Apple Foliar Disease Detection using Convolutional Neural Network Based Approach

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
SOFTWARE ENGINEERING

Submitted by

SARADINDU DAS

2K21/SWE/21

Under the supervision of

Dr. ABHILASHA SHARMA

(Assistant Professor)

DEPARTMENT OF SOFTWARE ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi 110042

MAY, 2023

DEPARTMENT OF SOFTWARE ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, **Saradindu Das**, Roll No **2K21/SWE/21** student of M.Tech (Software Engineering), hereby declare that the Project Dissertation titled "**Apple Foliar Disease Detection using Convolutional Neural Network Based Approach**" which is submitted by me to the Department of Software Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Saradindu Das

Place: Delhi

Saradindu Das

Date: 24/05/2023

2K21/SWE/21

DEPARTMENT OF SOFTWARE ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "**Apple Foliar Disease Detection using Convolutional Neural Network Based Approach**" which is submitted by **Saradindu Das**, Roll No – **2K21/SWE/21**, Department of Software Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Abhilasha Sharma

Place: Delhi

Dr. Abhilasha Sharma

Date: 24/05/2023

Assistant Professor

Department of Software Engineering

DEPARTMENT OF SOFTWARE ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I wish to express my sincerest gratitude to **Dr. Abhilasha Sharma** for her continuous guidance and mentorship that she provided me during the project. She showed me the path to achieve my targets by explaining all the tasks to be done and explained to me the importance of this project as well as its industrial relevance. She was always ready to help me and clear my doubts regarding any hurdles in this project. Without her constant support and motivation, this project would not have been successful.

Saradindu Das
Place: Delhi
Saradindu Das

Date: 24/05/2023
2K21/SWE/21

Abstract

Agriculture is the process of growing crops and raising livestock and cultivating other forms of food or fiber. It has been a fundamental activity for human societies throughout history, providing food and other resources necessary for survival. It also provides employment, economic growth and environmental conservation. Some farming practices, such as sustainable agriculture, can promote environmental conservation and biodiversity. Hence, plant illness can have a substantial effect on the economy, particularly in agricultural-dependent countries. Crop yield loss, trade restrictions, increased production costs, reduced agricultural productivity, and research and development costs are some of the ways plant diseases can affect the economy. It is essential to prevent and manage plant diseases to ensure food security and maintain a healthy agricultural sector. Farmers visually inspect their crops for symptoms of diseases, such as discoloration, spots, wilting, and deformities. Farmers can also use their sense of touch and smell to detect diseases, such as the sticky or slimy feel of plant leaves infected with fungal diseases and the foul smell of rotting or decaying plant material. However, traditional methods of plant disease detection do have limitations. Visual inspection and other traditional methods may not always detect diseases at an early stage, and there is a risk of misdiagnosis. Additionally, traditional methods may not be able to detect diseases that are not visible to the naked eye.

The other alternatives are the use of Artificial Intelligence (AI) which includes training computers to detect plant diseases using image recognition technology. AI can analyze thousands of images to detect subtle changes in plant health that may indicate the presence of diseases. Some of the regular AI techniques used for plant disease detection are Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), Random Forest (RF), Deep Belief Networks (DBNs), Transfer Learning. These AI methods are increasingly being used for plant disease detection because they offer a fast, accurate, and cost-effective way to diagnose plant diseases, which can help prevent crop losses and increase yields.

Here in this research work, a Multi-layered CNN model is introduced which is inspired from InceptionNet. The proposed model is trained on the “Plant Pathology 2020: FGVC7 dataset” and “Plant Pathology 2021: FGVC8 dataset”. This proposed model is compared with pertained models: InceptionV3. According to the findings, the suggested model outperforms other pre-existing models in terms of accuracy or performance and it’s able to detect the disease with a low error rate.

Contents

List of Table(s)

List of Figure(s)

List of Abbreviation(s)

PD	Plant Disease
PDD	Plant Disease Detection
SVM	Support Vector Machine
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
DBN	Deep Belief Network
AFD	Apple Foliar Disease
NAG	Nesterov Accelerated Gradient
RMS	Root Mean Square
$DCNN$	Deep Convolutional Neural Network
$F-RCNN$	Faster Regions with Convolutional Neural Network
TTA	Test-Time Augmentation
CAE	Convolutional AutoEncoder
APD	Apple Plant Disease
DNN	Deep Neural Network
CV	Computer Vision
NN	Neural Network
NLP	Natural Language Processing
ADC	Analog-to-Digital Converter
SGD	Sochastic Gradient Descent

Chapter 1

INTRODUCTION

Apples are one of the most popular and widely consumed fruits around the world. They are not only delicious but also packed with many essential nutrients and health benefits. Here are some of the benefits of apples: Nutrient-rich, Good for heart health, Boosts immune system, May help regulate blood sugar levels. Overall, apples are a nutritious and delicious fruit that can provide many health benefits when consumed as part of a balanced diet. However there are several diseases that can affect apple plants, some of which are induced by bacteria, viruses, or fungi. These diseases in the plants can be harmful in several ways, both to individual plants and to entire ecosystems. PD[1] can have significant economic impacts, both in terms of lost productivity for farmers and increased costs for consumers. It can have adverse effects on ecosystem function: and can affect the interactions between plants and other organisms, such as pollinators or herbivores, leading to changes in ecosystem function.

Detecting PDs[1] in their early stages is crucial for effective management and control of the disease. Managing apple diseases typically involves a combination of cultural, biological, and chemical control methods, including proper pruning and sanitation, use of disease-resistant varieties, and application of fungicides and bactericides when necessary. Traditionally, PDD involves visual inspection of plants by experienced plant pathologists or farmers. The presence of characteristic symptoms, such as spots, wilting, or deformities, is often used to diagnose the disease. In some cases, laboratory tests may be required to confirm the diagnosis. While these traditional methods are still commonly used for PDD, they can be time-consuming, expensive, and require specialized expertise.

In recent years, new technologies such as remote sensing, AI[2], and biosensors have emerged as promising tools for early and rapid detection of PDs[1]. These new approaches offer faster, more accurate, and cost-effective solutions for disease detection and management. The next alternatives are Imaging techniques which have emerged as promising tools for early detection of PDs[1]. They have the potential to revolutionize PDD by offering fast, non-destructive, and cost-effective solutions for early detection and management of PDs[1]. While imaging techniques offer several advantages for PDD, there are also some disadvantages to consider: Imaging equipment can be expensive, and specialized training may be required to use and interpret the images. Imaging techniques may only capture images of small areas of plant tissue, requiring multiple images to be taken to cover larger plants or fields. Imaging techniques may be impacted by natural factors such as humidity or lighting, which can affect the quality of the images captured. ML[3] techniques like SVMs[4] and K-Means Clustering[5] have shown promise in the detection and diagnosis of PDs[1]. But it comes with the added disadvantage of a complex feature extraction process which reduces the overall efficiency of the system. On the other hand, the DL techniques such as CNNs, RNNs[10], and DBNs[11]. These algorithms are trained

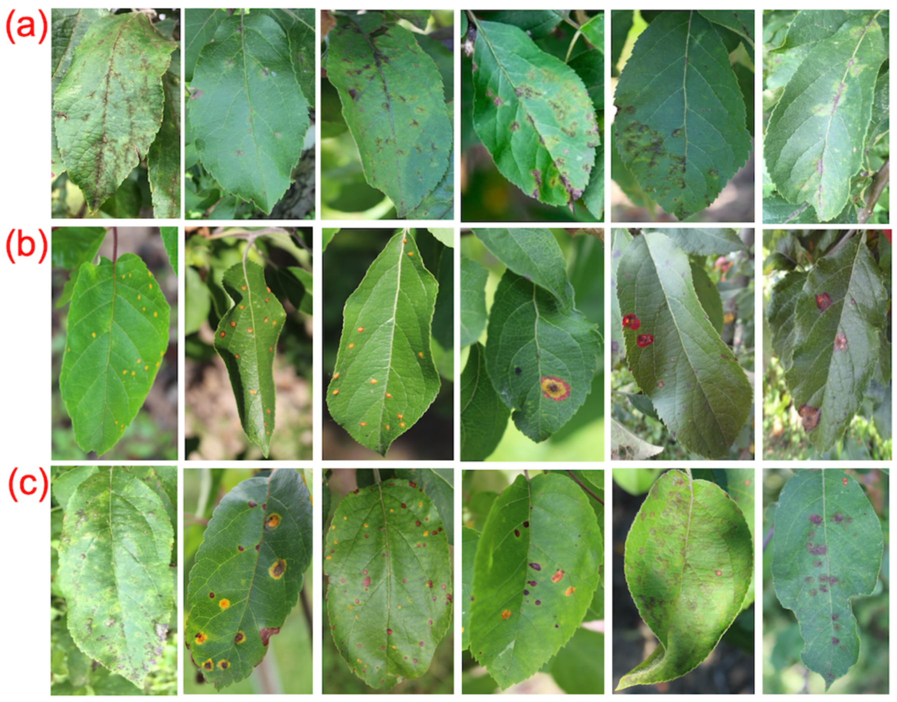


Figure 1.1: Images of disease symptoms on leaves: (a) Indirect sunrays on leaf (b) Direct sunrays on leaf (c) Strong reflection on the leaf[6]

on large datasets of plant images to learn the characteristic features of healthy and diseased plants, enabling automated disease detection and classification with high accuracy. DL[12] algorithms can be trained on large datasets of plant images to automatically detect disease symptoms with high accuracy. This can enable faster and more accurate disease detection, particularly in large-scale agricultural settings. DL[12] algorithms can detect subtle changes in plant tissue that may be difficult for humans or traditional machine learning algorithms to detect, leading to improved accuracy in disease detection. Overall, DL[12] techniques provide an effective method for the timely detection and management of PDs[1], with the potential to improve crop yields, reduce pesticide use, and promote sustainable agriculture. While DL[12] techniques offer many advantages for PDD, there are also several challenges that need to be addressed. Plants can vary widely in their morphology and appearance, making it difficult to develop generalized DL[12] models that can detect a wide range of diseases across different plant species. Labeling and annotating large datasets of plant images can be time-consuming and resource-intensive, particularly for rare or emerging PDs[1].

Here, in this work the FGVC7[13] dataset and FGVC8[14] dataset from the Plant Pathology 2020 and 2021 Kaggle competitions, respectively, are taken. I can observe apple leaves with unhealthy areas in Fig. 1.1 and in Fig. 1.2 I can see the effect of different diseases on apple leaves. CNN technique makes it quite simple to identify them. CNN will be used to identify AFDs[15], and it will be compared to previously trained models InceptionNet. 256x256 is the new size of the images. To ascertain the health of the plants, this process entails classifying photos from the test collection. The suggested paradigm is employed to categorize various illnesses. This work contributes the following:

- This research work examines previously suggested CNN models that are capable of identifying and classifying diseases. Additionally, a new model called Multilayer

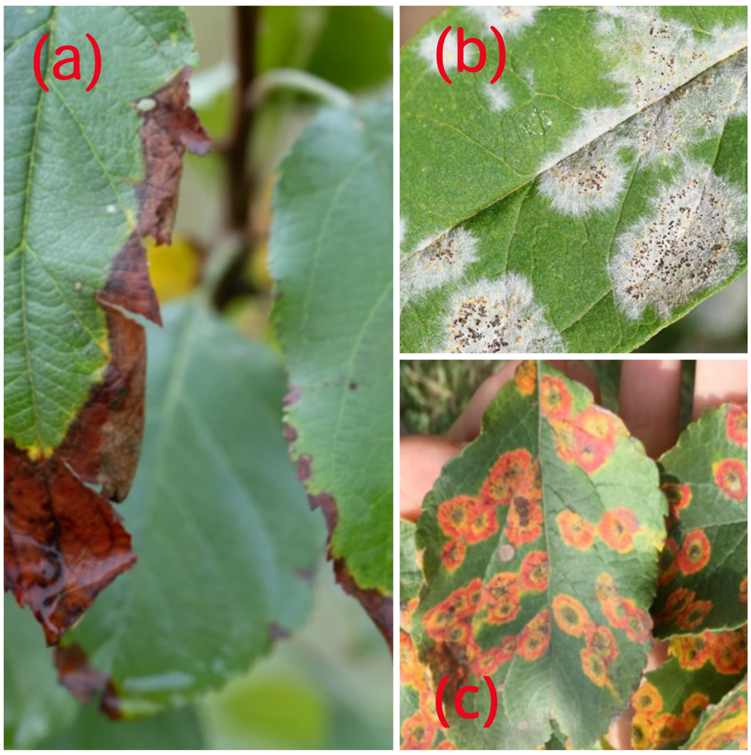


Figure 1.2: Images of different apple leaf diseases: (a) Fire blight (b) Powdery mildew on leaf (c) Cedar apple rust infection on the leaf[7]–[9]

CNN is introduced for this purpose. The newly suggested model is trained using two separate datasets: FGVC7[13] and FGVC8[14].

- The proposed Multilayer CNN model is inspired by InceptionNet and it consists of Convolution Layer, Batch Normalization Layer, Max-Pooling Layer, Flatten Layer, Dense Layer, Dropout Layer and Inception Block.
- Trained with 4 optimizers: Adam, NAG, RMS Prop and RMS Prop with Nesterov momentum and compared the results with each other. Learning rate set to 0.01 in each of the optimizers.
- Also compared the results with two CNN-based models that are already proposed.
- And shows that the proposed model (the Multilayer CNN model) works very well with RMS Prop with Nesterov momentum Optimizer which is trained on Plant Pathology 2021- FGCV8[14] Dataset.

Related work will be covered in Chapter 2 along with methods that have previously been suggested for identifying PDs. Fundamentals will then be covered in Chapter 3. Chapter 4 will provide a summary of the proposed work. Data evaluation will be explored in Chapter 5. Experiments and solutions will be addressed in Chapter 6, followed by the conclusion and future scope and references.

Chapter 2

RELATED WORK

The classification and discovery of plant illnesses are decisive for the agriculture industry, and DL[12] is a crucial module of detecting PD[1]. In this research work, some of the most significant research related to this field is discussed and various DL[12] methods which have been proposed in recent years are examined.

