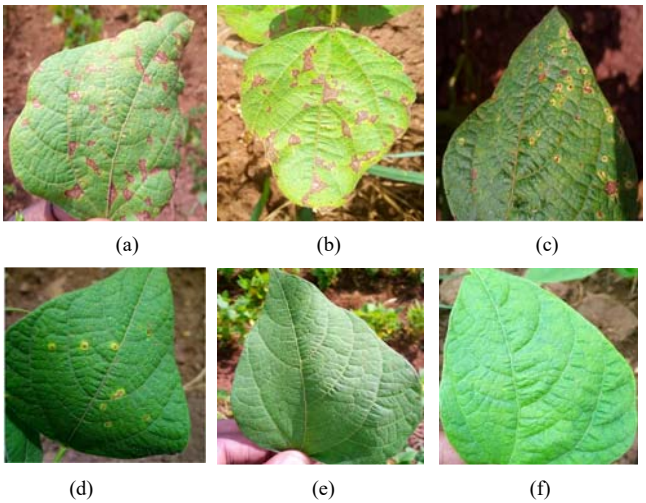


Fig 1. Shows the bean crop images as follows: (a), (b) denotes the Angular Leaf Spot, (c), (d) denotes the bean rust disease, and (e), (f) denotes the healthy leaf images.

4. Training a CNN from scratch using a small data sample

In computer vision, it is very often that small data samples are used to classify the images. Small data means data varies in a range of a few hundred to few thousands of images [30]. In this study, the focus is to classify the images of bean crop leaves (diseased or healthy). This dataset contains 1296 images. A small CNN has been trained with 1035 images from the training set, without using regularization; to carry a baseline for what results can be obtained. Experimental results show that validation accuracy in the range of 70-75% has been achieved and along with it, overfitting comes as the main issue in this basic implementation. To overcome this problem, data

augmentation and dropout have been used to mitigate the overfitting in computer vision. This technique improves accuracy by up to 83.46%. A flow diagram of CNN is shown in figure 2 that depicts the stages of data flowing from data collection to the classification of the bean crop.

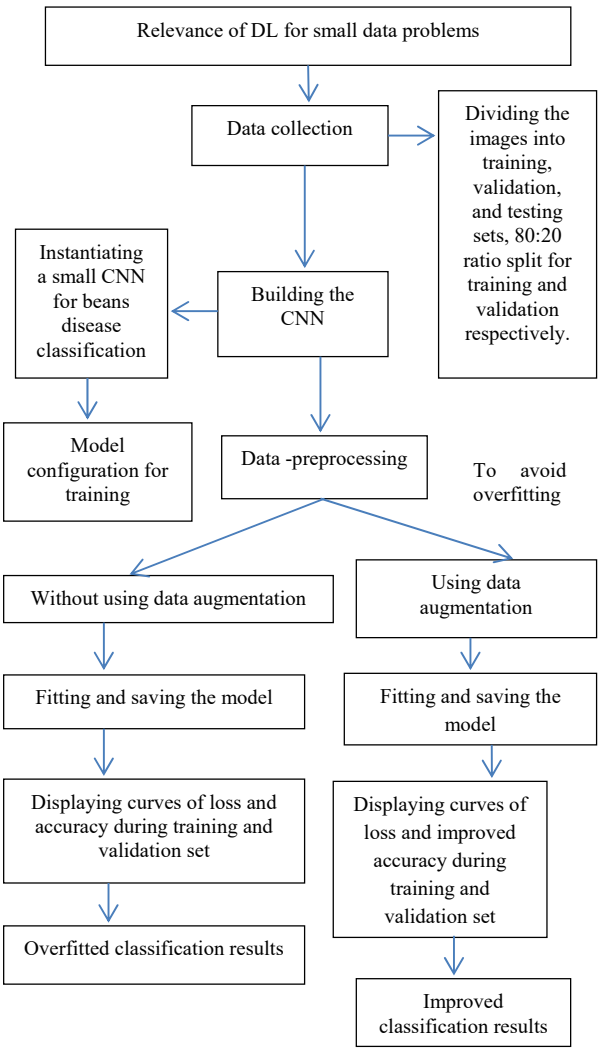


Fig 2.Shows the classification of bean crop using training from scratch.

