International Journal of Computing Sciences Research (ISSN print: 2546-0552; ISSN online: 2546-115X)
Vol. 7, pp. 2415-2430
doi: 10.25147/ijcsr.2017.001.1.167
https://stepacademic.net

Short Paper

A Comparative Study of Different Architectural Models of CNN for Plant Leaf Disease Detection

Vivek Kumar
Department of Computer Science, Central University of South Bihar, Gaya, India

Satrughan Kumar Singh
Department of Computer Science, Central University of South Bihar, Gaya, India

Jainath Yadav
Department of Computer Science, Central University of South Bihar, Gaya, India
jainath@cub.ac.in
(corresponding author)

Muniyan Sundararajan
Department of Mathematics & Computer Science, Mizoram University, Aizawl, India

Date received: May 16, 2023
Date received in revised form: August 5, 2023; August 17, 2023
Date accepted: August 21, 2023

Recommended citation:

Kumar, V., Singh, S. K., Yadav, J., & Sundararajan, M. (2023). A comparative study of different architectural models of CNN for plant leaf disease detection. *International Journal of Computing Sciences Research*, 7, 2415-2430. https://doi.org/10.25147/ijcsr.2017.001.1.167

Abstract

Purpose – From the last few decades, pattern recognition has become an emerging task of machine learning and image processing with robust integration. This paper provides a comparative study of different plant leaf disease detection techniques of the CNN model in the domain of image processing.

Method – In this paper, we compared three architectural models of CNN namely, AlexNet, VGG16Net, and ResNet for plant disease detection. AlexNet has five convolution layers followed by three fully connected layers. VGG uses a small receptive field followed by a ReLu unit and it has three fully connected layers. ResNet works on skip connection and it passes input data through the weight layer processing by model function.

Results – ResNet provides an effective result with 100 epoch iterations of dataset training and validation. ResNet achieved higher training and validation accuracy than AlexNet and VGG16Net models. ResNet has also achieved less training and validation loss. Finally, the experimental results have shown that ResNet is better than AlexNet and VGG16Net models.

Conclusion – In this study, we concluded that the residual network i.e. ResNet is showing better results than AlexNet and VGG16Net. Finally, the comparative experimental results have shown that ResNet provides effective output with 100 epochs.

Recommendations – The recognition rate of ResNet needs to be tested by increasing the number of epoch's iterations and adding more and new leaf data for training and testing datasets for future work. In future research, we recommended the development of an Android-based mobile App for plant leaf disease detection useful for farmers.

Research Implications – Farmers can easily operate this system on their smartphones with a few days technical training given by expert professionals to detect plant leaf disease.

Keywords – Plant disease detection, image processing, neural network, image segmentation, feature extraction & classification

INTRODUCTION

Machine learning is currently playing a vital role in the next generation of the computer world (Singh et al., 2021). Nowadays, machine learning facilitates an automatic intelligent system for pattern recognition. Automatic pattern recognition has become an important issue of image processing and machine learning (Singh et al., 2021). In today's world (He et al., 2015), the agricultural land mass is more than just a source of feeding. Agricultural productivity has become the foundation of the Indian economy. Because of this reason, detection of disease in the plants has become a major role. Plant disease detection incorporates rapidly increasing complexity for the observation of disease symptoms. As India is eminent for agriculture that means most of the population is engaged in the agriculture sector. According to data, the agriculture sector contributes 15.4% of GDP growth worth agricultural activity of $375.61 billion, India is the 2nd largest producer of agricultural products in the entire world. So it becomes most significant to take care of our crops and plants for our food security soon. Crop production can be enhanced to a great extent by new technology.

In machine learning, the computer intelligence expert models are trained by known datasets of the domain context (Singh et al., 2023). An automated computer system designed to help identify plant disease by the plant's visual appearance and symptoms could be of great help. Image processing has become an emerging technology with the integration of machine learning. Deep learning (a subset of machine learning) incorporates a good integration of the AI model with image processing for feature detection through dataset training and testing for finding optimal and accurate results and potentials. Object detection and image segmentation are used in image processing and computer vision to identify well-defined patterns (Singh et al., 2021). Convolutional Neural Network (CNN) is considered to be the leading method for object detection. In this paper, we consider detectors namely AlexNet, VGG16Net, and Residual Network (ResNet50). Each architecture should be able to be merged with any number of feature extractors depending on the application we need. We have compared the results of all three architectures on plant village datasets.