No.	**Ref.**	**Methodology**	**Conclusion**	**Dataset Used**
1.	[16]	With the help of Google Net[17] Inception structure and Rainbow concatenation, DCNN is proposed in this paper. There are five kinds of apple leaf diseases[15] that fiercely affect apple yield, including Alternaria leaf spot, Brown spot, Mosaic, Grey spot, and Rust. This advocated model is correlated with transfer learning[18], [19] approaches.	With Rainbow concatenation and the Inception Structure of Google Net[17], the model gives better accuracy than other transfer learning[18], [19] approaches.	Apple Leaf Disease Dataset (ALDD)
2.	[20]	In this paper, the DL[12] approach which is used is CNN. The suggested CNN model is practiced with an public repository called PlantVillage[21] having 39 distinct classes. Before training, this dataset undergoes six types of data augmentation methods. This proposed model is compared with transfer learning[18], [19] approaches.	Compared to popular transfer learning[18], [19] approaches, data augmentation can improve model performance. With transfer learning[18], [19], DL[12] models can be built with less training data, less time, and fewer computational costs.	PlantVillage[21]

3.	[22]	CNN was developed to identify tomato illnesses exhibit on monitored tomato plants, compared CNN with F-RCNN, and used Transfer Learning for disease recognition on tomato plant leaves.	Before training, data augmentation is done to increase the dataset size and reduce overfitting. Transfer Learning[18], [19] is used as well. In comparison to F-RCNN, the proposed model gives better accuracy.	PlantVillage[21]
4.	[23]	Designed DL[12] based on the NASNet[24] architecture. With the help of techniques such as differential learning rates, cyclical learning rates, and TTA, the model was fine-tuned.	With techniques such as differential learning rates, cyclical learning rates, and TTA, model accuracy is improved without reducing training efficiency. Although there are complex inter- and intra-class variations in the figures of plant leaves, the classification of plant leaves as either diseased or healthy appears promising.	PlantVillage[21]
5.	[25]	Proposed the DenseNet201 model which contains 201 layers. This proposed model is compared with transfer learning[18], [19] approaches.	As a result of this proposed model, the vanishing gradient problem has been solved, feature propagation has been improved, feature reuse has been promoted, and the number of parameters has been reduced. Overfitting occurs when networks have too many connections, which not only reduces their performance and metric capability, but also makes them more susceptible to overfitting.	PlantVillage[21]

6.	[26]	The advocated model is a CNN architecture that incorporates an SVM classifier[4]. The Mean Shift Algorithm is utilized for segmentation, while artificial computation is employed for the isolation of shape features. CNN is used to extract color features. This proposed model identifies 4 kinds of illness in rice plants.	The model gives high accuracy around 96.8% but involves a lengthy training process.	PlantVillage[21]
7.	[27]	The INC-VGG model, which incorporates both VGG19 and InceptionV3, was introduced through the use of transfer learning[18], [19]. The process involved applying three 3X3 ConvBN layers subsequent to the base layers of VGG19 models. This was followed by the application of two InceptionV3 layers and, finally, a Softmax layer.	This proposed model is a combination of two pretrained-model and it gives better accuracy than other pretrained models.	PlantVillage[21]
8.	[28]	Three convolutional layers, three max-pooling layers, and two fully linked layers make up the CNN model that has been advocated. Nine different forms of illnesses that typically damage tomato plants may be recognised using this approach.	This CNN model is Memory Efficient but it's giving low testing accuracy.	PlantVillage[21]
9.	[29]	The suggested CNN model includes six convolution layers and three pooling layers, which are implemented for analysing images of mango plant leaves.	This proposed model is simple and computationally efficient but gives low accuracy.	PlantVillage[21] and images from fields.

10.	[30]	Developed a Deep Siamese CNN. Photos of grape leaves were collected and sorted into four categories.	The proposed model handles the challenge of a small dataset. Although it gives high accuracy, it is not computationally efficient.	Photographs from the field.
11.	[31]	A combined model of CNN and CAE[32] is proposed.	The proposed model produces accurate results. When a reduced dimensional input image is employed, the amount of time required for training is reduced, and the program is capable of naturally detecting crucial features without any human mediation. The orientations are not encoded and the input data is not spatially invariant.	PlantVillage[21]
12.	[33]	An architecture based on CNNs was proposed for the apprehension of nine distinct kinds of diseases on tomato plant leaves. By using conventional architecture such as AlexNet[34] and GoogleNet[17], they developed a classifier.	This proposed model annihilates the need to take out features from figures in order to teach models but involves a lengthy training process.	PlantVillage[21]
13.	[35]	The researchers suggested a DCNN based on Bayesian learning[36] that applies Bayesian learning[36] principles on a residual network. This method combines the strength of Bayesian learning[36] with the effectiveness of residual networks for enhanced performance in disease identification tasks. This model is trained to detect tomato, potato, and pepper bell disease.	The study's outcomes indicate that the suggested method was successful in accurately detecting APDs[15]. This can prove to be beneficial for farmers as it enables early identification and treatment of PDs[1], leading to improved crop quality and yield. Overall, the article demonstrates the potential of DL[12] based approaches for improving PDD in apple crops.	PlantVillage[21]

14.	[37]	The article proposes a method for detecting APD[15] using leaf images through a CNN. The proposed approach for detecting diseases in apple leaves includes collecting a repository of images of both healthy and diseased leaves, pre-processing the images, and practicing a CNN model using the dataset. The article also explains the use of transfer learning[18], [19] and data augmentation techniques to enhance the model's achievement. The results prove that the proposed approach achieved high accuracy in identifying APD[15], which can be beneficial for early detection and treatment of PDs[1], leading to improved crop yield and quality.	The study's outcomes indicate that the suggested method was successful in accurately detecting APDs[15]. This can prove to be beneficial for farmers as it enables early identification and treatment of PDs[1], leading to improved crop quality and yield. Overall, the article demonstrates the potential of DL[12] based approaches for improving PDD in apple crops.	PlantVillage[21]
15.	[38]	The article introduces a complete DL[12] model for classifying corn leaf diseases using a dataset of images comprising both healthy and afflicted corn leaves. The model suggests using a pre-trained CNN architecture and fine-tuning techniques to accomplish substantial accuracy in disease classification. The article also talks about various pre-processing methods employed to improve the model's performance, such as image normalization and data augmentation.	The study demonstrates that the proposed model was effective in accurately classifying different types of diseases in corn leaves. The successful application of the model can provide farmers with a tool to detect PDs[1] at an early stage and take appropriate measures to prevent the spread of diseases, thus improving crop yield and quality. Overall, the article demonstrates the potential of DL[12] based approaches for improving PDD and classification in corn crops.	PlantVillage[21]

16.	[39]	The article presents an approach for detecting PD in cardamom using the EfficientNetV2 DL[12] model. The method consists of several steps, including data collection of healthy and diseased cardamom plants, image pre-processing, and training the EfficientNetV2 model on the collected dataset. The article also discusses the use of transfer learning[18], [19] and data augmentation techniques in order to optimize the efficiency of the model.	The proposed approach, which involves collecting a dataset of images of fit and afflicted cardamom plants, pre-processing the images, and training the EfficientNetV2 model on the dataset, was found to achieve high accuracy in detecting diseases in cardamom plants. This has the potential to assist farmers in early identification and treatment of PDs[1], ultimately leading to improved crop yield and quality. Overall, the article demonstrates the potential of DL[12] based approaches for improving PDD in cardamom plants.	PlantVillage[21]
17.	[40]	The article discusses a performance-optimized DL[12] based approach for detecting PDs[1] in horticultural crops in New Zealand. The approach uses CNNs and transfer learning[18], [19] in order to achieve good accuracy in the detection of a disease. The article discusses the repository utilized to train the CNN models, which contains images of both healthy and afflicted plants. The article also discusses the pre-processing techniques used to enhance the performance of the models.	The outcomes of the study demonstrate that the proposed method accomplished remarkable precision in identifying PDs[1] in horticultural crops, which can assist farmers in detecting and treating them promptly, resulting in improved crop quality and yield. Overall, the article demonstrates the potential of DL[12] based approaches for improving PDD in horticultural crops.	PlantVillage[21]

18.	[41]	The article discusses the use of DL[12] algorithms for detecting PDs[1] using images. It provides an overview of DL[12] models, datasets, and preprocessing techniques used for training these models. The article highlights the relevance of data augmentation, transfer learning[18], [19], and hyperparameter tuning for attaining elevated accuracy in PDD. It also discusses the challenges and future directions of image-based PDD using DL[12].	Overall, the article provides a comprehensive overview of the use of DL[12] algorithms for image-based PDD and highlights the potential for these algorithms to revolutionize agricultural production by enabling early and accurate detection of PDs[1].	All the datasets used in the task of PDD
19.	[42]	Proposes a DL[12] based method for the detection of automatic blight infection in potato and tomato plants. The proposed approach uses a DCNN model to categorize the plant images as healthy or afflicted on the basis of the presence of blight disease.	The suggested model in this research was practiced and assessed using a dataset containing 11,000 plant figures, and was compared to other advanced techniques. The results showed that it outperformed other methods in terms of accuracy and computational efficiency. This presents a promising DL[12]-based technique for the automatic detection of blight illness in tomato and potato plants, which could be practically useful in agriculture for early disease detection and management.	PlantVillage[21]

20.	[43]	Proposes a DL[12] based multi-task prediction system for detecting both PD[1] and plant species in images. The suggested system is comprised of two key components: the feature extraction network and the prediction network. The feature extraction network employs a pre-trained CNN model to derive significant attributes from the input images, while the prediction network consists of two branches. One branch is for disease detection and the other for species identification, both of which use the extracted features to make precise predictions. The two branches share the feature extraction network, which allows for joint learning of both tasks and improves prediction accuracy.	The study involves the use of a dataset comprising 10 different diseases and 9 plant species to train and evaluate the proposed system. Additionally, the paper compares the proposed system with other advanced approaches and demonstrates its superior performance in terms of accuracy. The findings of this study present a promising solution for multitask detection of PD[1] and species using DL[12], with potential applications in the field of agriculture and plant science.	Images captured from the farm.
21.	[44]	They studied the AlexNet[34] and GoogLeNet[17] DNN architectures and achieved a categorization proficiency of 99.55% on the PlantVillage[21] dataset using 50k sample images over 30 training epochs.	AlexNet[34] and GoogLeNet[17] DNN architectures give more accurate results as compared to simple CNN models.	PlantVillage[21]
22.	[45]	The authors developed a three-layer CNN system with an average accuracy of 94.9% after 40 epochs of training. This study used 800 leaf images of cucumber.	This model is prone to overfitting because it uses only 800 leaf images. And also, a simple model which uses only three layers.	Images captured from the farm.

Table 2.1: Summarized review of literature papers

The following are some of the key takeaways from the preceding table:

1. Data augmentation has shown to be a highly successful strategy for improving overall performance and making the model more resilient to noise.
2. Oversampling strategies for adjusting for uneven numbers of entities increase performance in classes with unequal numbers of entities.
3. For image classification tasks, DL[12] models are more accurate than machine learning models.
4. The classification accuracy improves as the number of photos per class used to train the models grows.
5. After achieving a particular high degree of accuracy, expanding the model's complexity to better suit the job yields decreasing returns.
6. When the model is trained for a small number of epochs (less than 20), the model frequently does not acquire enough features from the input to correctly categorize the test pictures.
7. The main advantage of using DL[12] Technique instead of ML[3] Technique is that DL[12] does Feature Extraction and Classification by itself but in ML[3] we do Feature Extraction Manually.

Chapter 3

FUDAMENTALS

The technology of the InceptionNet model based on CNN architecture has been put to use to detect diseases in apple plant leaves[15]. In this section, the preliminary of such a technique is explained and the python libraries are listed which have been used in the suggested implementation such as Keras, TensorFlow, and others.

3.1 Plant Disease

PD[1] are caused by various species such as bacteria, viruses, fungi, nematodes, and other pathogens, as well as non-infectious factors such as environmental stress, nutrient deficiencies, and physical damage. The symptoms of these diseases, which include wilting, discoloration, malformations, and slowed development, can affect many sections of the plant, including the leaves, stem, roots, flowers, and fruits. PD[1] have the potential to spread quickly, seriously harm gardens and crops, and diminish yields as well as create financial losses. They can be managed through a variety of techniques, including social norms, drugs, biological regulation, and genetic resistance. PD[1] can be reduced in severity by taking preventative measures such as good sanitation, crop rotation, and the use of disease-free seeds.

3.1.1 Classification of Plant Disease

PD can be classified based on the type of pathogen causing the disease or based on the symptoms they cause. Here are some common classifications of PDs[1]:

1. Based on the type of pathogen:

    - Fungal diseases: Instigated by fungi, such as blight, rust and powdery mildew.
    - Bacterial diseases: Induced by bacteria, such as soft rot, leaf spot and bacterial canker.
    - Viral diseases: Triggered by viruses, such as yellow vein mosaic, leaf curl and other mosaic virus.
    - Nematode diseases: Caused by nematodes, such as cyst nematode, lesion nematode and root-knot nematode.
    - Phytoplasma diseases: Caused by phytoplasma, such as aster yellows and stol bur.
    - Protozoan diseases: Caused by protozoans, such as downy mildew and plas modiophoromycetes.

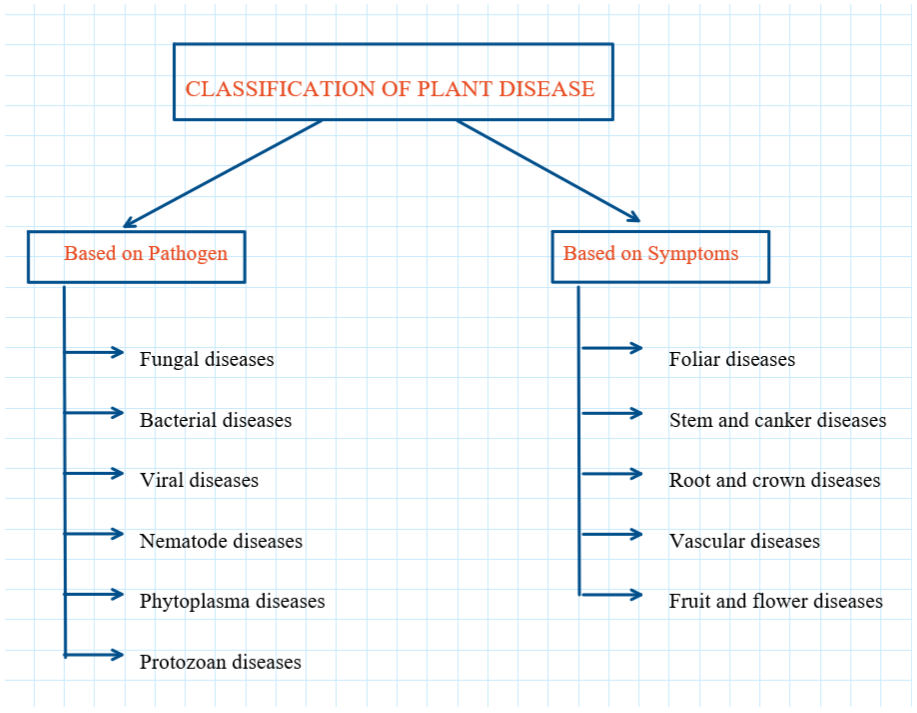


Figure 3.1: Classification of plant diseases

2. Based on the symptoms:

   - Foliar diseases: affecting leaves, such as leaf spot, blight, and rust.
   - Stem and canker diseases: affecting stems, such as canker, wilt, and dieback.
   - Root and crown diseases: affecting roots and the base of the plant, such as root rot and crown rot.
   - Vascular diseases: affecting the plant's transport system, such as wilts and yellows.
   - Fruit and flower diseases: affecting fruits and flowers, such as fruit rot and blossom blight.