4.1 The relevance of DL for small data problems

It is a common saying that DL only performs well when lots of input data are available [30]. This can be valid to say, as it is a basic characteristic of DL. It has the ability to learn the features automatically from the training samples, without any manual feature engineering. It can learn only when there are a lot of training samples available. Furthermore, CNNs does not solve the complex problem with only tens of samples. However, hundreds of samples can perform well for a small, simple, and well-regularized model. As CNNs can learn local and translation-invariant features, this being the reason that training a CNN from beginning on a small image data sample still give reasonable outcome despite a relative deficiency of data.

4.2 Data Directories

There were a total of 1296 images present in the data sample of the bean crop leaves (diseased and healthy). This data was divided into three directories:

- Training: It contains 1035 images.
- Validation: it contains 133 images.
- Testing: it contains 128 images.

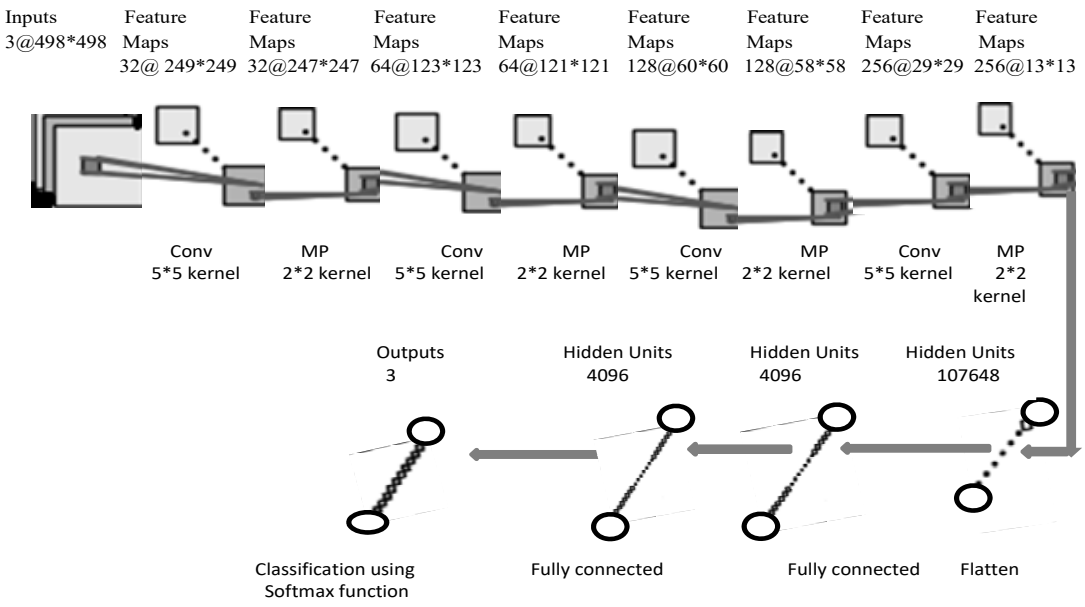
4.3 Building CNN

The CNN is a stack of convolutional layers (Conv2D) and pooling layers (Max-pooling2D), with Conv2D followed by a ReLU activation function. In the training from the scratch procedure, 4 Conv2D + ReLU, 4 max-pooling layers, 1 flatten layer, 2 dense layers, and 1 output layer was chosen to build the network. Subsequently, a flatten layer was placed just after the last max-pooling layer for further reduction of the feature map dimensions from higher to lower. Here, 500*500 pixels of images with bean crop leaves were fed to the CNN. After processing through all the layers in the network, a 29*29 feature map was achieved just before the flattening layer. Now, this flatten layer reduced this vector to a scalar count. As CNN is dealing with the multi-classification problem, a dense layer with value equals to 3 (three classes) was fed to the CNN, and softmax function was deployed at the output layer in order to classify the images of the crop leaves. Figure 3 shows a detailed process of convolution and pooling operations along with their activation maps that are deployed over the bean crop dataset to classify the diseased and healthy crop leaves.