The main contribution of the paper is to develop different CNN models for plant leaf disease detection and analyze their detection performance. The leaf disease detection software will be used to detect plant disease in the early stage. This automated soft computing system of plant leaf disease detection facilitates an advantage in the agricultural domains for careful monitoring of the plants and crop growth in the early stage and also reducing laborious efforts. It will help to increase the production of the crop and minimize the capital loss due to different plant diseases. The software will also help farmers to connect automatically to the world of the Internet to detect any new disease in the plant. It will provide any new updates in agricultural fields automatically. After detecting the disease in the plant, it can also recommend pesticides for the respective disease. Therefore, the disease detection system will be very useful to farmers.

LITERATURE REVIEW

Sapkal et al. (2018) proposed an approach for the diagnosis of affected leaf disease through segmentation and feature extraction and BPNN (back propagation neural network). Cortes et al. (2017) discussed in a paper the development of a multi-label categorical generative network and implemented a loss function that could create a discriminator that acts as a classifier for different plant diseases. Al-Hairy et al. (2011) studied building a model by using K-Means clustering, color co-occurrence methodology for texture analysis, and a neural network for the detection of leaf disease. Kranth et al. (2018) proposed comparing results using various machine learning algorithms decision trees, Naive Bayes, ANN, k-means, and RF for leaf disease prediction. The proposed work applies the concept of ensemble learning, implemented through various machine learning algorithms. The work is based on morphological features of the plant leaf. The research work is carried out on inbuilt datasets. Mangla et al. (2019) described a method for detecting and diagnosing paddy leaf disease using image pre-processing and segmentation techniques with SVM classifiers, but SVM is less accurate than CNN. Wallelign et al. (2018), reviewed in detail various solutions put forward to detect soybean plant disease identification using CNN.

In this paper, the feasibility of CNN to classify plant disease from leaf images taken under an uncontrolled environment is used and models are designed based on LeNet (LeCun et al., 1998) architecture. Since CNN requires a large amount of data, data augmentation is used to increase the training data. Although the data sample used in this work is unbalanced. Priyadharshini et al. (2019) used a deep learning methodology, a convolution neural network to get features. The pooling layer is used for sampling and dimensionality reduction. Akila et al. (2018), proposed a paper titled "Detection and classification of plant disease by using deep learning algorithm", to find and develop the more suitable deep learning methodologies. The proposed system can effectively identify different types of disease from the plant's area, but computational complexities became very high, thus it's time-consuming. The 03 main families of detectors namely Faster Region-based Fully Convolutional Network (Faster R-CNN), Region-based Fully Convolutional Network (R-FCN), and Single shot Multibox Detector (SSD) were used and they are discussed in the following subsections:

Single shot Multibox Detector (SSD)

Adeel et al. (2019) propose an automatic system for the segmentation and recognition of grape leaf diseases based on the MEAN-SSD model. They achieved an average recognition accuracy rate of 92%. It splits the convolution process to reduce the size so can fit well onto mobile devices. According to Sun (2021), a new apple leaf disease detection model is built by using the MEAN block and Apple-Inception module. This MEAN-SSD approach achieved a recognition accuracy of 83.12%. This approach can detect five diseases in apple plants.

Region-based Fully Convolutional Network (R-FCN)

Deep learning models are very useful for detecting disease in tomato plants (Rehana et al., 2023). The proposed Region-based CNN model detects tomato leaf diseases with an average accuracy of 96.31%.

Faster Region-based Fully Convolutional Network (Faster R-CNN)

In this paper (Priya et al., 2021), the Faster R-CNN model has been explored with RPN for recognizing disease images from cotton plants. This approach provides higher recognition accuracy and it is a comparatively faster approach. Shaikh et al. (2022) developed a Faster-RCNN model for rice leaf disease detection. They achieved recognition accuracy of 98.85%, 98.09%, 99.17%, and 99.25% for brown spot, rice blast, hispa, and healthy rice leaf, respectively.

METHODOLOGY

Dataset

The Dataset used here is the "plant village dataset" by Mohanty et al. (2016), having 54,306 images of defective and healthy plant leaves collected under controlled conditions, having spread over 38 class labels assigned to them. Figure 1 shows the sample of defective plant leaves from the Plant Village dataset. Figure 1 shows the sample of defective plant leaves from the Plant Village dataset.

Figure 1. Samples of defected plant leaf [source: PlantVillage dataset (Mohanty et al., 2016)]

Neural Network (NN)

In the neural network system, the data pre-processing phase is the initial phase in any machine learning project, leading to better results from the applied model [Singh et al., 2022). Neural networks are matured to work best on large datasets (Singh et al., 2022). Images are segmented into leaf and background portions with different sizes and color patterns. Finally, the convolutional neural network is applied as a classifier on extracted parameters. The mathematical calculation for error (E) (Equation 1) is expressed as:

$$Error: E = y - f(x), \qquad update\ weights \qquad \textit{Equation 1}$$

Where y and f(x) are the correct output and output function of the network.

