Computer Vision Based Approach for Detection of Disease in Cotton Plants

MSc Research Project
Masters of Science in Data Analytics

Steve Mendonca
Student ID: 20226691

School of Computing
National College of Ireland

Supervisor: Dr. Abid Yaqoob

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Steve Mendonca
Student ID:	20226691
Programme:	Masters of Science in Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Abid Yaqoob
Submission Due Date:	15/12/2022
Project Title:	Computer Vision Based Approach for Detection of Disease in Cotton Plants
Word Count:	5734
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	15th December 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Computer Vision Based Approach for Detection of Disease in Cotton Plants

Steve Mendonca
20226691

Abstract

Agriculture is one of the most important sectors in the well-being and survival of a country. This sector needs to be protected and catered for as it also increases the cash flow in the country. Cotton is one such important cash crop which needs to be taken care of. Sometimes fields of cotton are susceptible to diseases. These diseases can cause big problems as they spread among the plants rendering them useless. It is important to identify and take out these diseased plants early on before the damage is irreparable. This paper works around building a CNN model with multiple layers to identify and detect these diseased plants efficiently so that the concerned people could take action against them. The dataset used had 2310 images which was later augmented to avoid the problem of overfitting. The CNN model could identify the diseases with an accuracy of 96%. This model was compared with two pre-trained models, VGG16 and DenseNet121. The CNN model outperformed the two on certain important metrics showing that with further work done, it could be used on a higher scale and with better efficiency.

1 Introduction

Production and trade is one of the most important pillars towards sustaining a countries economy. The power to produce products which are high in demand could prove to be quite essential towards having a strong economy. Agriculture being one such important part, needs to be looked after and catered to, by the nations.

Agriculture could consist of a vast range of crops and plants that could be used. The crops that are used for trade and economical reasons are called as cash crops. These cash crops aid in increasing a country's financial status and build a stable economy. Cotton is one such cash crop which is considered to be a major source of income in many countries. Cotton is a crop that is easy to grow and can be grown in different weathers while it has a cycle that is similar to tomatoes. But with high yield of crops, the chances of them being vulnerable to diseases also increases. These diseases could destroy the yield and the farmer could face major losses. Through the chain of events, it could also affect the economy of a country that depends on cash crops. It is very important to pick out these diseased plants early on in the cycle so that they do not spread the disease and affect the other plants around them. Generally, this task is done manually using the help of experts who are good at identifying diseases. This process is quite tedious and takes an ample amount of time since the farmer has to go around each of the plant in a huge field and check for diseases in them. This is where technology and Machine Learning comes in. The vast field of Data Analytics has opened up many possibilities and ways to detect and classify images with respect to the features visible in the images. Many different researchers have worked towards building models and making life easier for the farmers. Recently, Deep Learning and Transfer Learning has taken the centre stage in classifying images which can be found to be very useful in the detection of diseases.

In this paper, a Convolutional Neural Network (CNN) model has been built to accurately detect and identify diseases in the leaves and stems of cotton plants. Transfer Learning has also been used to implement VGG16 and DenseNet121 architecture to maximize the output of the model and improve its performance. Various techniques and technologies were used to get the desirable output. The research question that is addressed in the paper is as follows:

- *How well does a custom made Convolutional Neural Network model work in comparison with various other models to detect diseases in cotton plants?*

This paper has focused on building a highly accurate model to with use of different technologies available, to get optimum results. The performance of the model is then compared with VGG16 and DenseNet121 to understand how well the model performs under similar constraints.

2 Related Work

In the last few years, many researchers have tried to build models and find solutions to help identify and detect diseases in plants early on. These papers are not only based on cotton, but on various types of plants giving it a wide variety of methods and technologies used. To move forward in the research it is important to go through work done previously on the same topic or topics similar to the research. This gives insight on what more could be done on the topic and which gaps could be improved in previous work. This section contains papers that covers different aspects of this paper and shows different work done in those areas. The sections divided are augmentation, work on machine learning models, work on transfer learning and work on customized CNN models. The papers are in chronological order for better understanding of the advancements made.