4.4 Data pre-processing

It is used to pre-process the provided data into floating-point vectors, the data readable by CNN. Data pre-processing [31] includes reading of images, decoding image content into RGB, floating-point conversion, rescaling of pixels from 0-255 to a range of 0-1. It results in batches of (20, 500, 500, 3), which means 20 samples were taken at a time, resized image of 500*500 pixels, and 3 represents the RGB.

4.5 Classification without data augmentation



Conv- convolutional layer, MP- max-pooling layer

Fig 3. A detailed CNN with 12 layers, based on Bean Crop disease classification dataset.

*"This figure is generated by adapting the code from https://github.com/gwding/draw_convnet".

Whenever, CNN classifies the images directly without using any other supporting approaches such as data augmentation, dropout, weight-decay, etc., then the accuracy and loss plots show stall. This problem is termed as overfitting. In this scenario, Training accuracy arose linearly and reaches 100%, and validation accuracy stalls at 70-75%. Classification accuracy of 35.72% was obtained for bean crop disease detection. Loss plots (for both training and testing) also stall after a few iterations. Training and validation curves for both accuracy and loss are shown in figure 4.

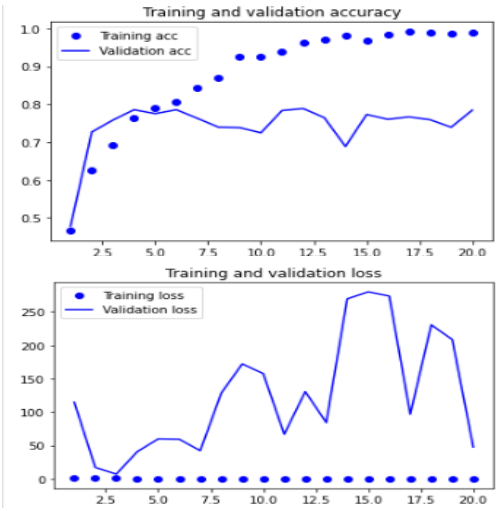


Fig 4. Shows the Overfitted accuracy and loss curves for training and validation.

4.6 Effects of data-augmentation on the classification

Whenever there are only a few of the training instances are available, in such cases, CNN is restricted to learning and is incapable of training the network that could generalize on new samples. This is the overfitting issue in training [30]. To avoid this problem, there is a need to introduce the network with every potential facet of the data distribution.

Data augmentation generates more training instances from the available training set, through augmenting the instances using a range of random transformations that produce similar-looking images to the available data. So, the network will not be able to see the same image twice at training time. It benefits the network by exposing it to many more features of data and performs better generalization. Data augmentation [31] may include the following parameters: rotation, width-shift, height-shift, shear, zoom, horizontal flip, etc. Some of the augmented training process; a random set of neurons is selected. Both

of the techniques (data augmentation and dropout) improve the accuracy of approximately up to 84%.

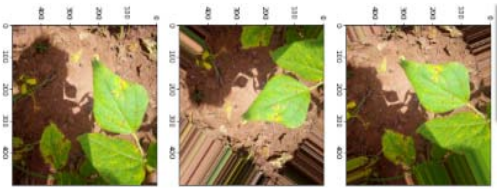


Fig 5. Shows some augmented instances.

A classification accuracy of 56.08% was obtained for bean crop disease detection. Accuracy and loss results with the data augmentation and dropout are shown in figure 6.

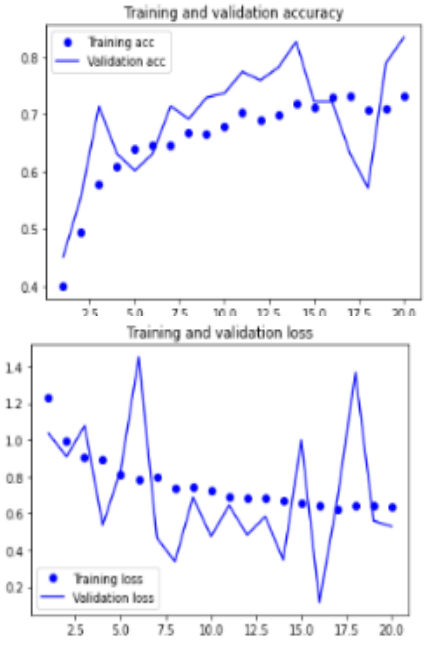


Fig 6. Training and validation accuracy using data augmentation and dropout.