3.1.2 Apple Foliar Disease

Apple trees are susceptible to several foliar diseases, which can substantially abate the quality and yield of the fruit. Here are some common foliar diseases of apple trees:

1. Apple scab: This particular foliar disease is prevalent and can cause significant damage to apple trees. It is induced by the fungus known as Venturia inaequals, which infects the leaves, fruit, and twigs of the tree. Apple scab causes brown or olive-green lesions on the leaves, which can cause them to fall off prematurely. The disease can also cause scabbing on the fruit, which reduces their market value.

2. Cedar apple rust: This disease is caused by the fungus Gymnosporangium juniperi virginianae, which requires two hosts to achieve its life cycle: apple trees and juniper trees. The disease causes yellow spots on the leaves, which develop into orange or brown spore-producing structures. It can also cause deformed fruit and stunted growth in severe cases.

3. Powdery mildew: This disease is instigated by several species of fungi, including Po dosphaera leucotricha and Podosphaera tridactyla. It causes a white or grayish powdery coating on the leaves and can stunt the growth of the tree if left untreated.

4. Fire blight: This bacterial disease is induced by Erwinia amylovora and can cause significant damage to apple trees. It causes a wilted, burnt appearance in the leaves, which can quickly spread to the branches and fruit of the tree. Infected branches should be pruned and removed immediately to prevent the spread of the disease.

These are just a few examples of foliar diseases[15] that can affect apple trees. Preventative measures such as proper pruning and sanitation practices, as well as the use of fungicides, can help manage these diseases and protect the health and productivity of apple trees.

3.2 Convolutional Neural Network

A CNN is a kind of DNN[46] which is frequently incorporated in CV[47] tasks such as image and video validation. It is a specialized NN[46] architecture that is designed to manage data with a grid-like topology, such as figures, by performing a series of convolutional and pooling operations. The main building blocks of a CNN are convolutional level, pooling level, and fully connected level. Convolutional layers are obliged to isolate features from input data by enforcing a series of filters that slide over the input data and output a set of feature maps. Pooling layers are used to downsample the feature maps and diminish the spatial dimensionality of the data. Fully connected layers are utilized to classify the input data based on the extracted features. During training, the weights of the filters in the convolutional layers are adjusted through backpropagation, which is a process that involves computing the gradient of the loss function in reference to the network parameters and updating them accordingly. This process is repeated over many iterations until the network learns to recognize patterns in the input data and produce accurate predictions. CNNs have achieved advanced performance on a varied range of CV[47] tasks, including object detection, image classification, and semantic segmentation. They are also commonly used in other domains such as NLP[48] and speech recognition[49], where the input data can be represented as a grid-like structure.

3.3 InceptionNet

InceptionNet, also known as GoogLeNet[17], is a CNN architecture developed by researchers at Google in 2014. It was designed to upgrade the reliability of image segmentation tasks while abating the number of attributes required in the system. The InceptionNet architecture incorporates several modules called Inception modules, each of which contains multiple parallel convolutional layers with different filter sizes. These parallel layers are concatenated along the depth axis, allowing the network to capture features at different spatial scales. In addition to the convolutional layers, InceptionNet also includes pooling layers, batch normalization layers, and fully connected layers. One of the key features of InceptionNet is the use of 1x1 convolutions, which are used to reduce the number of channels in the input feature maps and reduce computational complexity. In addition, InceptionNet also uses a technique called "bottleneck" layers, which use 1x1 convolutions to reduce the dimensionality of the input before applying larger convolutional filters. InceptionNet was able to achieve innovative competence on the ImageNet dataset,

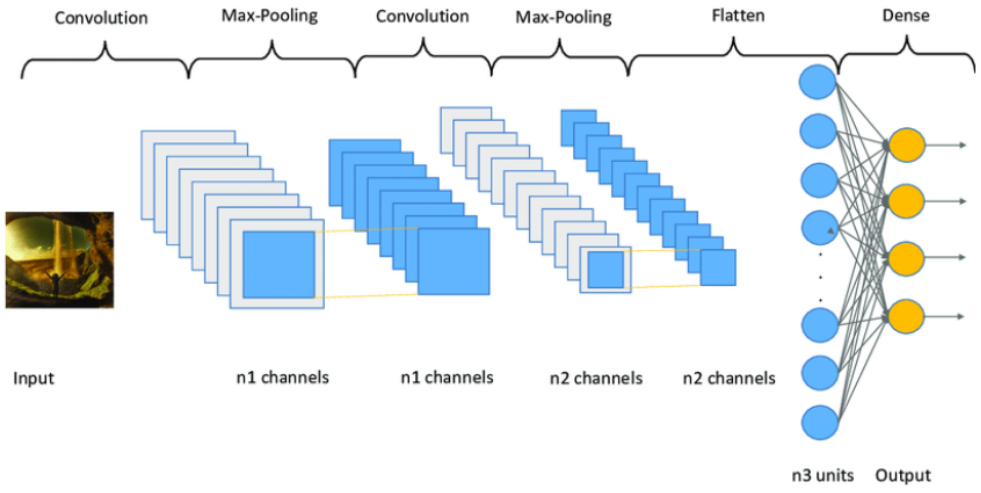


Figure 3.2: Convolutional Neural Network[50]

a large-scale image recognition dataset, while using fewer parameters than previous advanced models. This has made it a popular choice for image classification tasks and has inspired the development of other Inception-based architectures, such as Inception-v3[51] and Inception-ResNet[52].

When the Inception module is dissected into its constituent parts, it is simple to unpack and comprehend. Little-experienced DL[12] practitioners can get some benefit by knowing the hypotheses of the researchers who created the Inception network, as well as certain conventional techniques and terminology used in more recent CNN.

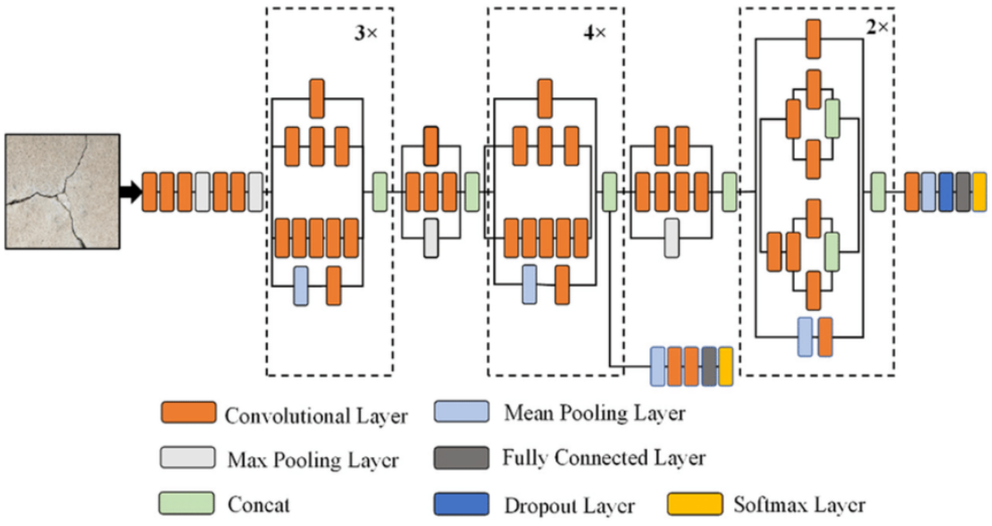


Figure 3.3: InceptionNet[53]

Chapter 4

PROPOSED WORK

4.1 Problem Statement

Farming, as we all know, is the art of growing crops, and it is essential to the survival of humanity as a whole. As a result, its production rate ought to be high. A plant's sickness will have an impact on its productivity rate. Thus, it is important to identify PDs[1] as soon as possible. The process of detecting PDs[1] entails creating a precise and effective mechanism for doing so. PD[1], which may significantly affect crop output and quality, can be brought on by a variety of agents, including bacteria, fungus, viruses, and environmental circumstances. To stop the spread of a PD[1] and reduce crop damage, early discovery and diagnosis are essential.

PD[1] diagnosis has traditionally depended on visual inspection by skilled professionals, which may be time-consuming, expensive, and prone to human mistake. The development of automated PDD systems utilizing ML[3] and CV[47] methods is a result of technological advancements. In order to categorize the photographs as healthy or ill, these systems generally require taking pictures of plants, collecting pertinent information, and then utilizing ML[3] techniques. The objective is to provide reliable and effective tools for PD[1] identification that farmers and agricultural professionals can use to manage crop health.

4.2 Solution

A multilayered CNN model is suggested in this section. The first paragraph of this section describes the data gathering procedure. The second and third subsections address image processing and data augmentation, while the fourth and fifth subsections explain the proposed model and the optimizers that were utilized. The suggested solution's overall workflow is shown in Fig. 4.1.

4.2.1 Data Acquisition

Data acquisition is the process of collecting data from various sources and converting it into a digital format that can be processed by a computer or other electronic device. Data acquisition systems are used in various possible implementations, from scientific research to industrial process control. The process of data acquisition typically involves three main components:

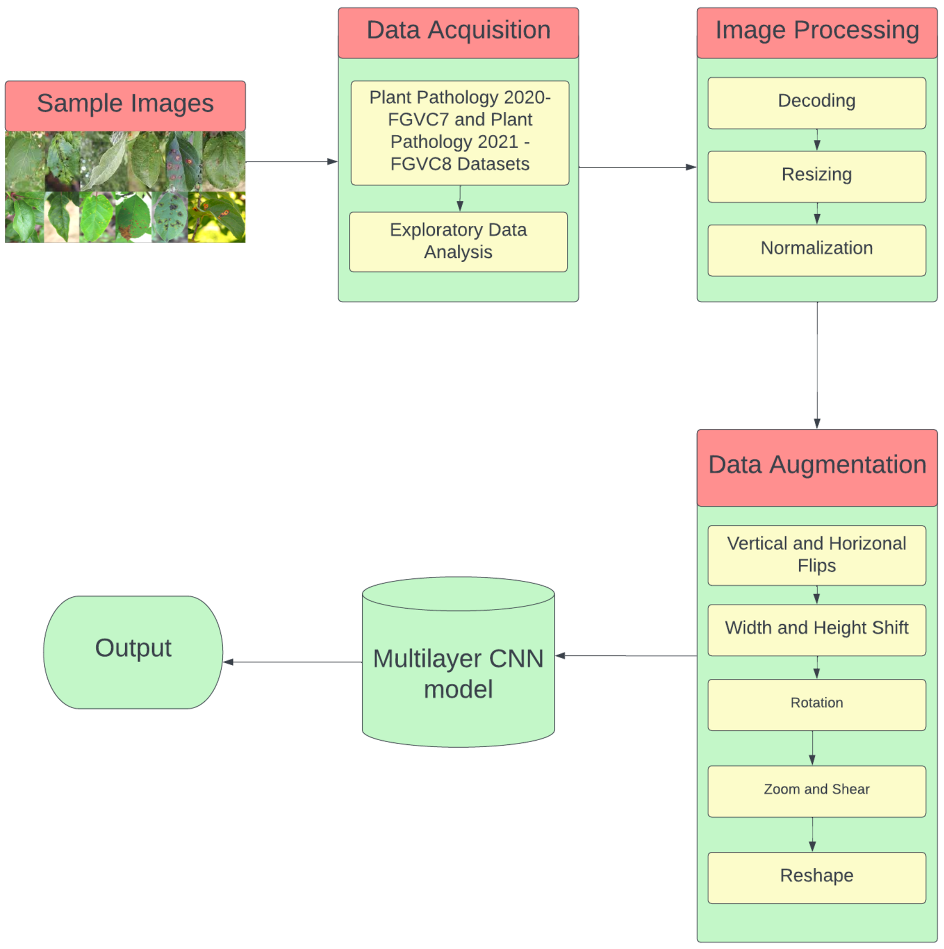


Figure 4.1: Graph showing proposed methodology process

1. Sensors or instruments: These are devices that measure physical or electrical properties of the environment, such as temperature, pressure, or voltage.

2. Data acquisition hardware: This is the equipment that is used to connect the sensors or instruments to a computer or other electronic device. Data acquisition hardware typically includes ADCs that convert the analog signals from the sensors into digital signals that can be processed by a computer.

3. Data acquisition software: This is the software that is used to control the data acquisition hardware and to collect, store, and analyze the data that is being acquired. Data acquisition software can include drivers for the data acquisition hardware, user interfaces for configuring and controlling the hardware, and data processing and analysis tools.

Data acquisition systems can be designed for a wide range of applications, from simple data logging to complex process control and monitoring systems. Some common applications of data acquisition systems include:

1. Scientific research: Data acquisition systems are used in scientific research to collect and analyze data from experiments and observations.

2. Industrial process control: Data acquisition systems are used in industrial process control to monitor and control manufacturing processes, such as temperature control in chemical reactions or pressure control in oil drilling.
3. Environmental monitoring: Data acquisition systems are used in environmental monitoring to measure and analyze air and water quality, weather patterns, and other environmental factors.
4. Medical monitoring: Data acquisition systems are used in medical monitoring to measure and analyze patient data, such as heart rate and blood pressure.

Overall, data acquisition plays a crucial role in collecting and analyzing data from various sources, allowing for better decision making, process control, and scientific research.