2.1 Studies related to Augmentation

In the research paper by (Kobayashi et al.; 2018), the authors have discussed how augmentation can be used to increase the accuracy and working of models. They used five different types of augmentation techniques like shear conversion, cutout, etc. They used Inception score and Frechet Inception distance to evaluate if the augmentation technique made the model better or no. The dataset used was a plant disease detection dataset which images of plants with diseases present. They concluded that the rotation method for augmentation gave the best outcome, but using only one type of augmentation would not let the model train for various other instances of images. The authors could run instances with multiple types of augmentation at once.

(Wongbongkotpaisan and Phumeechanya; 2021) used 6 different variations of augmentation that helped increase the accuracy of their models. The augmentation techniques used were image enhancement, geometric transformation, adding noise, image filtering, flips and rotations. They built a CNN model and also used transfer learning with VGG19 and Mobilenet. They used both local-based and global-based augmentation techniques to augment the data. This augmented data was then used to train these models giving accuracy of above 90% in all but one of them. This shows how augmentation could improve the working of the model.

2.2 Studies related to Machine Learning

Machine Learning being one of the ways to go about to build classifying models was used a lot before. (Sehgal and Mathur; 2019) used various Machine Learning models to detect disease in plants. These Machine learning models were trained and gave various outcomes with Support Vector Machine (SVM) being the best with 72% accuracy which is fairly low. The other models that were used were Random Forest, Naive Bayes and Decision Tree.

Gobalakrishnan et al. (2020) then did a review on the possible machine learning techniques regarding image processing and classification. They reviewed the techniques and work done by previous researchers and compared their work with each other. They came up with the result that compared to all machine learning models the CNN models were doing better with high accuracy. This pushes the inclination of machine learning models not doing as well in image classification.

In the paper by (Oh et al.; 2020), the research done was based in USA and the data collection was done by an unmanned drone. The authors used multiple regression models to identify Tar Spots on corn plants. They used Lins concordance correlation coefficient (CCC) to determine the best model out of the rest. Using PCA and nu-SVR they got the best model with a CCC value of 92%.

The work done by (Deepa et al.; 2021) shows how a SVM model is used to accurately classify if a leaf is infected or no. This was done by using feature extraction to hep the model train better and understand the images better. They used Gray co-occurrence matrix (GLCM) to do this feature extraction. Some of the features extracted using GLCM are contrast, correlation and homogeneity. After this the model was trained the and the model gave out pretty accurate results. The precise accuracy of the model was not defined and hence it would be difficult to gauge how well the model performed.

Finally in the research paper by (Alyas and Mohammed; 2022), they took a bit more complex approach towards identification of diseases in plants. They used pre-processing to increase the quality of the images after which they used segmentation. After this, they used K-means clustering which is used to create different clusters which helps in identification of the disease. This was then put in an OTSU classifier which uses these clusters to directly convert grey scale images to binary. This binary data is then put into the SVM classifier to obtain the necessary result. This entire process was a very novel and well thought process which gave them a great accuracy of 96.7%. Even though this process is good, it takes a lot of time and computational power.

2.3 Studies related to Transfer Learning

(Rubini and Kavitha; 2021) used transfer learning as a means to identify and detect diseases in plant images. The authors used VGG16 and DenseNet to identify the disease in plant images by first pre-processing and characterizing the images in a three dimensional vector. This helped get a form of segregation in the images and makes it easier to train the models. VGG16 came up with an accuracy of 92% while DenseNet came up with an accuracy of 98.25%. This showed that the DenseNet had a better grip over identifying diseased plants.

In the research later done by (Guan; 2021), the authors used 4 different transfer learning models, namely, Resnet, Inception, Inception Resnet and Densenet. The authors used a very large dataset to train the models which made the results of the models more efficient. The database consisted of 10 plant species and 61 classes that had more than 36000 images. The images were further augmented in three different images creating a large number of images to work with. They used the four models in a stacking method which increased the outcome of the model built giving an accuracy score of 87%. This brought out a good way of using transfer learning models.