5. Training using a pre-trained CNN

For small image data samples, the use of the pre-trained networks is a general and very effective choice [30]. A pre-trained network can be defined as a saved model which has been previously trained over a bulk of available data such as ImageNet. The spatial hierarchy of their learned features is very generic; thus, it might be useful for a diverse range of computer vision problems with totally different classes than those of the actual ones. In this study, a pre-trained VGG16 model is used to classify the healthy and diseased bean crop leaves. However, the original VGG16 model was trained over the ImageNet dataset (includes 1000 classes of 1.4 million images). Pre-trained networks can be used in two ways: feature extraction and fine-tuning.

5.1 VGG16 Model

Simonyan et al. [32] developed a DL architecture with 16 convolutional layers and 3x3 convolution filters were used to increase the depth of the network. It showed significant improvement in the accuracy of large- scale image recognition. The weight configuration for this model is publicly available. VGG comprised of 138 million parameters that make it very challenging to handle. Figure 7 represents VGG16 deep architecture. Some technical insights of VGG16 are depicted below [32]:

- The first convolutional layer accepts 224 * 224 sized RGB images and then these images are passed through the stacked convolutional layers. On the next step, activation maps (filters) are deployed with a very minor receptive field of 3*3 pixels. These receptive fields help to capture the notion of up, down, left, right, and center in the image. Each hidden layer is followed by ReLU functionality.

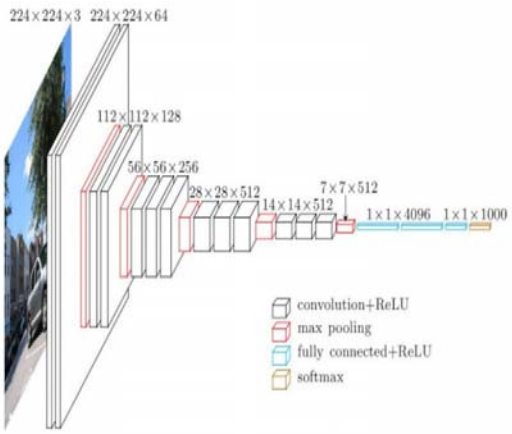


Fig 7. VGG16 models architecture [32].

- Spatial padding of 1-pixel is preserved for the convolution operations, after every 3*3 convolution.
- Spatial pooling with a 2*2 pixel window is followed with the max pooling operations, with a stride of 2.
- There are three fully connected layers (two with 4096 channels and one of 1000 channels) present

Manuscript received October 5, 2020
Manuscript revised October 20, 2020
https://doi.org/10.22937/IJCSNS.2020.20.10.26

that is used after the convolution and max pooling operations.

- Finally, a classification layer soft-max is used to classify the desired objects in the dataset.

5.2 Feature extraction

The pre-trained network uses the representations that are learned previously while training the network on a large dataset [30]. It extracts the required features from the new data, using the previously trained network. For feature extraction, some of the layers of the convolutional base are kept in frozen form and other ones need to be changed according to the problem requirement. Figure 8 represents the classification of images with a frozen convolutional base taken from the pre-trained network. Pre-trained CNNs used for image classification tasks consist of two parts:

- **Convolutional base:**

It is a series of pooling and convolution layers that are used to extract features from a previously trained CNN, deploying over new data, and new classifier. In a situation, where a new dataset is completely different from the pre-trained one, then it is better to use only the initial few layers from the convolutional base to perform feature extraction, instead of the entire convolutional base.

- **Densely connected classifier:**

A new classifier is swapped with that one used in the pre-trained model. This classifier is placed on the top of the output layer to perform prediction or classification. It contains the value of the classes that need to be used for the purpose.

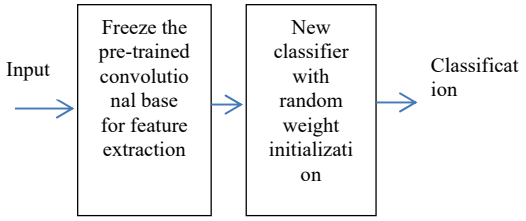


Fig 8.Classification with frozen convolutional base.