4.2.2 Image Processing

Image processing is a field that deals with the manipulation and analysis of digital images. It involves the use of various algorithms and techniques to transform, enhance, or extract information from images. Some of the common techniques used in image processing include image filtering, edge detection, image segmentation, image restoration, and feature extraction. These techniques can be used individually or in combination to achieve various image processing goals. There are many applications of image processing, including:

1. Medical imaging: Image processing is used to analyze medical images, such as X rays, MRI scans, and CT scans, to aid in the diagnosis and treatment of medical conditions.
2. Security and surveillance: Image processing is used to enhance and analyze surveillance camera images, as well as to identify and track objects or people of interest.
3. Robotics and automation: Image processing is used in robotics and automation systems to provide vision capabilities, such as object detection and recognition.
4. Entertainment: Image processing is used in the creation of visual effects for movies and video games.

In recent years, DL[12] techniques such as CNNs have also been used in image processing, achieving advanced results in various image related tasks which are object detection, segmentation, and image classification.

4.2.3 Data Augmentation

This is a method of expanding the magnitude of a dataset by employing different transformations or modifications to the authentic data. The intention of data augmentation is to expand the diversity of the dataset, which can boost the performance of machine learning models by reducing overfitting and improving generalization. There are several techniques that can be used for data augmentation, depending on the type of data and the task at hand. Some common techniques include:

1. Image augmentation: This involves applying transformations such as rotation, flip ping, scaling, cropping, and color adjustments to images.

2. Text augmentation: This involves applying techniques such as synonym replacement, word deletion, and word swapping to text data.

3. Audio augmentation: This involves applying transformations such as time stretching, pitch shifting, and noise addition to audio data.

The data augmentation process typically involves the following steps:

1. Selection of the augmentation technique: The first step is to select the appropriate augmentation technique based on the type of data and the task at hand.

2. Configuration of the augmentation parameters: Each augmentation technique has several parameters that can be configured to control the degree of transformation. For example, the degree of rotation or the amount of noise added to an image can be controlled by adjusting the parameters.

3. Application of the augmentation: The augmentation technique is applied to the original data to generate new augmented data points.

4. Incorporation of the augmented data into the dataset: The augmented data is then added to the original dataset to increase its size and diversity.

Data augmentation can be performed manually, but it can also be automated using various libraries like PyTorch, Keras and TensorFlow. These libraries provide built in functions for common augmentation techniques, making it easy to incorporate data augmentation into the machine learning pipeline.

4.2.4 Proposed Model

A CNN based model called Multilayer CNN is proposed which contains Convolution Layer, Batch Normalization Layer, Max-Pooling Layer, Flatten Layer, Dense Layer, Dropout Layer and Inception Block. Here Inception Block is inspired from InceptionNet. And in this proposed architecture, ReLU[54], LeakyReLU[55] and Softmax[56] activation functions are used. Multilayer CNN architecture is given in Table 4.1. In Multilayer CNN model, there are 5 Convolution Layers, 1 Batch Normalization Layers, 2 Max-Pooling Layers, 1 Flatten Layer, 4 Dense Layers (3 layers having ReLU[54] activation function and 1 layer having Softmax[56] activation function), 6 Dropout Layers having dropout rate 0.25, 0.3 and 0.5 and 2 Inception Module.

S.No.	**Name**	**Activations**	**Total Learnables**
1.	Image_Input	256x256x3	0
2.	conv2d	254x254x12	3584
3.	dropout	254x254x612	0
4.	conv2d_1	252x252x64	73792
5.	max_pooling2d	126x126x64	0
6.	dropout_1	126x126x64	0
7.	inception_module	126x126x16	0
8.	conv2d_2	124x124x32	46112
9.	dropout_2	124x124x32	0

10.	conv2d_3	122x122x16	4624
11.	max_pooling2d_1	61x61x16	0
12.	dropout_3	61x61x16	0
13.	inception_module	61x61x80	0
14.	conv2d_4	59x59x8	5768
15.	max_pooliing2d_2	29x29x8	0
16.	dropout_4	29x29x8	0
17.	flatten	6728	0
18.	dense	4096	27561984
19.	dense_1	1024	4195328
20.	dense_2	128	131200
21.	dense_3	• 4 (Plant Pathology 2020) • 12 (Plant Pathology 2021)	• 516 • 1548
	Total params:		• 32,126,244 (Plant Pathology 2020) • 32,127,276 (Plant Pathology 2021)

Table 4.1: Detailed Architecture of Multilayer CNN Model

Detailed explanation of each layer is given below:

1. Input: Defines the shape of the input tensor, which is (input image, input image, 3) for this model.

2. Conv2D: Performs 2D convolution on the input tensor with a specified number of filters (64 in the first layer), kernel size of 3x3, and padding of 'same', which means the output feature maps will have the same spatial dimensions as the input.

3. BatchNormalization: Normalizes the outputs of the preceding layer to ensure the mean activation is approximately near to 0 and the activation standard deviation is close to 1.

4. Activation: Applies the ReLU[54] activation function to the output of the previous layer.

5. MaxPooling2D: Downsamples the input tensor along the spatial dimensions by taking the maximum value within a specified pool size (default is 2x2) and strides (default is pool size).
6. inception_module: A custom layer that applies the inception module that necessitates the result of the preceding layer as input and returns a feature map with increased depth through concatenation of different convolutional filters.
7. Flatten: Used to convert the 3D output of a convolutional layer into a 1D vector that can be sustained into a fully connected section. The Flatten layer essentially "flattens" the result of the preceding layer into a long, one-dimensional array of values.
8. GlobalAveragePooling2D: Takes the average of each attribute map in the previous layer along the spatial dimensions to reduce the tensor to a similar length vector as that of the number of filters.
9. Dense: A fully connected layer that necessitates the vector result of the preceding layer and applies a specified number of neurons (9216, 4096, 1024, and 128 in this model) with ReLU[54] activation.
10. Dropout: Randomly sets a fraction of input units to 0 at each update during training to prevent overfitting.
11. BatchNormalization: Applies batch normalization to the result of the preceding layer.
12. Dense: A fully connected layer that takes the output of the previous layer and applies a softmax activation to classify the input into one of 4 possible classes(in case of Plant Pathology 2020-FGCV7[13]) or 12 possible classes(in case of Plant Pathology 2021-FGCV8[14]) [as defined by the number of neurons in the output layer].

The Module that is defined in this work is Inception Module

4.2.5 Inception Module

Inception Module is inspired by InceptionNet. Inception Net is a DCNN architecture that uses inception modules, which allow for efficient information processing and feature extraction at multiple scales. Inception Module in the proposed work contains 3 Convolution Layers of kernel size 1x1 with activation function ReLU[54] and also the padding is same, 3 Convolution Layers of kernel size 3x3 with activation function ReLU[54] and also the padding is same, 1 Convolution Layers of kernel size 5x5 with activation function ReLU[54] and also the padding is same and 1 Max-Pooling Layer having pool size of 3, strides of 1x1, the padding is same and the Concatenation Layer to merge some Layers.. The architecture of InceptionNet is given in Fig.4.1. In Fig. 4.1, The Inception Module takes an input tensor of shape (h, w, c) and applies a series of operations to it to produce an output tensor of shape (h, w, 4 * filter 3x3). The module consists of four branches of convolutional layers: a 1x1 convolution branch, a 3x3 convolution branch, a max pooling and 1x1 convolution branch, and a 1x1 convolution followed by 5x5 convolution and then 3x3 convolution branch. The output of each branch is concatenated along the channel axis to generate the final result. The diagram shows the flow of data through the Inception module, starting with the input tensor and ending with the output tensor. The operations performed at each layer are shown, along with the shape of the tensors at each stage. The output tensors of each branch are combined by concatenating them along the channel axis to form the final output tensor.

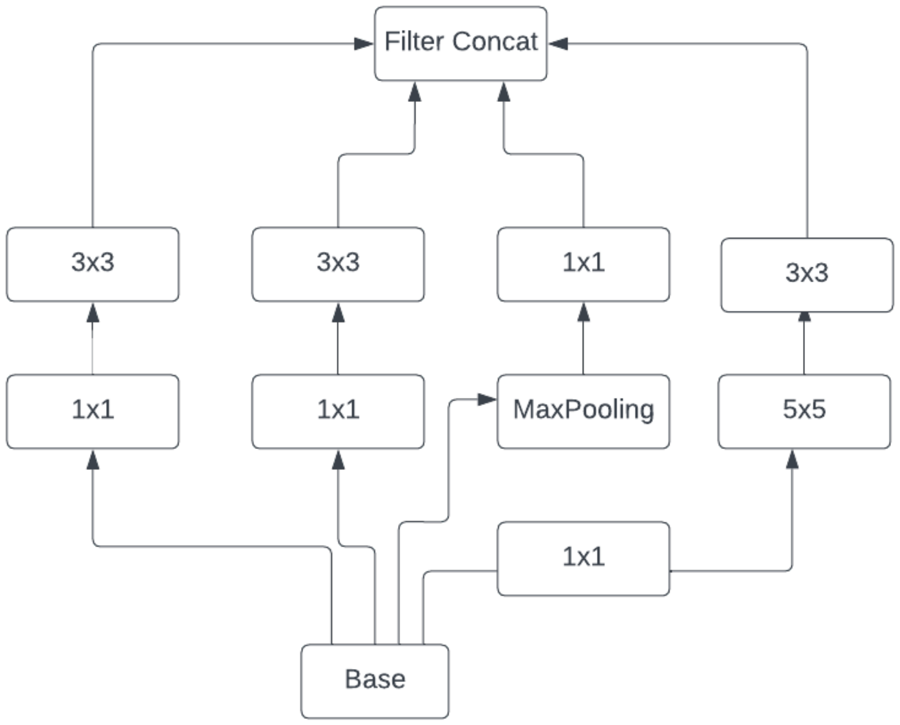


Figure 4.2: Architecture of Inception Module

In Fig. 4.1, The Inception Module takes an input tensor of shape (h, w, c) and applies a series of operations to it to produce an output tensor of shape (h, w, 4 * filter_3x3).

4.2.6 Weight Initializers Used

In this proposed Multilayer CNN model, two kernel initializers are used: LeCun-uniform initializer and another one is He-uniform initializer. The LeCun-uniform kernel initializer and the He-uniform kernel initializer are two commonly used methods for initializing the weights of convolutional kernels in neural networks. The LeCun-uniform kernel initializer is named after Yann LeCun, and is designed to initialize the weights of kernels in a way that helps to prevent gradients from vanishing or exploding during training. The initializer uses a uniform distribution to randomly initialize the weights of each kernel, with a range determined by the formula sqrt(3/fan in), where fan in is the number of input channels to the layer. This initialization method is commonly used in convolutional layers of NNs[46], and has been shown to work well in practice for a wide range of tasks. The He-uniform kernel initializer, on the other hand, is named after Kaiming He, and is a modified version of the LeCun-uniform initializer that is specifically designed for use with ReLU[54] activation functions. The He-uniform initializer sets the range of the uniform distribution used to initialize the weights to sqrt(6/fan in), where fan in is the number of input channels to the layer. This initialization method is preferred when using ReLU activation functions because it helps to prevent the "dying ReLU" problem, where a large number of neurons in the network can become permanently "dead" (i.e. outputting zero) due to a zero gradient during training. Both the LeCun-uniform and He-uniform kernel initializers have been shown to work well in practice for a variety of DL[12] tasks, and are widely used in advanced neural network architectures. The choice of which initializer to use may depend on the specific requirements of the task at hand, and the characteristics of the dataset being used for training. In this work, LeCun-uniform Kernel Initializer is used in convolution layers during defining Inception Module and Dense Module and He-uniform Kernel Initializer is used in convolution layers during defining the overall model.

4.2.7 Optimizers Used

In this proposed work, 4 optimizers are used to check which optimizer gives the better results with the proposed Multilayer CNN model. Optimizers are: Adam, NAG, RMS Prop and RMS Prop with Nesterov momentum. Optimizers in DL[12] are algorithms used to update the weights and biases of a NN[46] during training in order to mitigate the loss function. The choice of optimizer can have a considerable impression on the performance of the model, and there are several different optimizers available to choose from. Here's a brief overview of each optimizer used in this work:

1. **Adam:** The Adam update rule combines the benefits of both momentum and RMSprop, and adapts the learning rate for each parameter based on the estimated first and second moments of the gradient. The update guideline for the Adam optimizer can be conveyed mathematically as stated below:

   - Calculate the gradient of the objective function in reference to the model components:
   $$g_t = \nabla_\theta J(\theta_t) \tag{4.1}$$
   where g_t is the gradient at time step t, θ_t is the model parameters at time step t, and $J(\theta_t)$ is the objective function at time step t.
   - Update the exponential moving average of the first moment of the gradient:
   $$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.2}$$
   where m_t is the first moment estimate at time step t, β_1 is a hyperparameter that controls the decay rate of the moving average, and m_0 is initialized to 0.
   - Update the exponential moving average of the second moment of the gradient:
   $$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{4.3}$$
   where v_t is the second moment estimate at time step t, β_2 is a hyperparameter that controls the decay rate of the moving average, and v_0 is initialized to 0.
   - Compute the bias-corrected first and second moment estimates:
   $$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{4.4}$$
   $$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{4.5}$$
   where t is the current time step.

- Update the model parameters:

$$\theta_{t+1} = \theta_t - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{4.6}$$

where α is the learning rate, ϵ is a small constant added for numerical stability, and $\sqrt{}$ denotes the square root operation.

2. **Nesterov Accelerated Gradient:** Nesterov Accelerated Gradient (NAG) is a vari ant of the standard stochastic gradient descent (SGD) optimizer, which improves convergence by using a "lookahead" update of the gradient. This lookahead update takes into account the current momentum in the optimization process and allows the optimizer to more accurately estimate the direction of the gradient at the next time step.
The update rule for Nesterov Accelerated Gradient is as follows:

$$v_t = momentum * v_{t-1} + learning_rate * gradient(w_{t-1} - momentum * v_{t-1}) \tag{4.7}$$

where w_t is the value of the weights at iteration t, v_t is the momentum term at iteration t, learning rate is the learning rate, and momentum is the momentum coefficient.