AlexNet is developed by Krizhevsky et al. (2012) at ImageNet Large Scale Visual Recoginition Challenge (ILSVRC). AlexNet has 5 convolutional layers which are followed by 3 fully connected layers. Figure 2 shows the block diagram of AlexNet architecture.

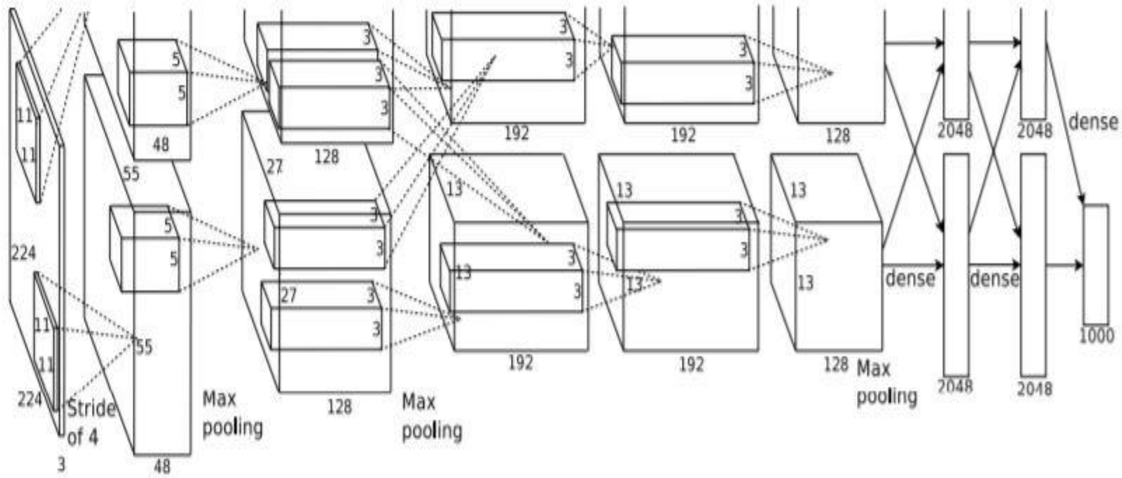


Figure 2. Block diagram of AlexNet architecture

AlexNet assumes the input image is of size 224x224x3, where 3 represents the RGB channel of the color image. The 1st and 2nd convolutional layers have been considered with sizes of 96 and 256 kernels, respectively. The ReLu activation function was used. And border padding is kept "the same" as the input image. After the 1st convolutional layer, the normalization is done and the result then goes to max pooling. The 3rd, 4th, and 5th layers are used and they don't interrupt the pooling or normalization layer. Each of the convents used in architecture extracts more and more complicated features from the image. The 02 fully connected dense layers are used for feature classification. The result of the final convolution layer is the most complex features extracted from AlexNet.

Simonyan et al. (2015) proposed VGG16Net for deep CNN. VGG16Net takes the input of 224x224 pixels of RGB image. VGG uses a small receptive field (3x3 with a stride of 1) followed by a ReLu unit. VGG has 03 fully connected layers. The 1st and 2nd layers have 4096 channels each and the 3rd layer has 1000 channels, 1 for each class. All VGG's hidden layers use ReLu. VGG has many variants, among which VGG16 is famous as its name derived from its architecture using 16 layers in total among 13 convolutional layers, 2 fully connected layers, and 1 output layer. The block diagram for processing of dataset using VGG16Net architecture is shown in Figure 3.

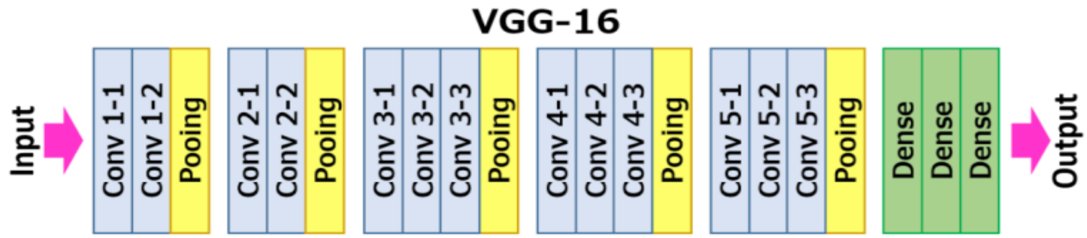


Figure 3. Block diagram for processing of dataset using VGG16Net architecture

ResNet (He et al., 2015) is known as a residual network. ResNet works on skip connection. As it is known, the deep NN with backpropagation suffers problems generated by smaller vanishing gradients without any adjustments. To overcome this problem, Microsoft introduced a deep residual learning framework. Figure 5 shows the workflow diagram of ResNet for passing input data through the weight layer processing by model function. Figure 4 shows skip connections annotated as identity. The connections make the network learn the mapping function to pass the input parameters through the block, but it doesn't pass through weight layers. This process resolves the vanishing gradient problem and allows efficient training through skip connection layers.