Furthermore, there are two possible ways to extract features from the convolutional base, namely, feature extraction without data augmentation and feature extraction with data augmentation.

5.2.1 Feature extraction without data augmentation

This process of feature extraction allows the user to run the convolutional base along with a densely connected classifier directly on the dataset. It needs to run the convolutional base only once for each input image. This technique does not allow for data augmentation. Features are extracted by defining the appropriate feature extraction function which then extracts the required features from the convolutional base. These features are in the form of pixels and edges. In this process, training is quite fast. This method gave a validation accuracy of 92.43% which is better than the training from scratch. Classification accuracy of 65.06% was obtained for bean crop disease detection. Experimental analysis shows that it gives overfitted results without using data augmentation on a small data sample. Overfitted plots for accuracy and loss are shown in figure 9.

5.2.2 Feature extraction with data augmentation

In the second approach, features are extracted using the data augmentation techniques which give far better results than the previous one. This technique allows for data augmentation during training. The convolutional base is extended with some layers and running the model end to end on the data inputs. This method is slow and expensive, as it requires GPU support during implementation. This method gave a validation accuracy of 92.43% which is much better than the feature extraction without data augmentation. Accuracy and loss plots for training and validation are shown in figure 10. Classification accuracy of 78.26% was obtained for bean crop disease detection.

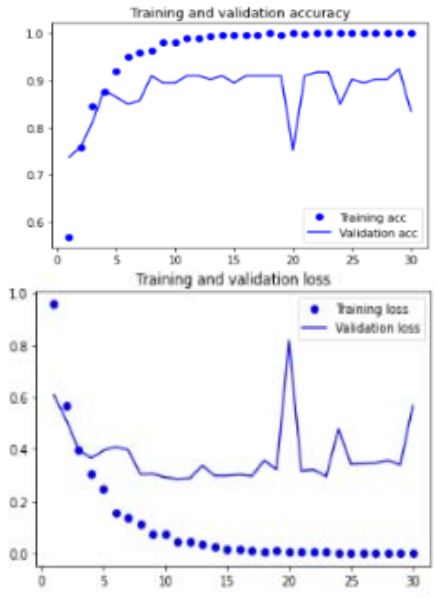


Fig 9. Training and validation curves for accuracy and loss metrics are shown for simple feature extraction

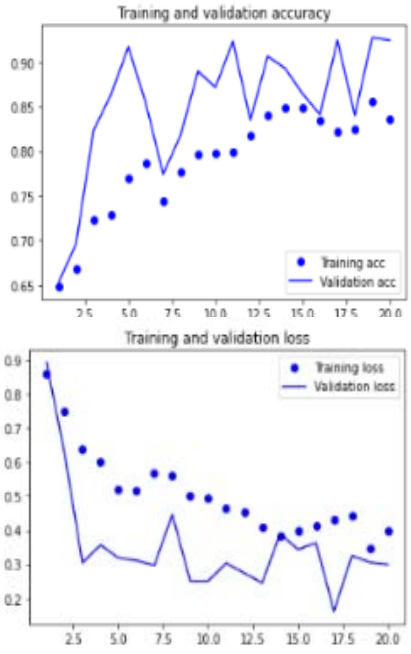


Fig 10. Training and validation curves for accuracy and loss metrics are shown for feature extraction with data augmentation.

5.3 Fine-tuning

Fine-tuning is another complementary technique to reuse the model, besides the feature extraction. This approach comprises unfreezing some of the top layers from the convolutional base of the model that is utilized to extract features [30]. Subsequently, these top layers are trained, jointly with the fully connected classifier (newly added part). It is termed as fine-tuning, as it marginally regulates more abstract representations of the pre-trained model to form it more problem relevant. During implementation, 4 convolutional blocks were frozen and the last block was fine-tuned for the VGG16 model. RMSProp optimizer was used as a fine-tuning parameter with a learning rate of 1e-5. Experimental results show that the validation accuracy of 97.06% was achieved. Accuracy and loss plots for training and validation are shown in figure 11. Classification accuracy of 96.06% was obtained for bean crop disease detection.