In the work done by (Dutta and Rana; 2021), the authors have used various methods to come up with solutions to detect diseases. They used MobileNet V2 as the transfer learning model in their research. The authors applied various optimizers to check which one would be the ideal one to use while detecting disease in plants. AdaMax, Adagrad and Adadelta were some of the few optimizers that were used to test the model. Their research concluded that AdaMax gave a maximum accuracy of 98.51% which is very good performance by the model and this was then compared with other models done by other researchers. Their model stood out because the transfer learning model used was a very good one and the best possible optimizer was chosen.

Nagi and Tripathy (2021) used multiple pre-trained models to identify diseases in plants. AlexNet, VGG16, VGG19 and MobileNet were the pre-trained models that were used. SGD, RMSProp and Adam Optimizer were the three optimizers that were experimented with and these gave the result that the Adam Optimizer gave the best outcome of the three. MobileNet model came out on top with 98.53% accuracy but the models were trained on a dataset of only 3423 images. Since the number images are low, the models were under-trained and the results could not be totally trusted. Another point to be noted is that the data was split on a 70-30 ratio.

In the research paper by (Pajjuri et al.; 2022), the authors used four different pre-trained models, namely, AlexNet, VGG16, GoogleNet and ResNet50 V2. These models are state-of-the-art models. The four models are trained using three different datasets to test which of these models performs best in different disease detection scenarios. These three datasets are vast in size and train the model very well. These models gave out very good results. During the tests, it was noticed that VGG16 consistently did better than the remaining models. VGG16 being the most simple out of the rest gave out commendable results.

2.4 Studies related to Customized CNN models

Apart from using pre-trained models, building self-made customized models was one of the ways to go about to identify diseased plants. These models can be built and tuned depending on the requirement of the model. (Sardogan et al.; 2018) built a CNN model and used Learning Vector Quantization (LVQ) algorithm in their model. They used ReLu activation and MaxPooling after each layer and only one convolutional layer was used. This model worked on a dataset of only 500 images and the mode was under-trained. The accuracy of the model was 86% which is an average score given that the model is under-trained.

Militante et al. (2019) built a model with four layers consisting of convolutional layer, pooling layer, fully connected layer and the output layer. The authors pre-processed the images before training the models by reducing the sizes of the image to appropriate dimensions. This model was trained using a dataset of 35000 images. This trained the model better and the model had a train accuracy of 96.5%. The test accuracy was given to be 100% which should not be practically possible and hence the model could be overfitted.

In the research paper by (Sharma et al.; 2020), the authors have also built a four layered CNN model which is similar to the previous model that was seen. The model is trained with a dataset of 20000 images. This is a sufficient amount of data to train a model. This model was then compared with 4 other machine learning models and the CNN model outperformed the machine learning models by a great margin. The comparison made with the machine learning model would be unfair as they do not perform with image related data.

The work by (V et al.; 2021) gives insight on the 5 layered convolutional neural network model built to detect diseases in plants. 13 different types of plant images were used to train the model. The variation in the types of plants gave the model a better understanding of diseased leaves and how to identify them. The issue with the dataset that it was highly unbiased which in turn did not give the model a good grasp over identification. Some plant images were below 100 which was a significantly low number of images to train the model on. However, the accuracy of the model was shown to be 96.3% but not a lot of tuning was done.

(Lakshmanarao et al.; 2021), in their work not only built models to detect diseases in plants but drew comparisons with previously existing models built for the same datsets. The dataset chosen had three types of plants namely, potato, tomato and bell pepper and each of the plants had diseases classified. The dataset was clean and had a large number of images to train the model. To get a better understanding of the working of the model, the author divided the dataset into 3 parts. The CNN model used was a simple four layered model. The model gave out better scores as compared to existing models edging out previous scores of 97% and 94% by 98.3% and 95% respectively.

In the latest work compared to the rest, (K et al.; 2022) built a model with multiple layers consisting of the convolutional layer, pooling layer, fully connected layer, dropout layer and the authors have also used activation functions. The dataset consists of 4 classes of diseases. The authors have described the testing precision as 99.6% and to be better than previous work. The authors have not given a lot more information based on the performance of the model but have focused on the working and needs of the different layers of a CNN.