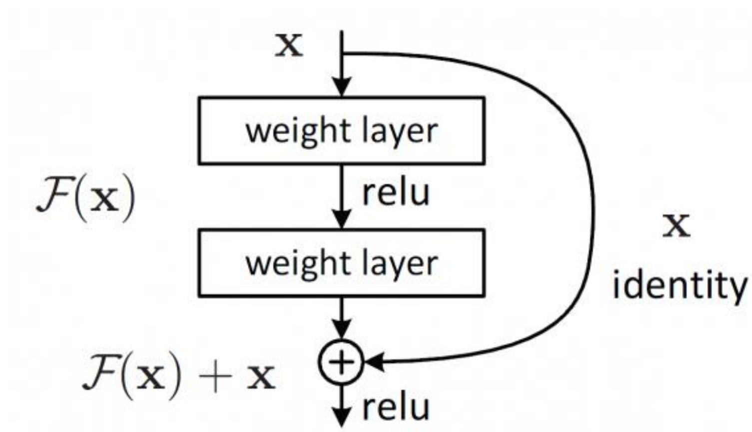


Figure 4. ResNet for passing input data through the weight layer processing by model function

EXPERIMENTAL RESULTS

The experiment setup requires a computer system with an Intel Core i3 processor or higher and a minimum of 8 GB of RAM (Random Access Memory). The experiment was carried out on Microsoft Windows 8 Operating System and Python was pre-installed along with different essential Libraries. The leaf detection task has been performed on the plant village dataset (Mohanty et al., 2016). The total number of defective and healthy images in the dataset is 54,306 and the dataset has 38 class labels. These images have been collected under different conditions. For comparison purposes, we have run the three CNN models for 50 and 100 epoch iterations, respectively.

The result presented in this section is related to training with the whole dataset split into 80:20 ratios in the training and testing dataset. It is known that convolutional neural networks can learn features when trained on large datasets, the results achieved with all three neural networks i.e. AlexNet, VGG16Net, and ResNet are different in the same scenarios (see Figures 5 to 7).

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 54, 54, 96)        34944
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 27, 27, 96)        0
_________________________________________________________________
batch_normalization_9 (Batch (None, 27, 27, 96)        384
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 17, 17, 256)       2973952
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 8, 8, 256)         0
_________________________________________________________________
batch_normalization_10 (Batc (None, 8, 8, 256)         1024
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 6, 6, 384)         885120
_________________________________________________________________
batch_normalization_11 (Batc (None, 6, 6, 384)         1536
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 2, 2, 256)         884992
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 1, 1, 256)         0
_________________________________________________________________
batch_normalization_13 (Batc (None, 1, 1, 256)         1024
_________________________________________________________________
flatten_2 (Flatten)          (None, 256)               0
_________________________________________________________________
dense_5 (Dense)              (None, 1024)              263168
_________________________________________________________________
dropout_4 (Dropout)          (None, 1024)              0
_________________________________________________________________
batch_normalization_14 (Batc (None, 1024)              4096
_________________________________________________________________
dense_6 (Dense)              (None, 4096)              4198400
_________________________________________________________________
dropout_5 (Dropout)          (None, 4096)              0
_________________________________________________________________
batch_normalization_15 (Batc (None, 4096)              16384
_________________________________________________________________
dense_7 (Dense)              (None, 1000)              4097000
_________________________________________________________________
dropout_6 (Dropout)          (None, 1000)              0
_________________________________________________________________
batch_normalization_16 (Batc (None, 1000)              4000
_________________________________________________________________
dense_8 (Dense)              (None, 15)                15015
=================================================================
Total params: 14,710,063
Trainable params: 14,695,071
Non-trainable params: 14,992
_________________________________________________________________
Found 4447 images belonging to 15 classes.
Found 940 images belonging to 15 classes.
```

Figure 5. AlexNet model dataset processing using Python-TensorFlow