3. **RMS Prop:** RMSProp (Root Mean Square Propagation) is an optimization algorithm commonly used in DL[12] to update the weights of a neural network during training. It was introduced by Geoff Hinton in 2012 as an extension to the SGD algorithm. The key idea behind RMSProp is to adjust the learning rate of each weight based on the root mean square of its gradients. Specifically, RMSProp maintains a moving average of the squared gradients for each weight, and divides the learning rate by the square root of this moving average. This has the effect of scaling down the learning rate for weights that have large, fluctuating gradients, while allowing the learning rate for weights that have small, consistent gradients to remain relatively high. The RMSProp algorithm has been shown to improve the convergence speed and generalization performance of NNs[46] compared to traditional SGD. It is particularly effective for networks with sparse gradients, such as those commonly encountered in NLP[48] and CV[47] tasks.

The update rule for RMSProp can be expressed using the following formula:

$$cache = decay_rate * cache + (1 - decay_rate) * gradients * *2 \tag{4.8}$$

$$w = w - learning_rate * gradients/(sqrt(cache) + epsilon) \tag{4.9}$$

where:
gradients are the steepness of the cost function with respect to the weights. cache is a moving average of the squared gradients, initialized to 0. decay rate is a hyper parameter that controls the exponential decay of the moving average, typically set to 0.9. learning rate[57] is the learning rate used for updating the weights. epsilon is a small constant (e.g., 1e-8) added to the denominator for numerical stability. w are the weights of the neural network. The first line updates the cache by taking

a weighted average of the previous cache value and the squared gradients. The second line updates the weights by dividing the gradients by the square root of the cache (which scales the learning rate for each weight), and subtracting this value from the weights. This update rule allows the algorithm to adaptively adjust the learning rate for each weight based on the magnitude of its gradients, leading to more efficient and effective training of DNNs.

4. **RMSprop with Nesterov Momentum:** RMSprop with Nesterov Momentum is a variation of the RMSprop optimization algorithm that incorporates Nesterov momentum. Nesterov momentum is a technique that accelerates the convergence of the optimization algorithm by taking into account the momentum of the gradients. The update rule for RMSprop with Nesterov Momentum is similar to that of regular RMSprop, but with an additional momentum term added to the gradients:

$$cache = decay_rate * cache + (1 - decay_rate) * gradients * *2 \tag{4.10}$$

$$\begin{multline} momentum = momentum_rate * momentum - \\ learning\_rate * gradients/(sqrt(cache) + epsilon) \end{multline} \tag{4.11}$$

$$\begin{multline} w = w + momentum_rate * momentum - learning_rate * \\ gradients/(sqrt(cache) + epsilon) \end{multline} \tag{4.12}$$

where:
gradients are the steepness of the cost function with respect to the weights. cache is a moving average of the squared gradients, initialized to 0. decay rate is a hyperparameter that controls the exponential decay of the moving average, typically set to 0.9. momentum rate is a hyperparameter that controls the momentum term, typically set to 0.9. learning rate[57] is the learning rate used for updating the weights. epsilon is a small constant (e.g., 1e-8) added to the denominator for numerical stability. w are the weights of the NN[46]. momentum is the momentum vector, which is initialized to 0. The first line updates the cache by taking a weighted average of the previous cache value and the squared gradients. The second line updates the momentum vector by taking a weighted average of the previous momentum and the gradients. The third line updates the weights by subtracting the scaled gradients (using the updated momentum vector) from the current weights. By using Nesterov momentum, the algorithm is able to incorporate information about the momentum of the gradients into the update rule, leading to faster convergence and better generalization performance.

Chapter 5

DATA EVALUATION

In this model, two datasets, Plant Pathology 2020: FGCV7[13] and Plant Pathology 2021: FGCV8[14] have been used for the recommended approach. To gather the experimental data for the proposed method, Python 3.0 on an Intel i5 9th generation CPU with 12 GB of memory and an NVIDIA K80 GPU with 12 GB of memory have been used. We are now going to read the in-depth descriptions of the datasets that has been compared in the section below.

5.1 Datasets Used

Two real-world datasets, Plant Pathology 2020: FGCV7[13] and Plant Pathology 2021: FGCV8[14] have been used to identify a certain disease affecting apple plants. These datasets are shown in Table 2 The format and organization of all the datasets have been described in Table 5.1. This is a summary of the datasets that have been used.

Dataset Name	**Training Images**	**Testing Images**	**Total Images**
Plant Pathology 2020: FGCV7[7]	14,707	4,008	18,715
Plant Pathology 2021:FGCV8[8]	14,906	3,726	18,632

Table 5.1: Distribution of Dataset

1. **Plant Pathology 2020: FGCV7**
   The Kaggle community got access to a collection of images of leaves with various diseases like apple scab, cedar apple rust, and healthy leaves. This was the Plant Pathology 2020 - FGVC7 dataset, which was part of the 'Plant Pathology Challenge' in the Fine-Grained Visual Categorization (FGVC) workshop at CVPR 2020 (Computer Vision and Pattern Recognition). The goal of this challenge was to use the images of the training dataset to train a model that can correctly classify an image from the testing dataset into different disease categories or a healthy leaf. The dataset's distribution of these categories is as follows:

   - Healthy: 5162 images (27.6%)
   - Multiple Diseases: 1420 images (7.6%)
   - Rust: 3166 images (16.9%)

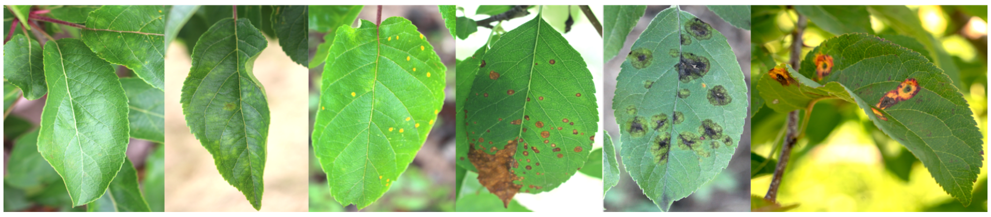


Figure 5.1: Sample Images of Plant Pathology 2020-FGCV7 Dataset[13]

The sample images of Plant Pathology 2020-FGCV7[13] dataset are given in Fig. 5.1. Some leaves may have more than one disease or condition, so they may belong to more than one label. The "Other" label is for images that do not fit into any of the four main categories, or that have labels that are not clear or definite. The dataset has more images of some classes than others, with the "Healthy" class having the most images and the "Multiple Diseases" class having the fewest. This can make it hard to train machine learning models that are good at all classes, and researchers and developers may have to use methods such as data augmentation and class balancing to make the model better.

2. **Plant Pathology 2021: FGCV8**
The Plant Pathology 2021: FGCV8[14] dataset is available on Kaggle and it contains images of apple leaves that are affected by four different diseases, namely Apple rust, Scab, Cedar apple rust, and Healthy. The dataset consists of a total of 18632 images with a resolution of 2048x1365 pixels. The images were captured from different orchards around the world, and the dataset was released in 2021 for the objective of developing and testing machine learning models that can accurately detect and classify the diseases present in apple plants. The dataset is well-organized and labeled, making it suitable for supervised learning approaches. It is a valuable resource for researchers and practitioners working in the field of plant pathology and computer vision. The Plant Pathology 2021: FGCV8[14] dataset is distributed as a set of image files, with each image representing a leaf from an apple plant that is either healthy or affected by one of the following diseases: Apple rust, Scab, or Cedar apple rust. The dataset is well-balanced, with approximately 25% of the images representing each of the four categories, totaling 18632 images in total. This balance in the dataset ensures that machine learning models trained on this data are less likely to be biased towards any particular class, which is important for ensuring accurate disease detection in apple plants.

The distribution of these categories in the dataset is as follows:

- Healthy: 4826 images (25.9%)
- Complex: 821 images (4.4%)
- Rust: 6226 images (33.4%)
- Powdery Mildew: 4759 images (25.5%)

The sample images of Plant Pathology 2021-FGCV8[14] dataset are given in Fig. 5.2.

Some leaves may have more than one disease or condition, so they may belong to more than one label. The dataset has more images of some classes than others,

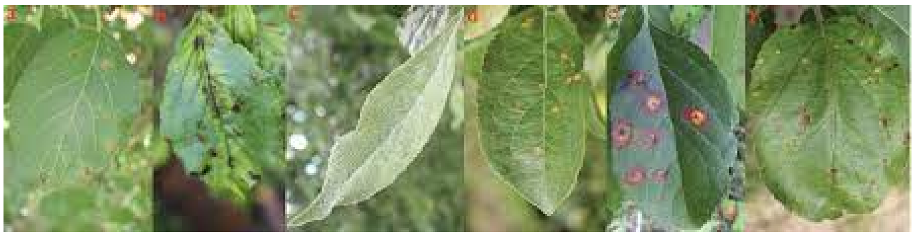


Figure 5.2: Sample Images of Plant Pathology 2021-FGCV8 Dataset[14]

with the “Complex” class having the fewest images. This can make it hard to train machine learning models that are good at all classes, and researchers and developers may have to use methods such as data augmentation and class balancing to make the model better.

Chapter 6

EXPERIMENTS AND RESULTS

In this chapter, the effectiveness and results of the suggested model has been discussed. The suggested model is evaluated using data from two datasets. The Data Evaluation section contains information on these datasets in depth. In this study, the suggested model was trained using four optimizers, and the outcomes were compared. Adam, NAG, RMS Prop and RMS Prop with Nesterov momentum were the four optimizers employed. The Chapter: Proposed Work provides an explanation of these optimizers. Each optimizer has a learning rate[57] setting of 0.01 for learning. further contrasted the outcomes with two previously suggested CNN based algorithms.

6.1 Performance Metrics

Accuracy, Precision, and Recall are the three performance measures utilized in this case to compare the outcomes. These metrics are employed to assess the effectiveness of classification algorithms, particularly CNN-based models for the identification of PDs[1].

1. **Accuracy:** The degree to which a measurement, computation, or forecast is accurate or precise is referred to as accuracy. The number of accurate forecasts or measurements divided by the overall count of accurate predictions or measurements is often stated as a percentage or ratio. In other words, accuracy assesses a model's or system's performance in terms of the accuracy with which it can recognise or categorize data. Accuracy represents the proportion of accurate predictions made by an algorithm among all the predictions it has made. It can be calculated by dividing the number of true positives (TP) and true negatives (TN) by the total number of instances (TP + TN + FP + FN), where FP is false positives and FN is false negatives1. Accuracy measures how well a model can classify all instances correctly, regardless of their class. Let's say a company is trying to predict which job candidates will be successful in their role. They use a test to evaluate candidates' skills, and they use the results of the test to make their hiring decisions. The company hires 100 candidates based on their test results. After six months on the job, the company evaluates how well each employee is performing and categorizes them as either successful or not successful based on predetermined criteria. If the company correctly identified 80 out of the 100 successful candidates using the test, and correctly identified 10 out of the 100 unsuccessful candidates, then the accuracy of their test is:
   Accuracy = (number of correct predictions) / (total number of predictions)
   Accuracy = (80 + 90) / 200

Accuracy = 0.85 or 85%

This means that the test had an accuracy of 85%, meaning that it correctly identified 85% of the candidates who would be successful on the job, and incorrectly identified 15% of the candidates who would not be successful on the job.

$$Accuracy = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{6.1}$$

2. **Precision:** Precision is the fraction of relevant instances among all retrieved in stances2. It can be calculated by dividing the number of true positives (TP) by the number of predicted positives (TP + FP). Precision measures how well a model can avoid false positives, or how accurate its positive predictions are. Let's say a doctor wants to determine whether a patient has a certain disease, and uses a diagnostic test to make the determination. The test results can be positive or negative.The doctor performs the test on 100 patients, and the test results show 40 positive results and 60 negative results. The doctor knows from previous experience that the true prevalence of the disease in the patient population is 20%.The doctor is interested in the precision of the test, which refers to the proportion of positive test results that are truly positive (i.e., the proportion of true positives among all positive test results).If the doctor examines the 40 patients who tested positive and finds that 30 of them truly have the disease, while 10 do not, then the precision of the test is:
   Precision = 30 / (30 + 10)
   Precision = 0.75 or 75%

   This means that the precision of the test is 75%, meaning that 75% of the patients who tested positive actually have the disease, while 25% of the positive results were false positives.

$$Precision = \frac{\text{Number of Correctly Predicted Positive Instances}}{\text{Number of Total Positive Positive Predictions you made}} \tag{6.2}$$

3. **Recall:** Recall is the fraction of relevant instances that were retrieved/2. It can be calculated by dividing the number of true positives (TP) by the number of actual positives (TP + FN)2. Recall measures how well a model can capture positive cases, or how sensitive it is to positive instances. Let's say a company wants to predict which customers are likely to churn (i.e., stop using their services). They use a machine learning model to make these predictions, which outputs a score for each customer indicating their likelihood of churning. The company has a total of 1,000 customers, of which 200 have already churned. The machine learning model predicts that 300 customers are likely to churn in the future. The company is interested in the recall of the model, which refers to the proportion of actual churners that are correctly identified by the model (i.e., the proportion of true positives among all actual positives). If the model correctly predicts 150 out of the 200 customers who have already churned, then the recall of the model is:
   Recall = 150 / (150 + 50)
   Recall = 0.75 or 75

   This means that the recall of the model is 75%, meaning that the model correctly identified 75% of the customers who actually churned, while 25% of the actual churners were not identified by the model.

$$Recall = \frac{\text{Number of Correctly Predicted Positive Instances}}{\text{Number of Total Positive Positive Instances in a Dataset}} \tag{6.3}$$

In the context of PD[1] precision, accuracy, and recall can be utilized to determine the capability of machine learning models that are used to detect and diagnose PDs[1] based on visual symptoms. Accuracy refers to how often the model correctly identifies whether a plant is diseased or not, based on its symptoms. This can be calculated by comparing the model's predictions with the true values(i.e., the actual disease status of the plant), which can be determined by laboratory testing or visual inspection by a human expert. High accuracy indicates that the model is able to correctly identify diseased plants, as well as healthy plants, with a high degree of accuracy. Precision refers to how often the model correctly identifies diseased plants, out of all the plants that it predicted to be diseased. This can be important in the context of PD[1] management, as false positives (i.e., plants that are incorrectly identified as diseased) can lead to unnecessary treatments and costs. High precision indicates that the model is able to accurately identify diseased plants, with few false positives. Recall refers to how often the model correctly identifies diseased plants, out of all the plants that are actually diseased. This can be important in the context of plant disease management, as false negatives (i.e., plants that are incorrectly identified as healthy) can lead to untreated plants and continued spread of the disease. High recall indicates that the model is able to accurately identify most of the diseased plants, with few false negatives. Overall, high accuracy, precision, and recall are desirable in a plant disease diagnosis model, as they indicate that the model is able to correctly identify diseased plants, with few false positives or false negatives. This can help to ensure effective disease management and reduce the spread of PDs[1].