Through the review of previously done work, it is visible that a lot of research has been done on the detection of diseases in plants other than cotton. Many different types of augmentations were used that helped in understanding the need of having a good number of images to train the model well. Understanding the working of different types of models, it could be seen that Machine Learning models were not the best suited for image classification and customized CNN models were the best way to go about it. Since VGG16 and DenseNet were top performing transfer learning models in previous work, they could be used as a point of contention against a customized model built with different techniques.

3 Methodology

For every successful project, a certain method or path needs to be followed for its execution. Systematic steps and a process helps to understand the progress done and how the project is moving forward. The authors have chosen the Knowledge Discovery of Database (KDD) methodology in this research as the steps present in KDD were linear to the path the author thought would be best to follow during the research. (Vishwakarma and Jain; 2022), in their work have discussed the importance of KDD and how the steps present in it are beneficial towards the progress of the project. The steps followed for this research are as follows:

- Data Selection
- Data Pre-processing
- Data Transformation
- Use of appropriate Data Mining Techniques
- Evaluation of the result

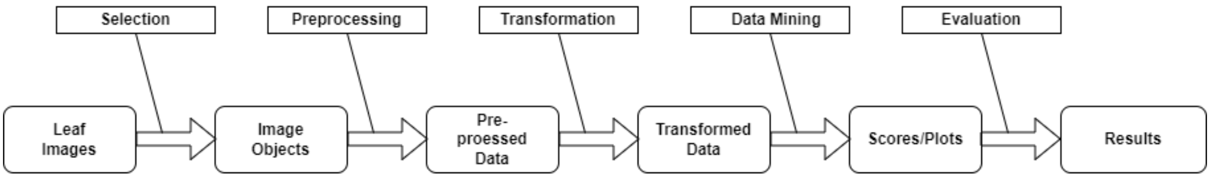

Figure 1: Research Design

Figure 1 gives a visual representation of how the methodology was followed through the research.

3.1 Data Selection:

As seen in Figure 1, the first step of the research was to select a suitable dataset. It is necessary to a get an appropriate and suitable dataset as the data needs to be good enough to train the model. Kaggle being an open-source repository for data, provided the author with options to choose a suitable dataset. Until recent times, there were not a lot of datasets for cotton plant disease detection as the need for it was not given enough importance. The dataset used for this research is the Cotton Disease Dataset[1] from Kaggle. The dataset contains 4 classes, that are, *Healthy plants, Diseased plants, Healthy leaves and Diseased leaves*. The datatset has 2310 images split into the four classes.

From table 1, the distribution of the data among classes is visible. It can be seen that the images are not very unbalanced and this helps train the model better.

[1]https://www.kaggle.com/datasets/janmejaybhoi/cotton-disease-dataset

Table 1: Data Distribution

Class	**No. of Images**
Diseased cotton leaf	356
Diseased cotton plant	921
Fresh cotton leaf	519
Fresh cotton plant	514
Total	2310

3.2 Data Pre-processing:

Before the data is used to train a model, the data needs to be pre-processed. Data pre-processing is one of the most important steps that needs to be followed before building a model as it prepares the data to be used. Normally the data cleaning is done to make the data more usable but the data in the dataset that is used is already clean and with minimal noise. First, the images are loaded by parsing the files one after the other and reading the images on. The next step that was followed was the resizing of the images. The images are resized depending on the dimensions that can be input in the model. By doing so the size of all images remain constant which helps in training the model well. The dimension that all the images were resized to was 256*256.

3.3 Data Transformation:

Once the data is pre-processed, it needs to be transformed into the desirable form before it can be used. Since the dataset has only 2310 images, the model is susceptible to overfitting. Overfitting of a model can cause the model to perform poorly when tested. To avoid this the dataset is augmented so that the model can perform better. Five different types of augmentation techniques were used in this paper from the 'skimage' and 'matlab' libraries. The techniques used are given below:

- *adjust_gamma(gamma=0.5)*: With this technique gamma correction is done on the image at the power of 0.5
- *adjust_gamma(gamma=2)*: This is the same technique with different gamma correction value creating a different set of augmented images.
- *fliplr*: This technique flips the image left to right.
- *flipud*: This technique flips the image up to down.
- *random_noise*: Using this technique, random noise was added in one set of the augmented images.