```
Using TensorFlow backend.
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 224, 224, 32)      896
_________________________________________________________________
dropout_1 (Dropout)          (None, 224, 224, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 224, 224, 32)      9248
_________________________________________________________________
dropout_2 (Dropout)          (None, 224, 224, 32)      0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 112, 112, 32)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 112, 112, 64)      18496
_________________________________________________________________
dropout_3 (Dropout)          (None, 112, 112, 64)      0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 112, 112, 64)      36928
_________________________________________________________________
dropout_4 (Dropout)          (None, 112, 112, 64)      0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 56, 56, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 56, 56, 128)       73856
_________________________________________________________________
dropout_5 (Dropout)          (None, 56, 56, 128)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 56, 56, 128)       147584
_________________________________________________________________
dropout_6 (Dropout)          (None, 56, 56, 128)       0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 28, 28, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 100352)            0
_________________________________________________________________
dense_1 (Dense)              (None, 15)                1505295
=================================================================
Total params: 1,792,303
Trainable params: 1,792,303
Non-trainable params: 0
_________________________________________________________________
Found 4447 images belonging to 15 classes.
Found 940 images belonging to 15 classes.
```

Figure 6. VGG16Net model dataset processing using Python-TensorFlow

Model: "sequential_3"

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 256, 256]	1,792
BatchNorm2d-2	[-1, 64, 256, 256]	128
ReLU-3	[-1, 64, 256, 256]	0
Conv2d-4	[-1, 128, 256, 256]	73,856
BatchNorm2d-5	[-1, 128, 256, 256]	256
ReLU-6	[-1, 128, 256, 256]	0
MaxPool2d-7	[-1, 128, 64, 64]	0
Conv2d-8	[-1, 128, 64, 64]	147,584
BatchNorm2d-9	[-1, 128, 64, 64]	256
ReLU-10	[-1, 128, 64, 64]	0
Conv2d-11	[-1, 128, 64, 64]	147,584
BatchNorm2d-12	[-1, 128, 64, 64]	256
ReLU-13	[-1, 128, 64, 64]	0
Conv2d-14	[-1, 256, 64, 64]	295,168
BatchNorm2d-15	[-1, 256, 64, 64]	512
ReLU-16	[-1, 256, 64, 64]	0
MaxPool2d-17	[-1, 256, 16, 16]	0
Conv2d-18	[-1, 512, 16, 16]	1,180,160
BatchNorm2d-19	[-1, 512, 16, 16]	1,024
ReLU-20	[-1, 512, 16, 16]	0
MaxPool2d-21	[-1, 512, 4, 4]	0
Conv2d-22	[-1, 512, 4, 4]	2,359,808
BatchNorm2d-23	[-1, 512, 4, 4]	1,024
ReLU-24	[-1, 512, 4, 4]	0
Conv2d-25	[-1, 512, 4, 4]	2,359,808
BatchNorm2d-26	[-1, 512, 4, 4]	1,024
ReLU-27	[-1, 512, 4, 4]	0
MaxPool2d-28	[-1, 512, 1, 1]	0
Flatten-29	[-1, 512]	0
Linear-30	[-1, 38]	19,494

Total params: 6,589,734
Trainable params: 6,589,734
Non-trainable params: 0

Found 4447 images belonging to 15 classes.
Found 940 images belonging to 15 classes.

Figure 7. ResNet model dataset processing using Python-TensorFlow

We applied the three models in two scenarios. The first one is 50-epochs iterations and the second one is 100-epochs iterations, while comparing these three network models on four parameters tables, i.e. training accuracy, validation accuracy, training loss, and validation loss. The results of these models are depicted in Tables 1 to 4.