6. Comparison of results

In this paper, a comparison is drawn for, training a CNN from scratch and using pre-trained models for a small labeled dataset on healthy and diseased leaves of the bean crop. Moreover, training a CNN from scratch is further classified into two parts: using the model with data augmentation and without data augmentation.

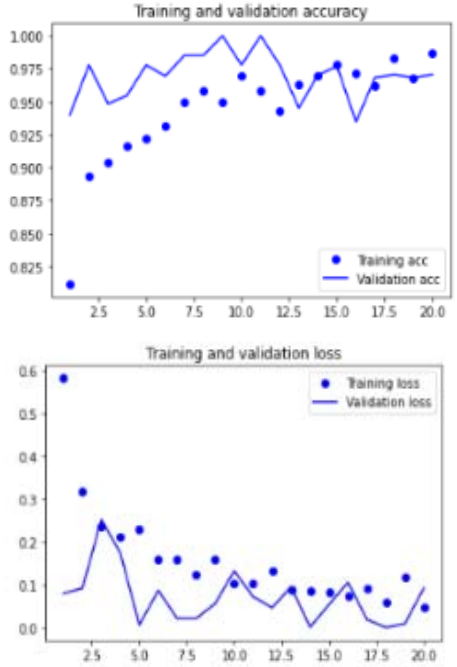


Fig 11. Training and validation curves for accuracy and loss metrics are shown for fine-tuning of the pre-trained network.

When CNN was trained from scratch using a simple stack of four convolutional layers, four max-pooling layers, and two fully connected layers, then a classification

accuracy of 35.72% was achieved. The outcome of this experiment showed that CNN was facing an overfitting issue. In order to mitigate this problem, data augmentation and drop out techniques were added to enhance the classification accuracy. It improved the classification results by 56.08%. Similarly, pre-trained models are also categorized as training using feature-extraction and fine-tuning of models. When a pre-trained model VGG16 was deployed by extracting the features from its convolutional base, then the classification accuracy reached up to 65.06%. Pre-trained models also face the overfitting problem over small datasets. To avoid the overfitting issue, data augmentation and drop out techniques were added along with the feature extraction process. An improvement of 78.26% was achieved in the classification accuracy for bean crop diseases. Furthermore, the fine-tuning of the pre-trained model was also performed by the selection of appropriate hyper-parameters. It gave far better results from the previous techniques with a classification accuracy of 96.06%. Table 3 presents the comparison of all the evaluated results achieved for the classification of the bean crop leaf diseases.

Table 3: Comparison of experimental evaluation.

Performance Metrics	***Training of CNN from scratch***		***Training using Pre-trained Models***		
	Without data augme-ntation	***Using dropout and data augment-tation***	***Feature extraction***		***Fine-tuning***
			Simple	***Using data augment-ation***	
Training Accuracy	0.989	0.7312	1.000	0.8367	0.9867
Training Loss	0.047	0.634	0.0014	0.3991	0.0483
Validation Accuracy	0.743	0.8246	0.9248	0.9243	0.9706
Validation Loss	0.4767	0.529	0.3398	0.2992	0.0926
Classification Accuracy	0.3572	0.5608	0.6506	0.7826	0.9606

7. Conclusion

In this paper, the focus is to clear the concept of training a CNN from scratch and using pre-trained models in the context of the agricultural domain. An experimental study is carried on the bean crop leaf images (infected/healthy). Experimentations have demonstrated the usefulness of fine-tuned models over small samples of training data, using training from scratch and pre-trained fine-tuned model, with a significant accuracy jump from 0.70 (approximately) to 0.9706. Using the layer-wise fine-tuning approach, the effective depth of fine-tuning can be determined in order to find the best classification results. Our implementations further ensure that how potential are the fine-tuned models as compared to the training from scratch for a small amount of labeled data.

Acknowledgment

Authors are thankful to the Department of Science & Technology, Government of India, Delhi for funding a project on "Application of IoT in Agriculture Sector" through ICPS division. This work is a part of the project work.

References

[1] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193–202, 1980.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv Prepr. arXiv1409.1556*, 2014.

[7] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, 2014, pp. 818–833.

[8] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 806–813.

[9] H. Azizpour, A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 36–45.

[10] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Comput. Electron. Agric.*, vol. 145, pp. 311–318, 2018.