Table 2 shows the values in the classes after the data is augmented. It can be seen that the number of images have significantly increased which can help avoid overfitting in the model.

Table 2: Data Distribution after Augmentation

Class	**No. of Images**
Diseased cotton leaf	1780
Diseased cotton plant	4605
Fresh cotton leaf	2595
Fresh cotton plant	2570
Total	11550

3.4 Use of appropriate Data Mining Techniques:

After the data is transformed, an appropriate model is to be made to detect the disease in the cotton plant leaves and stems. Through the literature review, it can be noticed that CNN models and transfer learning models perform better when image based classification is considered as compared to Machine learning models. In this research, a CNN model has been built to detect the disease in the cotton plants. Apart from the CNN model, Transfer Learning has also been used, in which pre-trained models like VGG16 and DenseNet121 have been used. These models are built and have preset features with appropriate changes made to the final few layers. The output of these models are then evaluated to see which model performs better and how they can be made better. Therefore in this paper, Transfer Learning and a CNN model is applied to detect diseases in cotton plants.

3.5 Evaluation of the result:

Once the model is trained, the performance of the model needs to be tested so that if the performances are poor, changes can be made to get the satisfactory results. These are many metrics that can be used to understand the performance of a model. The metrics that are used in this paper are Accuracy, Precision, Recall and F1 score.

- **Accuracy:** The accuracy of a model is the strength at which the model can identify the correct class.
- **Precision:** This metric can tell us how accurately can the model identify the positive class out of the total positive class data.
- **Recall:** This metric tells us how accurately can the model identify positive class out of the entire dataset including all the classes.
- **F1-score:** The F1 score is a metric that is used when there is a trade-off between the precision and recall. It helps understand capability of the model to identify the positive class and negative class.

4 Design Specification

This section discusses in detail the flow of the research in the paper. The steps followed through this research, keeping KDD in mind, is shown in the Figure 2. A few steps have already been discussed in the previous section.

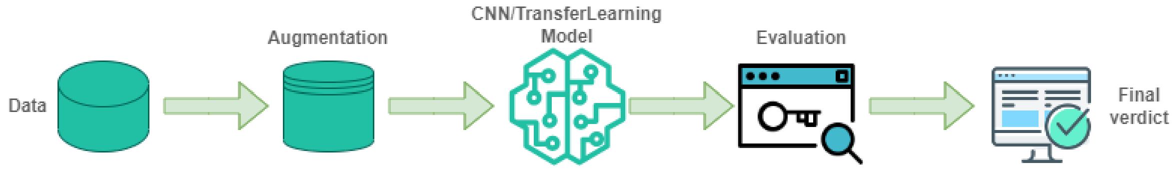

Figure 2: Research Design

The first step was to load the data into the Jupyter IDE[2]. Once the data was loaded, since the size of the data was not enough and the data was susceptible to overfitting, augmentation of the dataset was performed. This augmentation was done using five different techniques as mentioned in the above sections where the number of images were increased from 2310 to 11550. After the data is augmented all the images are resized to 256*256 before they can be used for the models. The architecture of the models are further discussed below.

4.1 Transfer Learning

VGG16 and DenseNet121 are two pre-trained models that are used in transfer learning to detect the diseases in the cotton plants. Both of these models were built and trained on the 'ImageNet' Dataset and the same weights were used in these models.

4.1.1 VGG16

In the paper by Tao et al. (2021), they have highlighted how VGG16 is one of the most commonly used pre-trained model. As the name suggests it has a total of 16 weighted layers. The convolutional layers have the same structure. Apart from the convolutional layer, it consists of MaxPooling layers, fully connected layers and has a few hidden layers with the ReLu activation function. Since this model has been trained with 'ImageNet' before, the weights are directly taken from there.

4.1.2 DenseNet121

DenseNet121 was used since it was one of the smallest DenseNet models and could be easily replicated while not being very different from the other versions. Bing-jin et al. (2020) explain in their paper that Densenet is similar to Resnet. The key difference that can be observed between the two is that all the layers in the model are densely connected with layers before and after them, hence getting the name. Like the VGG16 model, weights from the 'ImageNet' dataset model are used in this paper. It has a total of 120 convolutional layers and including these layers there is one more trainable layer. Hence it is called DenseNet 121.