Table 1. Training accuracy of CNN architecture

CNN architecture	Partition strategy	50 epochs	100 epochs
AlexNet	Training set = 80%	0.87	0.92
VGG16Net	&	0.09	0.92
ResNet	Testing set = 20%	0.90	0.95

Table 2. Validation of the accuracy of CNN architecture

CNN architecture	Partition strategy	50 epochs	100 epochs
AlexNet	Training set = 80%	0.78	0.94
VGG16Net	&	0.06	0.91
ResNet	Testing set = 20%	0.07	0.96

Table 3. Training loss of CNN architecture

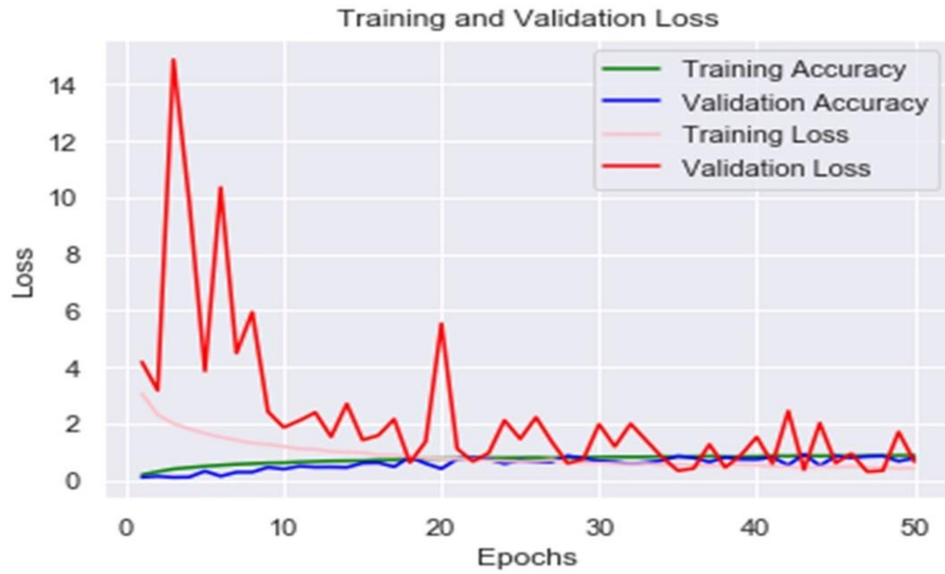
CNN architecture	Partition strategy	50 epochs	100 epochs
AlexNet	Training set = 80%	0.41	0.24
VGG16Net	&	2.68	2.02
ResNet	Testing set = 20%	0.37	0.16

Table 4. Validation loss of CNN architecture

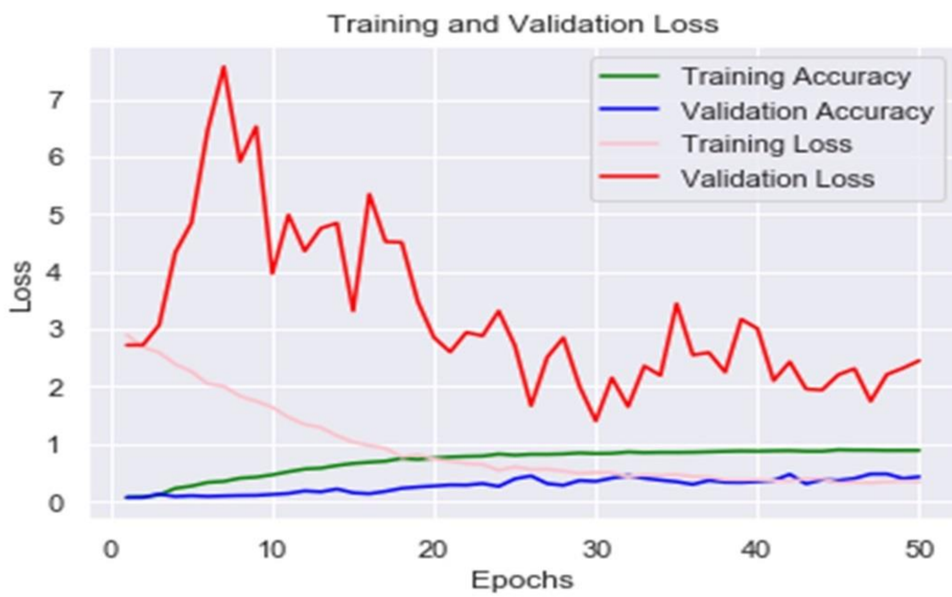
CNN architecture	Partition strategy	50 epochs	100 epochs
AlexNet	Training set = 80%	0.64	0.15
VGG16Net	&	2.68	0.12
ResNet	Testing set = 20%	3.10	0.10

DISCUSSIONS

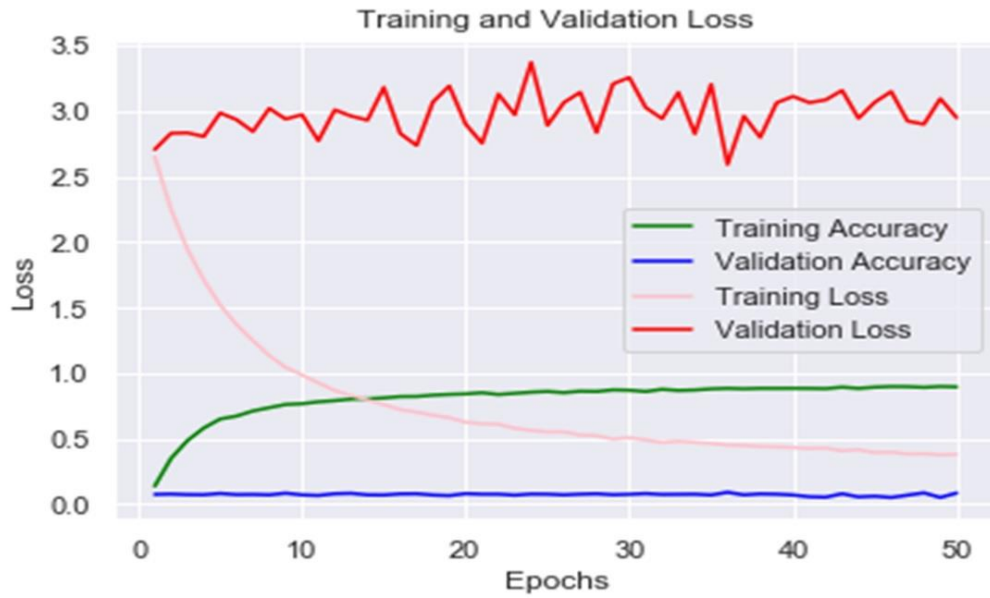
As we see from Table 1, in training accuracy, ResNet has 90% accuracy with 50 epochs and 95% accuracy with 100 epochs, respectively. In contrast, VGG16Net achieved 9% and 92% accuracy with 50 epochs and 100 epochs, respectively. From Table 2, in validation accuracy, we observe that AlexNet has 78% accuracy with 50 epochs and 94% accuracy with 100 epochs, respectively. However, when comparing ResNet with 100 epochs to AlexNet and VGG16 Net, we see that ResNet achieved the highest validation accuracy with 100 epochs. We observe that ResNet has less training loss with 50 epochs and 100 epochs as compared to AlexNet and VGG16Net (see Table 3). From Table 4, AlexNet has achieved less validation loss as compared to VGG16Net and ResNet in 50 epochs. But when we compared ResNet with 100 epochs to AlexNet and VGG16Net, the result shows ResNet has less validation loss than VGG16Net and AlexNet.



AlexNet CNN



VGG16 Net CNN



ResNet CNN

Figure 8. Graphical representation of plotting loss vs. epochs curves of three CNN architectures

Plotting loss vs. epochs curves under all three convolutional neural network architectures i.e. AlexNet, VGG16Net, and ResNet are shown in Figure 8. These plots show the ResNet training accuracy is highest and gradually increasing in 50 epochs, while the validation accuracy slightly remains constant. But, as compared to 100 epoch iterations, ResNet achieved the highest validation accuracy with 100 epochs. Whereas AlexNet training and validation accuracy is constant, validation loss drastically fluctuates from 4 to within under 2 extended under 50 epochs. VGGNet training and validation loss is constantly and slowly decreasing. Thus, the comparative result shows ResNet achieved higher accuracy and efficient results that obtained the best recognition rate than AlexNet and VGG16Net in 100 epoch iterations.

CONCLUSIONS AND RECOMMENDATIONS

In the research we concluded, we observed that the residual network i.e. ResNet is showing promising results in image feature identification and training and validation of image datasets in convolutional neural networks among its peers. Finally, the comparative experimental results have shown that ResNet provides effective output with 100 epochs and is better than AlexNet and VGG16Net models. The recognition rate of ResNet needs to be tested by increasing the number of epoch iterations and adding more and new leaf data for training and testing datasets for future work. In future research, we recommend the development of an Android-based mobile App for plant and crop disease detection in the agricultural sector. The farmers can easily capture leaf images through their mobile phones. The plant leaf disease detection system will process the captured images for disease diagnosis. It will detect and prevent the disease from happening in the early stages. This application software can be economical and it will also save their time for farmers.