6.2 Analysis and Visualization of the Experimental Result

According to the details provided, the suggested technique has been tested on two different datasets to evaluate its performance using Accuracy, Precision, and Recall as evaluation metrics. Additionally, the performance of the proposed model has been compared using four optimizers, namely Adam, Nesterov Accelerated Gradient, RMS Prop, and RMSprop with Nesterov Momentum. The purpose of using these evaluation metrics is to ascertain the reliability of the suggested model in anticipating the correct output for a given input. Accuracy refers to the possibility of correct estimation made by the model, while precision and recall are utilized to assess the productivity of the model in identifying the true positive and false positive cases. By comparing the proficiency of the proposed model using different optimizers, it can be determined which optimizer is more effective in improving the performance of the model. Adam is a popular optimizer that is known to perform well on a wide range of DL[12] tasks. NAG is another optimizer that is effective in accelerating the convergence of DNNs[46], particularly in cases where the optimization landscape is highly non-convex or contains a large number of local minima. RMSProp is designed to overcome the limitations of the Adagrad optimizer, which can lead to a decaying learning rate over time and slow convergence. In RMSProp, the gradient is scaled by a running average of the squared gradient, which helps to adjust the learning rate based on the magnitude of the gradient. RMSprop with Nesterov Momentum is a variant of the RMSprop optimizer that incorporates the NAG algorithm for momentum updates. This optimizer combines the benefits of both RMSprop and NAG to accelerate convergence and improve the robustness of the optimization process.

Overall, the choice of optimizer depends on the precise characteristics of the data repository and the model architecture. Therefore, it is significant to analyze with separate optimizers to find the one that works best for a given task.

6.2.1 Performance on Plant Pathology 2020- FCVG7 dataset

1. **Adam Optimizer:** The model is practiced on the dataset for 50 epochs using the Adam optimizer. This model obtained the maximum accuracy of 0.8656, precision of 0.8991, recall of 0.8329, and the lowest loss of 0.3443. These performance metrics are for the course. This can be seen in the Fig. 6.1.

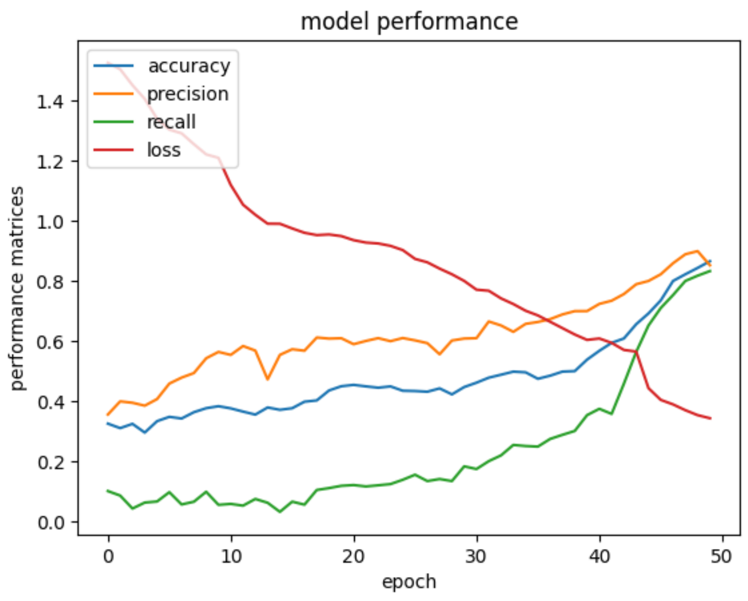


Figure 6.1: Performance Evaluation of the Proposed model using Adam Optimizer on Plant Pathology 2020-FCVG7 Dataset[13]

2. **Nesterov Accelerated Gradient Optimizer:**
The model is practiced on the dataset for 50 epochs using the Nesterov Accelerated Gradient Optimizer. This model achieved the highest accuracy of 0.8537, precision of 0.8871, recall of 0.8169, and the lowest loss of 0.4186. These performance metrics are for the course. This can be seen in the Fig. 6.2.

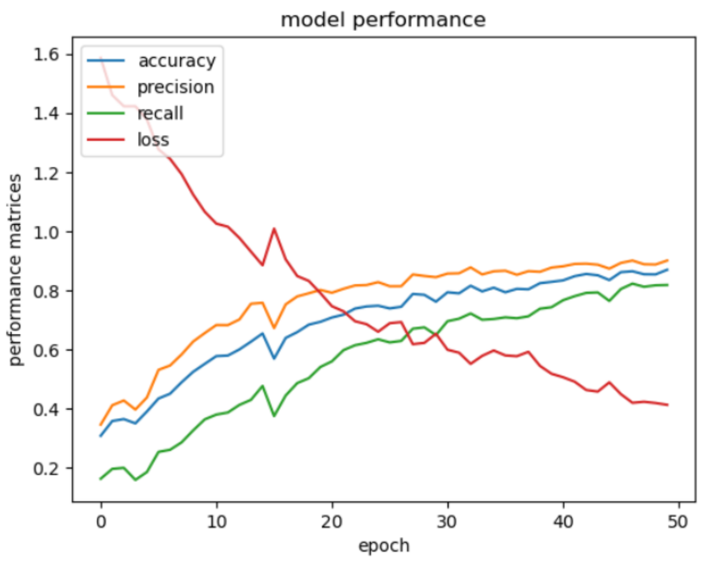


Figure 6.2: Performance Evaluation of the Proposed model using Nesterov Accelerated Gradient Optimizer on Plant Pathology 2020-FCVG7 Dataset[13]

3. **RMS Prop Optimizer:** The model is practiced on the dataset for 50 epochs using the RMS Prop Optimizer. This model achieved the highest accuracy of 0.6261, precision of 0.7275, recall of 0.4591, and the lowest loss of 0.9249. These performance metrics are for the course. This can be seen in the Fig. 6.3.

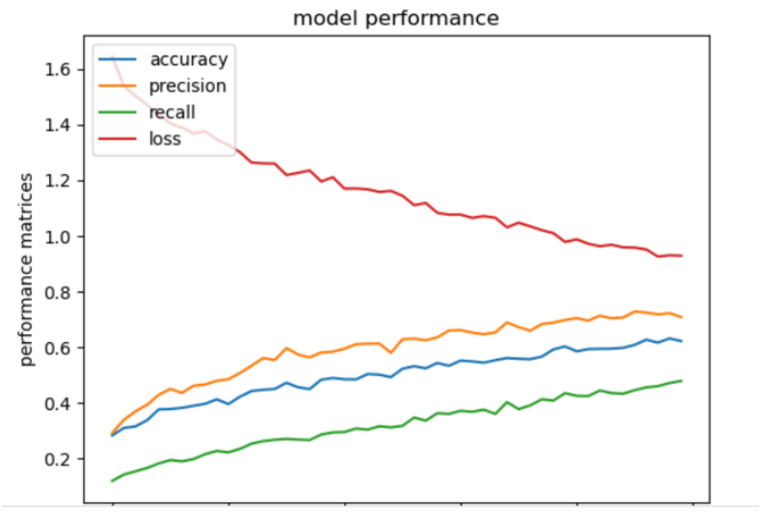


Figure 6.3: Performance Evaluation of the Proposed model using RMS Prop Optimizer on Plant Pathology 2020-FCVG7 Dataset[13]

4. **RMSprop with Nesterov Momentum Optimizer:** The model is practiced on the dataset for 50 epochs using the RMSprop with Nesterov Momentum Optimizer. This model achieved the highest accuracy of 0.9082, precision of 0.9234, recall of 0.8885, and the lowest loss of 0.2758. These performance metrics are for the course. This can be seen in the Fig. 6.4.

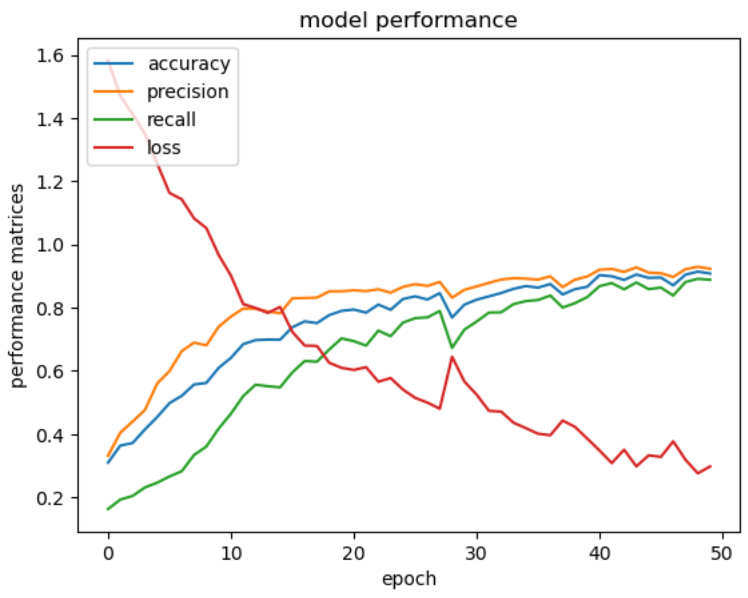


Figure 6.4: Performance Evaluation of the Proposed model using RMSprop with Nesterov Momentum Optimizer on Plant Pathology 2020-FCVG7 Dataset[13]

6.2.2 Performance on Plant Pathology 2021- FCVG8 dataset

1. **Adam Optimizer:** The model is practiced on the dataset for 50 epochs using the Adam optimizer. This model accomplished the maximum accuracy of 0.8996, precision of 0.9023, recall of 0.8729, and the lowest loss of 0.0803. These performance metrics are for the course. This can be seen in the Fig. 6.5.

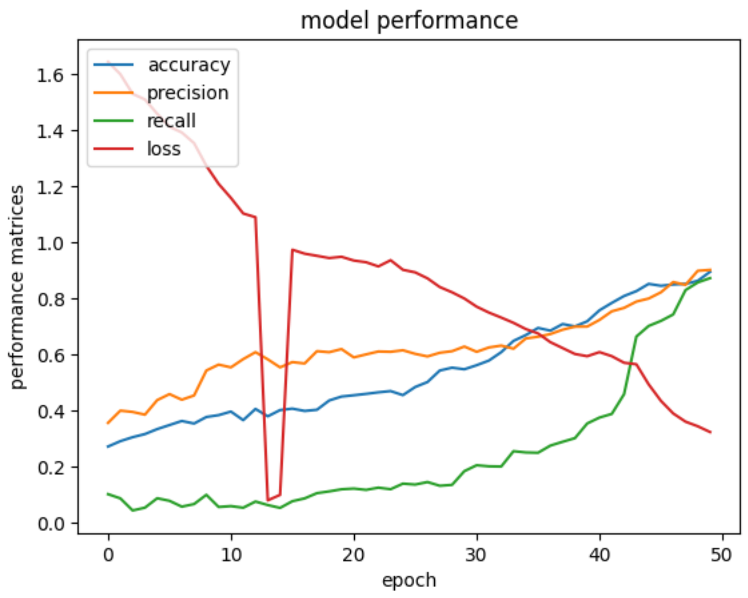


Figure 6.5: Performance Evaluation of the Proposed model using Adam Optimizer on Plant Pathology 2021-FCVG8 Dataset[14]

2. **Nesterov Accelerated Gradient Optimizer:**

The model is practiced on the dataset for 50 epochs using the Nesterov Accelerated Gradient Optimizer. This model achieved the highest accuracy of 0.8656, precision of 0.9134, recall of 0.8729, and the lowest loss of 0.3502. These performance metrics are for the course. This can be seen in the Fig. 6.6.

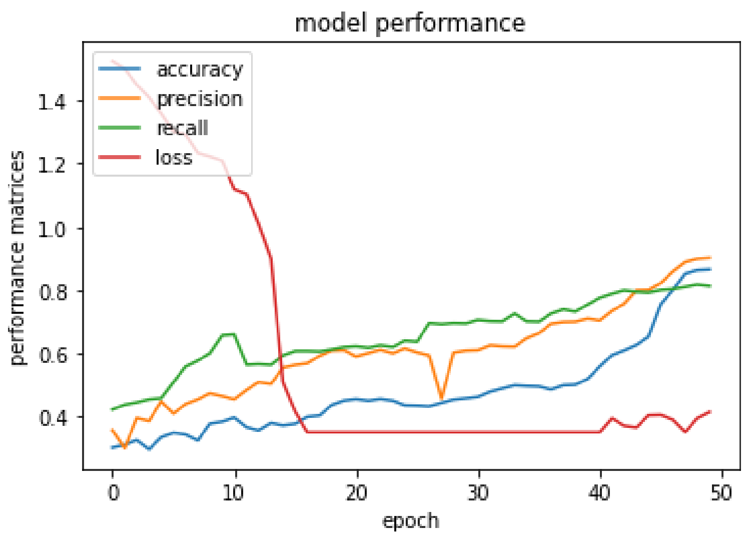


Figure 6.6: Performance Evaluation of the Proposed model using Nesterov Accelerated Gradient Optimizer on Plant Pathology 2021-FCVG8 Dataset[14]

3. **RMS Prop Optimizer:**
The model is practiced on the dataset for 50 epochs using the RMS Prop Optimizer. This model achieved the highest accuracy of 0.6356, precision of 0.7123, recall of 0.5029, and the lowest loss of 0.645. These performance metrics are for the course. This can be seen in the Fig. 6.7.