4.2 CNN model

A CNN model is made of multiple layers that help process and classify the data. Some of these layers are the convolutional layer, MaxPooling layer, dense layer, etc. The model used in this paper makes use of most of these layers. Figure 3 gives an overview of the architecture of the CNN model.

[2]https://jupyter.org/

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 254, 254, 32)	896
max_pooling2d_5 (MaxPooling2D)	(None, 127, 127, 32)	0
batch_normalization_5 (BatchNormalization)	(None, 127, 127, 32)	128
conv2d_6 (Conv2D)	(None, 125, 125, 64)	18496
max_pooling2d_6 (MaxPooling2D)	(None, 62, 62, 64)	0
batch_normalization_6 (BatchNormalization)	(None, 62, 62, 64)	256
conv2d_7 (Conv2D)	(None, 60, 60, 64)	36928
max_pooling2d_7 (MaxPooling2D)	(None, 30, 30, 64)	0
batch_normalization_7 (BatchNormalization)	(None, 30, 30, 64)	256
conv2d_8 (Conv2D)	(None, 28, 28, 96)	55392
max_pooling2d_8 (MaxPooling2D)	(None, 14, 14, 96)	0
batch_normalization_8 (BatchNormalization)	(None, 14, 14, 96)	384
conv2d_9 (Conv2D)	(None, 12, 12, 32)	27680
max_pooling2d_9 (MaxPooling2D)	(None, 6, 6, 32)	0
batch_normalization_9 (BatchNormalization)	(None, 6, 6, 32)	128
dropout_2 (Dropout)	(None, 6, 6, 32)	0
flatten_2 (Flatten)	(None, 1152)	0
dense_3 (Dense)	(None, 128)	147584
dropout_3 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 4)	516

Total params: 288,644
Trainable params: 288,068
Non-trainable params: 576

Figure 3: Model Summary

- As seen in Figure 3, the model consists of 5 convolutional layers with different parameters and output shapes. The convolutional layer takes in the images and converts them into features or parameters.
- Each of these convolutional layers are followed by Maxpooling layers that reduce the parameters that are received from the convolutional layer into a 2*2 matrices. The MaxPooling layer only picks up the very important parameters.
- Once the important features are picked using the MaxPooling layers, Batch Normalization is used. This helps the process faster and helps the model to learn better.
- These 3 layers are present in blocks and are consecutively used 5 times. The output of the fifth block goes through a dropout function which helps control the overfitting in the model.
- Finally the output which is a filter map is then flattened. This flattened output is then put through a dense layer which is a fully connected layer that gives that classification of the image.

5 Implementation

This section discusses how the models were implemented in order to be able to efficiently and accurately identify the diseased plants from the plants that were not diseased. The environment, packages and the parameters used will be discussed in the subsections below.

5.1 Environment

For the research done in this paper, python 3.9.1 was used with the Jupyter Notebook environment. Many libraries and packages were used, out of which one of the most important ones was 'TensorFlow'[3]. 'TensorFlow' is mainly used for image processing. It is one of the 'Keras'[4] was used to make the most of the TensorFlow library. The 'skimage' library was used for augmentation purposes. Functions like random_noise and rotate were used to change the images.

5.2 Architectures

The basic architectures of the models are discussed in previous sections. The more minute details about the models are seen in this subsection.

5.2.1 Transfer Learning

As discussed earlier, since both the transfer learning model were pre-trained with the 'imagenet' dataset, the weights were already set in those models. The output was then flattened using a flatten layer. This flattened output was then put through a Dense output layer with 'SoftMax' activation function. This is done to further normalize the output. As an optimizer for the model, 'Adam' optimizer was used. The number or epochs were set at 30 for both the models.