IMPLICATIONS

This study provides direct implication for the agricultural sector through the given insights into the automatic detection system of plant leaf disease and the proposed model of the system that determine leaf disease and classification patterns that help identify appropriate information about healthy and unhealthy plants and provides mitigative preventive measures of the plant's growth in the early stage. Farmers can easily operate this system on their smartphones with a few days of technical workshops given by expert professionals to detect plant leaf disease.

ACKNOWLEDGEMENT

We would like to thank our institutions.

FUNDING

This study is partially funded by the affiliations of the authors.

DECLARATIONS

Conflict of Interest

The authors declare that they have no conflict of interest.

Informed Consent

Not applicable.

Ethics Approval

This paper does not contain any studies with human participants or animals performed by any of the authors.

REFERENCES

Adeel, A., Khan, M. A., Sharif, M., Azam, F., Shah, J. H., Umer, T., & Wan, S. (2019). An automated system based on a novel saliency approach and canonical correlation analysis based multiple features fusion. *Sustainable Computing: Informatics and Systems*, 24, 100349.

Akila, M., & Deepan, P. (2018). Detection and classification of plant leaf diseases by using deep learning algorithm. *International Journal of Engineering Research & Technology (IJERT),* 6(07), 1-5.

Al-Hiary, H., Bani-Ahmad, S., Reyalat, M., Braik, M., & Alrahamneh, Z. (2011). Fast and accurate detection and classification of plant diseases. *International Journal of Computer Applications*, 17(1), 31–38.

Cortes, E. (2017). Plant disease classification using convolutional network and generative adversarial networks. *Reports 2017, Stanford University*, 2017, 4321-4325.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *Computer Vision and Pattern Recognition, Cornell University*. 2015, 1-12.

Kranth, G. P. R., Lalitha, M.H., Basava, L., & Mathur, A. (2018). Plant disease prediction using machine learning algorithms. *International Journal of Computer Applications*, 182(25), 1–7.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems,* 25, 1-9.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998, November). Gradient-based learning applied to document recognition. In *Proceedings of the IEEE* 1998 86(11) (pp. 2278-2324). doi: 10.1109/5.726791

Mangla, N., Raj, P. B., Hegde, S. C., & Pooja, R. (2019). Paddy leaf disease detection using image processing and machine learning. *International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering*, 7(2), 97–99.

Mohanty, S. P., Hughes, D. P., & Salathe, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7(1419), 1-10.

Priya, D., Devisurya, Dharani, Geetha, & Kiruthika. (2021). Cotton leaf disease detection using Faster R-CNN with Region Proposal Network. *International Journal of Biology and Biomedicine*, ISSN: 2367-9085, 6, 23-35.

Priyadharshini, M. K., Sivakami, R., & Janani, M. (2019). Sooty mould mango disease identification using deep learning. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 8(5S), 402-405.