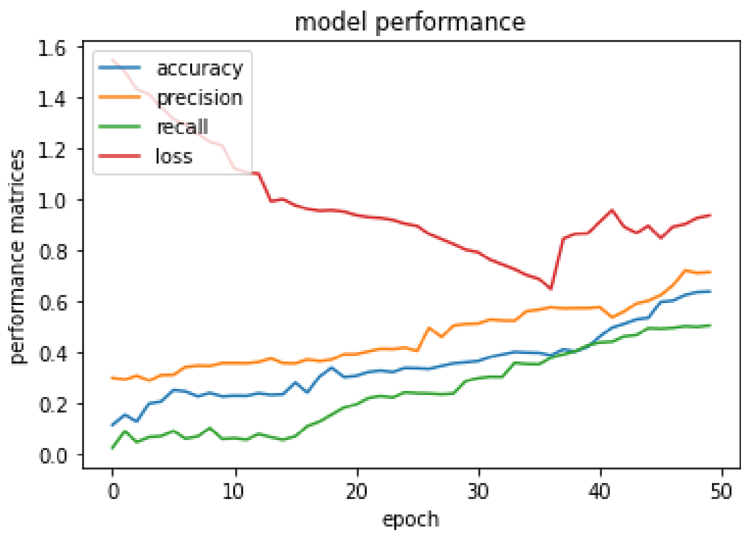


Figure 6.7: Performance Evaluation of the Proposed model using RMS Prop Optimizer on Plant Pathology 2021-FCVG8 Dataset[14]

4. **RMSprop with Nesterov Momentum Optimizer:**
   The model is practiced on the dataset for 50 epochs using the RMSprop with Nesterov Momentum Optimizer. This model achieved the highest accuracy of 0.9256, precision of 0.9523, recall of 0.9029, and the lowest loss of 0.2543. These performance metrics are for the course. This can be seen in the Fig. 6.8.

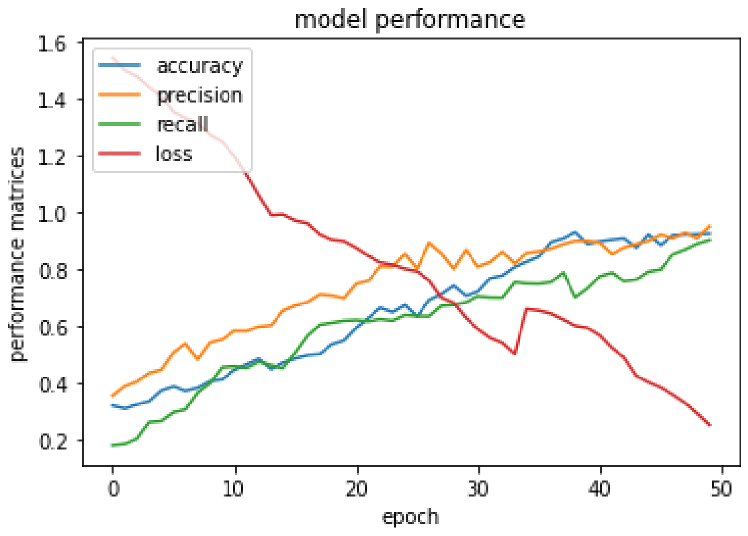


Figure 6.8: Performance Evaluation of the Proposed model using RMSprop with Nesterov Momentum Optimizer on Plant Pathology 2021-FCVG8 Dataset[14]

6.2.3 Summary of the Results

Summary for Plant Pathology 2020- FCVG7 dataset

Optimizers	**Accuracy**	**Precision**	**Recall**
Adam (in %)	86.56%	89.91%	83.29%
Nesterov Accelerated Gradient (in %)	85.37%	88.71%	81.69%
RMS Prop (in %)	**62.61%**	**72.75%**	**45.91%**
RMSprop with Nesterov Momentum (in %)	**90.82%**	**92.34%**	**88.85%**

Table 6.1: Performance Evaluation of Plant Pathology 2020: FGCV7 dataset[13]

In Table 6.1, It is shown that RMSprop with Nesterov Momentum optimizer gives better results with a loss of 0.2758 and RMSprop optimizer gives worst results with a loss of 0.9249.

Summary for Plant Pathology 2021- FCVG8 dataset

In Table 6.2, It is shown that RMSprop with Nesterov Momentum optimizer gives better results with a loss of 0.2543 and RMSprop optimizer gives worst results with a loss of 0.645.
Based on the information presented in Tables 6.1 and 6.2, it can be deduced that the proposed model performs better on the Plant Pathology 2021-FGCV8[14] dataset compared to the Plant Pathology 2020-FGCV7[13] dataset. Furthermore, the proposed model

Optimizers	**Accuracy**	**Precision**	**Recall**
Adam (in %)	89.96%	90.23%	87.29%
Nesterov Accelerated Gradient (in %)	86.56%	91.34%	81.76%
RMS Prop (in %)	**63.56%**	**71.23%**	**50.29%**
RMSprop with Nesterov Momentum (in %)	**92.56%**	**95.23%**	**90.29%**

Table 6.2: Performance Evaluation of Plant Pathology 2021: FGCV8 dataset[14]

was trained using RMSprop in conjunction with the Nesterov Momentum Optimizer on both datasets. To delve deeper into the findings, a comparison was made between the proposed model and two pre-existing CNN-based models, namely Model1[58] and Model2[16]. Both of these models were trained using RMSprop and the Nesterov Momentum Optimizer, specifically on the Plant Pathology 2021-FGCV8 dataset[14]. Results are given in Fig. 6.9.

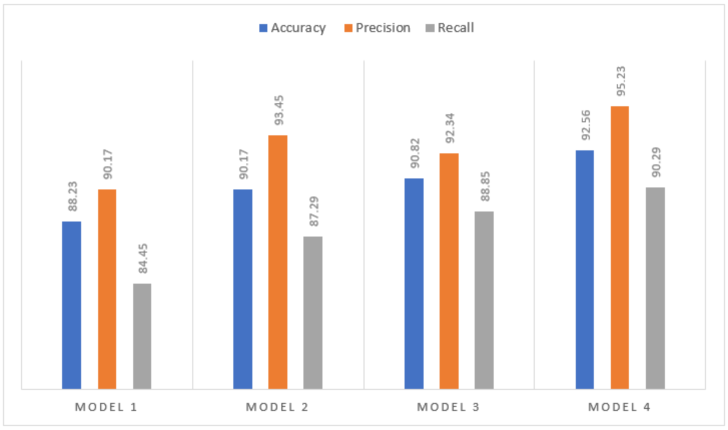


Figure 6.9: Comparison of Performance using RMSProp with Nesterov Momentum Optimizer

As depicted in Figure 6.9, three models were evaluated: Model1[58] and Model2[16], which are CNN-based models, and Model3, which is a multi-layer CNN model trained on the Plant Pathology 2020-FGCV7[13] dataset. Additionally, Model4 is a proposed multi-layer CNN model trained on the Plant Pathology 2021-FGCV8[14] dataset. All models were trained using an RMSprop and the Nesterov Momentum optimizer. Based on the conclusions, it can be concluded that Model4 surpasses the other models, obtaining a precision of 95.23%, a recall of 90.29% and accuracy of 92.56%.

Chapter 7

CONCLUSION & FUTURE SCOPE

CNNs have already shown great promise in the field of PDD and diagnosis. However, there are several potential areas for future development and application of CNNs in this field. Some of these include:

1. More accurate and robust models: CNNs have shown high accuracy in detecting PDs from visual symptoms, however there is still potential for enhancement regarding accuracy, robustness, and generalization to new diseases and plant species.

2. Integration with other data sources: While visual symptoms are a good indicator of PD, they can be influenced by other factors such as environmental conditions, nutrient deficiencies, and pest damage. By integrating visual data with other sources of information such as genetic data, environmental data, and agronomic data, more accurate and robust models can be developed.

3. Real-time disease detection: With the use of sensors and cameras in the field, real-time PDD is becoming more feasible. CNN models can be trained to work with real-time data and provide instant feedback to farmers and agronomists, allowing for timely interventions and disease management.

4. Mobile and web-based applications: Mobile and web-based applications that integrate CNN models for plant disease detection can provide easy access to farmers and growers, even in remote areas with limited resources. Such applications can provide real-time information on disease prevalence, management strategies, and even predict disease outbreaks.

Overall, CNNs hold great potential for the future of PDD and management. With continued development and research, these models can perform a key function in enhancing crop productivity and reducing the spread of PDs.

In this study, test has been conducted to see how well the Multilayer CNN model can recognize PDs. The model performed very well in distinguishing between healthy and sick plants, and also in naming the disease that each plant had. The model learned from a big collection of plant pictures, which helped it to identify the features and patterns that are related to different PDs. Different measures such as accuracy, precision and recall have been used to assess how well the model did, and the results showed that the model was very accurate in all measures. The Multilayer CNN model's capacity to learn properties straight from the picture data reduces the need for manually created features and is one of its primary advantages. Also, the analysis of complicated data sets with high-dimensional inputs, such as plant photos, is made possible by the application of DL techniques, such as CNNs. The technology may also be used to detect plant illnesses in real time, enabling farmers to take preventative action before the disease spreads.

Although Multilayer CNN models have shown promising results for PDD, there are some limitations to their use. Some of these limitations include:

1. **Computational requirements:** Computational requirements: Multilayer CNN models are typically deeper and more complex than traditional CNNs, which can lead to higher computational requirements for training and inference. This can make it more difficult to deploy Multilayer CNN models in resource-limited environments or on low-powered devices.

2. **Overfitting:** Multilayer CNN models may be susceptible to overfitting, specially when the dataset is minuscule or imbalanced. Overfitting can occur when the model learns to memorize the training data instead of generalizing to new data, leading to poor performance on unseen data.

3. **Dataset bias:** The performance of any machine learning model, including Multilayer CNN models, is highly dependent on the quality and diversity of the training dataset. If the dataset is biased or incomplete, the model may not generalize well to new data, leading to poor performance in real-world scenarios.

4. **Limited interpretability:** DL models, including Multilayer CNN models, can be difficult to interpret and understand, especially when it comes to how the model makes its decisions. This can make it difficult to identify and correct errors or biases in the model.

5. **Limited scope:** Multilayer CNN models are designed specifically for image-based classification tasks, such as PDD. While they may be highly effective for this specific task, they may not be applicable to other types of data or problems.

In summary, although Multilayer CNN models have shown promising results for PDD, there is still a significant potential for future research to further enhance their accuracy and efficiency. The agricultural industry can benefit greatly from the development of these models, as they can reduce the risk of crop losses and enhance food security. Future research can explore various avenues such as improving the quality of the dataset, incorporating transfer learning, developing mobile applications, integrating multiple modalities, and enhancing the explainability and interpretability of the models. These efforts can pave the way for more effective and sustainable disease detection solutions for the agricultural sector.

Bibliography

[1] "plant disease - Symptoms and signs — Britannica." https://www.britannica.com/science/plant-disease/Symptoms-and-signs (accessed May 25, 2022).

[2] J. Mccarthy, "WHAT IS ARTIFICIAL INTELLIGENCE?," 2007, Accessed: Mar. 31, 2023. [Online]. Available: http://www-formal.stanford.edu/jmc/

[3] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science (1979), vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/SCIENCE.AAA8415.

[4] D. A. Pisner and D. M. Schnyer, Support vector machine. Elsevier Inc., 2019. doi: 10.1016/B978-0-12-815739-8.00006-7.

[5] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," Appl Stat, vol. 28, no. 1, p. 100, 1979, doi: 10.2307/2346830.

[6] "[PDF] The Plant Pathology 2020 challenge dataset to classify foliar disease of apples — Semantic Scholar." https://www.semanticscholar.org/paper/The-Plant-Pathology-2020-challenge-dataset-to-of-Thapa-Snavely/17005a1bd4189707f17d8bef9a0909c9399f7171 (accessed Mar. 18, 2023).

[7] "Fire Blight Treatment: How To Recognize Fire Blight Symptoms." https://www.gardeningknowhow.com/plant-problems/disease/fire-blight-remedies-and-symptoms.htm (accessed May 17, 2023).

[8] "Cedar Apple Rust — NC State Extension Publications." https://content.ces.ncsu.edu/cedar-apple-rusts (accessed May 17, 2023).

[9] "Cancer-fighting drugs also help plants fight disease - Science & research news — Frontiers." https://blog.frontiersin.org/2018/10/17/plant-science-cancer-drugs-pathogens-disease/?amp=1 (accessed May 17, 2023).

[10] L. Medsker, D. L. Jain, and B. Raton London New York Washington, "RECURRENT NEURAL NETWORKS Edited by Design and Applications," 2001.

[11] G. E. Hinton, "Deep belief networks," Scholarpedia, vol. 4, no. 5, p. 5947, 2009, doi: 10.4249/SCHOLARPEDIA.5947.

[12] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553. Nature Publishing Group, pp. 436–444, May 27, 2015. doi: 10.1038/nature14539.

[13] "Plant Pathology 2020 - FGVC7 — Kaggle." https://www.kaggle.com/c/plant-pathology-2020-fgvc7/overview (accessed Mar. 31, 2023).

[14] "Plant Pathology 2021 - FGVC8 — Kaggle." https://www.kaggle.com/c/plant-pathology-2021-fgvc8 (accessed Dec. 14, 2022).

[15] "Apple: Diseases and Symptoms — Vikaspedia." https://vikaspedia.in/agriculture/crop-production/integrated-pest-managment/ipm-for-fruit-crops/ipm-strategies-for-apple/apple-diseases-and-symptoms (accessed May 11, 2023).

[16] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, "Real-Time Detection of Apple Leaf Diseases Using Deep Learning Approach Based on Improved Convolutional Neural Networks," IEEE Access, vol. 7, pp. 59069–59080, 2019, doi: 10.1109/ACCESS.2019.2914929.

[17] "Deep Learning: GoogLeNet Explained — by Richmond Alake — Towards Data Science." https://towardsdatascience.com/deep-learning-googlenet-explained-de8861c82765 (accessed May 26, 2022).

[18] S. Panigrahi, A. Nanda, and T. Swarnkar, "A Survey on Transfer Learning," Smart Innovation, Systems and Technologies, vol. 194, no. 10, pp. 781–789, 2021, doi: 10.1007/978-981-15-5971-6_83.

[19] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10. pp. 1345–1359, 2010. doi: 10.1109/TKDE.2009.191.

[20] G. Geetharamani and A. P. J., "Identification of plant leaf diseases using a nine-layer deep convolutional neural network," Computers and Electrical Engineering, vol. 76, pp. 323–338, Jun. 2019, doi: 10.1016/j.compeleceng.2019.04.011.

[21] "GitHub - spMohanty/PlantVillage-Dataset: Dataset of diseased plant leaf images and corresponding labels." https://github.com/spMohanty/PlantVillage-Dataset (accessed May 25, 2022).

[22] R. G. De Luna, E. P. Dadios, and A. A. Bandala, "Automated Image Capturing System for Deep Learning-based Tomato Plant Leaf Disease Detection and Recognition," IEEE Region 10 Annual International Conference, Proceedings/TENCON, vol. 2018-October, pp. 1414–1419, Feb. 2019, doi: 10.1109/TENCON.2018.8650088.