[11] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," *Sensors*, vol. 17, no. 9, p. 2022, 2017.

[12] I. Sa *et al.*, "weednet: Dense semantic weed classification using multispectral images and mav for smart farming," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 588–595, 2017.

[13] N.-N. Wang, Y.-C. Yang, D.-W. Sun, H. Pu, and Z. Zhu, "Shelf-life prediction of 'Gros Michel'bananas with different browning levels using hyperspectral reflectance imaging," *Food Anal. Methods*, vol. 8, no. 5, pp. 1173–1184, 2015.

[14] Y. Kawasaki, H. Uga, S. Kagiwada, and H. Iyatomi, "Basic study of automated diagnosis of viral plant diseases using convolutional neural networks," in *International Symposium on Visual Computing*, 2015, pp. 638–645.

[15] Q. Liang, S. Xiang, Y. Hu, G. Coppola, D. Zhang, and W. Sun, "PD2SE-Net: Computer-assisted plant disease diagnosis and severity estimation network," *Comput. Electron. Agric.*, vol. 157, pp. 518–529, 2019.

[16] A.-K. Mahlein, E. Alisaac, A. Al Masri, J. Behmann, H.-W. Dehne, and E.-C. Oerke, "Comparison and combination of thermal, fluorescence, and hyperspectral imaging for monitoring Fusarium head blight of wheat on spikelet scale," *Sensors*, vol. 19, no. 10, p. 2281, 2019.

[17] C. Veys *et al.*, "Multispectral imaging for presymptomatic analysis of light leaf spot in oilseed rape," *Plant Methods*, vol. 15, no. 1, p. 4, 2019.

[18] C. Xie, C. Yang, and Y. He, "Hyperspectral imaging for classification of healthy and gray mold diseased tomato leaves with different infection severities," *Comput. Electron. Agric.*, vol. 135, pp. 154–162, 2017.

[19] T. Chen, J. Zhang, Y. Chen, S. Wan, and L. Zhang, "Detection of peanut leaf spots disease using canopy hyperspectral reflectance," *Comput. Electron. Agric.*, vol. 156, pp. 677–683, 2019.

[20] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Front. Plant Sci.*, vol. 7, p. 1419, 2016.

[21] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," *Comput. Intell. Neurosci.*, vol. 2016, 2016.

[22] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, 2017.

[23] M. Brahimi, K. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: classification and symptoms visualization," *Appl. Artif. Intell.*, vol. 31, no. 4, pp. 299–315, 2017.

[24] J. G. Ha *et al.*, "Deep convolutional neural network for classifying Fusarium wilt of radish from unmanned aerial vehicles," *J. Appl. Remote Sens.*, vol. 11, no. 4, p. 42621, 2017.

[25] S. Ghosal, D. Blystone, A. K. Singh, B. Ganapathysubramanian, A. Singh, and S. Sarkar, "An explainable deep machine vision framework for plant stress phenotyping," *Proc. Natl. Acad. Sci.*, vol. 115, no. 18, pp. 4613–4618, 2018.

[26] M. Brahimi, S. Mahmoudi, K. Boukhalfa, and A. Moussaoui, "Deep interpretable architecture for plant diseases classification," in *2019 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, 2019, pp. 111–116.

[27] K. Nagasubramanian, S. Jones, A. K. Singh, A. Singh, B. Ganapathysubramanian, and S. Sarkar, "Explaining hyperspectral imaging based plant disease identification: 3D CNN and saliency maps," *arXiv Prepr. arXiv1804.08831*, 2018.

[28] M. A. Khan *et al.*, "CCDF: Automatic system for segmentation and recognition of fruit crops diseases based on correlation coefficient and deep CNN features," *Comput. Electron. Agric.*, vol. 155, pp. 220–236, 2018.

[29] A. C. Cruz, A. Luvisi, L. De Bellis, and Y. Ampatzidis, "Vision-based plant disease detection system using transfer and deep learning," in *2017 asabe annual international meeting*, 2017, p. 1.

[30] F. Chollet, *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG, 2018.

[31] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Comput. Electron. Agric.*, vol. 147, pp. 70–90, 2018.

[32] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.