Rehana, H., Ibrahim, M., & Ali, M. A. (2023). Plant Disease Detection using Region-Based Convolutional Neural Network. *Computer Vision and Pattern Recognition, Cornell University*. https://doi.org/10.48550/arXiv.2303.09063

Sapkal, A. T., & Kulkarni, U. V. (2018). Comparative study of Leaf Disease Diagnosis system using Texture features and Deep Learning Features. *International Journal of Applied Engineering Research,* 13(19), 14334–14340.

Shaikh, J. A., & Paradeshi, K. P. (2022). Faster R-CNN Based Rice Leaf Disease Detection method. *TELEMATIQUE*, ISSN: 1856-4194, 21(1), 5890 – 5901.

Simonyan, K., & Zisserman, A. (2015, April). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the ICLR 2015* (pp. 1-14).

Singh, S. K., Mohan, A., & Sundararajan, M. (2023). Isolated Spoken Word Transformation Using Feedforward Neural Network in Speaker Independent Speech Recognition.

Digital Transformation in Sustainable Growth (DTSG 2022), ISBN: 978-81-19149-52-0, pp. 105-114, https://www.kdpublications.in
Singh, S. K., Sundararajan, M., & Yadav, J. (2022). An automatic infant cry speech recognition using artificial neural network. *HYPOTHESIS - National Journal of Research in Higher Studies*, ISSN-2581-8953, V(2); 9-18.
Singh, S. K., & Yadav, J. (2021). Machine Learning & Image Processing for hand written digits and alphabets recognition from document image through MATLAB simulation. *IOP Conf. Series: Material Science and Engineering*, 1084(2021)012021, 1-8. doi: 10.1088/1757-899X/1084/1/012021
Sun, H., Xu, H., Liu, B., He, D., He, J., Zhang, H., & Geng, N. (2021). MEAN-SSD: A novel real-time detector for apple leaf diseases using improved light-weight convolutional neural networks. *Computers and Electronics in Agriculture*, 189 (2021), 1-11.
Wallelign, S., Polceanu, M., & Buche, C. (2018, June). Soybean plant disease identification using convolutional neural network. In *The Thirty-First International Florida Artificial Intelligence Research Society Conference (FLAIRS-31)* (pp. 146-151).

Author's Biography

Vivek Kumar is associated with the Department of Computer Science, Central University of South Bihar, Gaya, Bihar, India. He is working on machine learning in his PhD research work.

Satrughan Kumar Singh is associated with the Department of Computer Science, Central University of South Bihar, Gaya, Bihar, India. His research areas are computer vision, image processing, AI, machine learning, deep learning, cryptography, network security, digital watermarking, speech processing, DBMS, remote sensing, GIS, IT, data modeling, and computer simulation. He is a life member of Vigyan Bharati (VIBHA), India. He is also associated with many engineer associations. He is also a member of the editorial board and reviewer committee in several journals and conferences.

Jainath Yadav is associated with the Central University of South Bihar, Gaya, Bihar, India. His research areas are Speech signal processing, emotions and expressive speech analysis, digital signal processing, machine learning, deep learning, soft computing, and image and audio watermarking.

Muniyan Sundararajan is associated with the Department of Mathematics and Computer Science, Mizoram University, Aizawl, Mizoram, India. His research areas are mathematical modeling, computer simulation, environmental modeling, artificial intelligence, and GIS. He was a senior principal scientist in CSIR before joining Mizoram University, Aizawl, Mizoram, India.