[23] A. Adedoja, P. A. Owolawi, and T. Mapayi, "Deep learning based on NASNet for plant disease recognition using leave images," icABCD 2019 - 2nd International Conference on Advances in Big Data, Computing and Data Communication Systems, Aug. 2019, doi: 10.1109/ICABCD.2019.8851029.

[24] "Review: NASNet — Neural Architecture Search Network (Image Classification) — by Sik-Ho Tsang — Medium." https://sh-tsang.medium.com/review-nasnet-neural-architecture-search-network-image-classification-23139ea0425d (accessed Apr. 06, 2023).

[25] V. Tiwari, R. C. Joshi, and M. K. Dutta, “Dense convolutional neural networks based multiclass plant disease detection and classification using leaf images,” Ecol Inform, vol. 63, no. March, p. 101289, 2021, doi: 10.1016/j.ecoinf.2021.101289.

[26] F. Jiang, Y. Lu, Y. Chen, D. Cai, and G. Li, “Image recognition of four rice leaf diseases based on deep learning and support vector machine,” Comput Electron Agric, vol. 179, Dec. 2020, doi: 10.1016/J.COMPAG.2020.105824.

[27] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, “Using deep transfer learning for image-based plant disease identification,” Comput Electron Agric, vol. 173, p. 105393, Jun. 2020, doi: 10.1016/J.COMPAG.2020.105393.

[28] M. Agarwal, A. Singh, S. Arjaria, A. Sinha, and S. Gupta, “ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network,” Procedia Comput Sci, vol. 167, no. 2019, pp. 293–301, 2020, doi: 10.1016/j.procs.2020.03.225.

[29] U. P. Singh, S. S. Chouhan, S. Jain, and S. Jain, “Multilayer Convolution Neural Network for the Classification of Mango Leaves Infected by Anthracnose Disease,” IEEE Access, vol. 7, pp. 43721–43729, 2019, doi: 10.1109/ACCESS.2019.2907383.

[30] A. Smetanin, A. Uzhinskiy, G. Ososkov, P. Goncharov, and A. Nechaevskiy, “Deep learning methods for the plant disease detection platform,” AIP Conf Proc, vol. 2377, no. October, 2021, doi: 10.1063/5.0068797.

[31] P. Bedi and P. Gole, “Plant disease detection using hybrid model based on convolutional autoencoder and convolutional neural network,” Artificial Intelligence in Agriculture, vol. 5, pp. 90–101, 2021, doi: 10.1016/j.aiia.2021.05.002.

[32] Y. Zhang, “A Better Autoencoder for Image: Convolutional Autoencoder”.

[33] M. Brahimi, M. Arsenovic, S. Laraba, S. Sladojevic, K. Boukhalfa, and A. Moussaoui, “Deep Learning for Plant Diseases: Detection and Saliency Map Visualisation,” no. June, pp. 93–117, 2018, doi: 10.1007/978-3-319-90403-0_6.

[34] M. Z. Alom et al., “The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches,” 2018, [Online]. Available: http://arxiv.org/abs/1803.01164

[35] G. Sachdeva, P. Singh, and P. Kaur, “Plant leaf disease classification using deep Convolutional neural network with Bayesian learning,” Mater Today Proc, vol. 45, pp. 5584–5590, 2021, doi: 10.1016/j.matpr.2021.02.312.

[36] Read, “Bayesian Learning”.

[37] V. K. Vishnoi, K. Kumar, B. Kumar, S. Mohan, and A. A. Khan, “Detection of Apple Plant Diseases Using Leaf Images Through Convolutional Neural Network,” IEEE Access, vol. 11, pp. 6594–6609, 2023, doi: 10.1109/ACCESS.2022.3232917.

[38] H. Amin, A. Darwish, A. E. Hassanien, and M. Soliman, “End-to-End Deep Learning Model for Corn Leaf Disease Classification,” IEEE Access, vol. 10, pp. 31103–31115, 2022, doi: 10.1109/ACCESS.2022.3159678.

[39] C. K. Sunil, C. D. Jaidhar, and N. Patil, “Cardamom Plant Disease Detection Approach Using EfficientNetV2,” IEEE Access, vol. 10, pp. 789–804, 2022, doi: 10.1109/ACCESS.2021.3138920.

[40] M. H. Saleem, J. Potgieter, and K. M. Arif, “A Performance-Optimized Deep Learning-Based Plant Disease Detection Approach for Horticultural Crops of New Zealand,” IEEE Access, vol. 10, pp. 89798–89822, 2022, doi: 10.1109/ACCESS.2022.3201104.

[41] A. V. Panchal, S. C. Patel, K. Bagyalakshmi, P. Kumar, I. R. Khan, and M. Soni, “Image-based Plant Diseases Detection using Deep Learning,” Mater Today Proc, Aug. 2021, doi: 10.1016/J.MATPR.2021.07.281.

[42] A. O. Anim-Ayeko, C. Schillaci, and A. Lipani, “Automatic blight disease detection in potato (Solanum tuberosum L.) and tomato (Solanum lycopersicum, L. 1753) plants using deep learning,” Smart Agricultural Technology, vol. 4, p. 100178, Aug. 2023, doi: 10.1016/J.ATECH.2023.100178.

[43] A. S. Keceli, A. Kaya, C. Catal, and B. Tekinerdogan, “Deep learning-based multi-task prediction system for plant disease and species detection,” Ecol Inform, vol. 69, p. 101679, Jul. 2022, doi: 10.1016/J.ECOINF.2022.101679.

[44] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” Front Plant Sci, vol. 7, no. September, p. 1419, Sep. 2016, doi: 10.3389/FPLS.2016.01419/BIBTEX.

[45] Y. Kawasaki, H. Uga, S. Kagiwada, and H. Iyatomi, “Basic study of automated diagnosis of viral plant diseases using convolutional neural networks,” Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9475, pp. 638–645, 2015, doi: 10.1007/978-3-319-27863-6_59/COVER.

[46] C. M. Bishop, “Neural networks and their applications,” Review of Scientific Instruments, vol. 65, no. 6, p. 1803, Jun. 1998, doi: 10.1063/1.1144830.

[47] R. A. Jarvis, “A Perspective on Range Finding Techniques for Computer Vision,” IEEE Trans Pattern Anal Mach Intell, vol. PAMI-5, no. 2, pp. 122–139, 1983, doi: 10.1109/TPAMI.1983.4767365.

[48] K. R. Chowdhary, “Natural Language Processing,” Fundamentals of Artificial Intelligence, pp. 603–649, 2020, doi: 10.1007/978-81-322-3972-7_19.

[49] M. Halle and K. Stevens, “Speech Recognition: A Model and a Program for Research,” IRE Transactions on Information Theory, vol. 8, no. 2, pp. 155–159, 1962, doi: 10.1109/TIT.1962.1057686.

[50] “Python Tensorflow - tf.keras.layers.Conv2D() Function - GeeksforGeeks.” https://www.geeksforgeeks.org/python-tensorflow-tf-keras-layers-conv2d-function/ (accessed May 17, 2023).

[51] “Inception-v3 Explained — Papers With Code.” https://paperswithcode.com/method/inception-v3 (accessed Apr. 22, 2023).

[52] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” 31st AAAI Conference on Artificial Intelligence, AAAI 2017, pp. 4278–4284, Feb. 2016, doi: 10.1609/aaai.v31i1.11231.

[53] “The architecture of Inception-V3 model. — Download Scientific Diagram.” https://www.researchgate.net/figure/The-architecture-of-Inception-V3-model_fig5_349717475 (accessed Mar. 17, 2023).

[54] A. M. Fred Agarap, “Deep Learning using Rectified Linear Units (ReLU),” Mar. 2018, Accessed: Apr. 04, 2023. [Online]. Available: https://arxiv.org/abs/1803.08375v2

[55] C. Banerjee, T. Mukherjee, and E. Pasiliao, “An empirical study on generalizations of the RelU activation function,” ACMSE 2019 - Proceedings of the 2019 ACM Southeast Conference, pp. 164–167, Apr. 2019, doi: 10.1145/3299815.3314450.

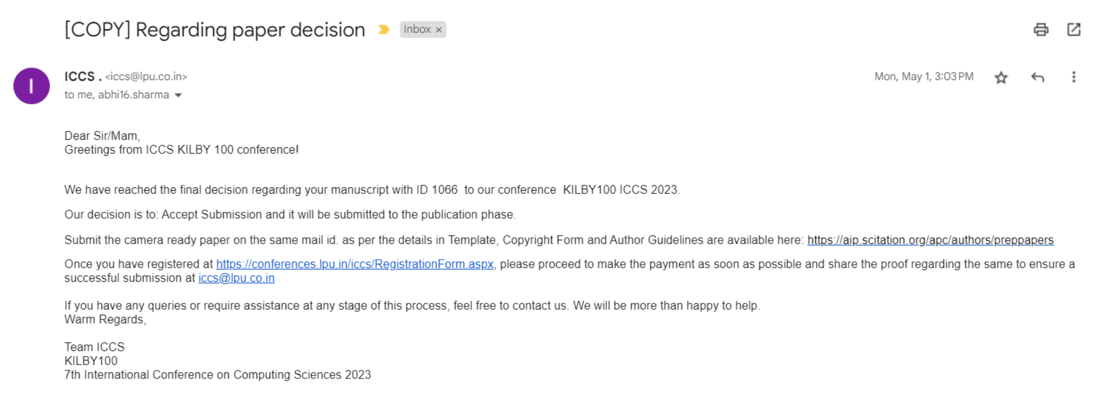
[56] M. Wang, S. Lu, D. Zhu, J. Lin, and Z. Wang, “A High-Speed and Low-Complexity Architecture for Softmax Function in Deep Learning,” 2018 IEEE Asia Pacific Conference on Circuits and Systems, APCCAS 2018, pp. 223–226, Jan. 2019, doi: 10.1109/APCCAS.2018.8605654.

[57] “Learning rate - Wikipedia.” https://en.wikipedia.org/wiki/Learning_rate (accessed Apr. 06, 2023).

[58] S. P. Singh, K. Pritamdas, K. J. Devi, and S. D. Devi, “Custom Convolutional Neural Network for Detection and Classification of Rice Plant Diseases,” Procedia Comput Sci, vol. 218, pp. 2026–2040, 2023, doi: 10.1016/j.procs.2023.01.179.

LIST OF PUBLICATIONS

1. Saradindu Das, Abhilasha Sharma, *"Convolutional Neural Network Based Plant Disease Detection: A Review"*. The paper has been **Accepted** at the 7th International Joint Conference On Computing Sciences (**ICCS-2023**), May 2023. Indexed by **Scopus.** Paper Id: 1066



[COPY] Regarding paper decision

ICCS . <iccs@lpu.co.in>
to me, abhi16.sharma

Mon, May 1, 3:03 PM

Dear Sir/Mam,
Greetings from ICCS KILBY 100 conference!

We have reached the final decision regarding your manuscript with ID 1066 to our conference KILBY100 ICCS 2023.

Our decision is to: Accept Submission and it will be submitted to the publication phase.

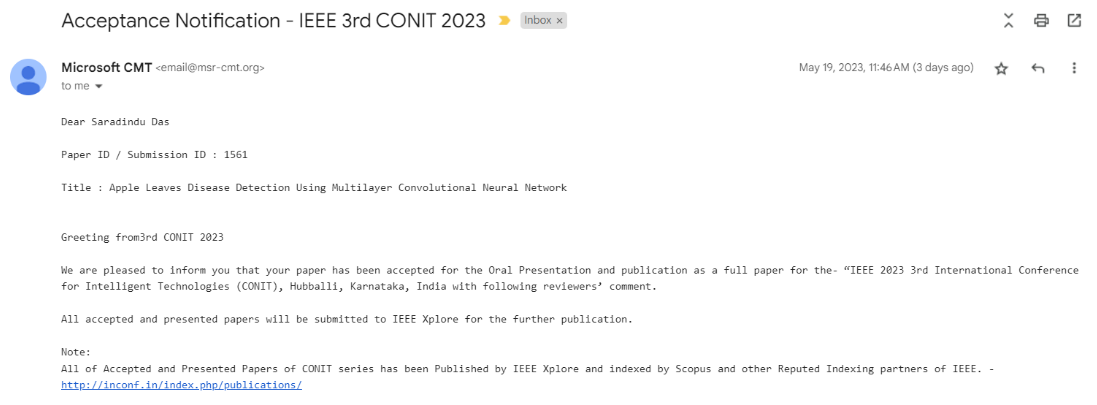
Submit the camera ready paper on the same mail id. as per the details in Template, Copyright Form and Author Guidelines are available here: https://aip.scitation.org/apc/authors/preppapers

Once you have registered at https://conferences.lpu.in/iccs/RegistrationForm.aspx, please proceed to make the payment as soon as possible and share the proof regarding the same to ensure a successful submission at iccs@lpu.co.in

If you have any queries or require assistance at any stage of this process, feel free to contact us. We will be more than happy to help.
Warm Regards,

Team ICCS
KILBY100
7th International Conference on Computing Sciences 2023

2. Saradindu Das, Abhilasha Sharma, *"Apple Leaves Disease Detection Using Multilayer Convolutional Neural Network"*. The paper has been **Accepted** at the International Conference on Intelligent Data Communication Technologies and Internet of Things (**CONIT-2023**), June 2023. Indexed by **Scopus.** Paper Id: 1561



Acceptance Notification - IEEE 3rd CONIT 2023

Microsoft CMT <email@msr-cmt.org>
to me

May 19, 2023, 11:46 AM (3 days ago)

```
Dear Saradindu Das

Paper ID / Submission ID : 1561

Title : Apple Leaves Disease Detection Using Multilayer Convolutional Neural Network


Greeting from3rd CONIT 2023

We are pleased to inform you that your paper has been accepted for the Oral Presentation and publication as a full paper for the- "IEEE 2023 3rd International Conference
for Intelligent Technologies (CONIT), Hubballi, Karnataka, India with following reviewers' comment.

All accepted and presented papers will be submitted to IEEE Xplore for the further publication.

Note:
All of Accepted and Presented Papers of CONIT series has been Published by IEEE Xplore and indexed by Scopus and other Reputed Indexing partners of IEEE. -
http://inconf.in/index.php/publications/
```