5.2.2 CNN model

The CNN model had a lot of layers and the parameters were set to optimize the performance of the model. As discussed earlier, each block in the model consists of a convolutional layer, MaxPooling layer and batch normalization. All the layers needed in this model were imported from the TensorFlow library. The batch size for the model was given as 32 and the model ran for 30 epochs which gave significant results. The number of filters used in the convolutional layers are 32, 64 and 96. The kernel size remains the same throughout being 3*3. The 'ReLu' activation function is used in all the blocks as it makes all the negative values of the neurons to 0. This helps the values not go too far wide apart. Keeping the output similar to the transfer learning models, 'SoftMax' activation function is used in the Dense layer at the output. Similar to the previous models, 'Adam' optimizer is used for the model. 'Categorical_crossentropy' is used to calculate the loss of the model

6 Evaluation

The models built and tested out in this paper are to detect diseases in cotton plants and hence the results show how well suitable the model is for the cause. Out of the multiple metrics that can be used to determine the performance of the model, the author chose to use the accuracy, F1 score, precision and recall. The author also looked at the accuracy plot and loss plot of the models through the time they were running.

Just by looking at the accuracy of the models in Table 3, VGG16 seems to have the best accuracy out of the three by marginally beating the custom CNN model. Just taking

[3] https://www.tensorflow.org/
[4] https://keras.io/

Table 3: Results of the models

Model	Accuracy
Custom CNN	96.04%
VGG16	96.83%
DenseNet121	80.63%

the accuracy into consideration would deem VGG16 as the best model but the author has made an in-depth evaluation of the models which is given below.

6.1 VGG16:

VGG16 being one of the most used pre-trained models among the three was expected to have very good results.

```
Accuracy:0.9683794466403162
F1 Score:0.9683794466403162
Confusion Matrix:
[[41  0  2  0]
 [ 0 75  0  3]
 [ 1  0 65  0]
 [ 0  1  1 64]]
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.99      0.96      0.97        78
           2       0.96      0.98      0.97        66
           3       0.96      0.97      0.96        66

    accuracy                           0.97       253
   macro avg       0.97      0.97      0.97       253
weighted avg       0.97      0.97      0.97       253
```

Figure 4: VGG16 Scores

From Figure 4, it can be seen that the scores of VGG16 and considerably very good. The confusion matrix can be used to get scores like the precision, recall and F1 score. It can be seen that the F1 score and accuracy both being 96.83% are on the higher side which is very encouraging and the individual precision and recall score for each individual instance is above 95%.

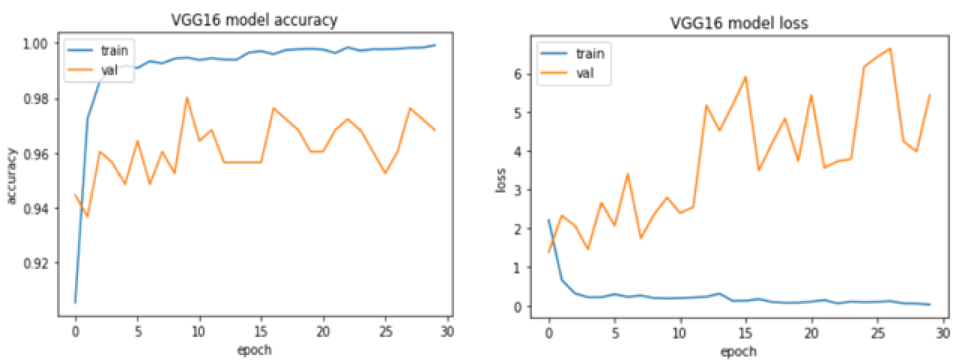

Figure 5: VGG16 Plots

The plots in Figure 5 show the history plots of the VGG16 models accuracy and loss over the course of 30 epochs. The model accuracy plot shows a steep increase in the first

5 epochs after which there is a slight gradual increase over the next 25 epochs showing that the model has accuracy is nearly at its peak. Similarly the model loss model has a drop in loss after the first 5 epochs and then there is a gradual decline which is a good sign.

6.2 DenseNet121:

DenseNet121 is the second pre-trained model that is to be evaluated after VGG16.

```
Accuracy:0.8063241106719368
F1 Score:0.8063241106719368
Confusion Matrix:
[[39  2  2  0]
 [ 0 78  0  0]
 [ 4  3 59  0]
 [ 0 38  0 28]]
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.91      0.91        43
           1       0.64      1.00      0.78        78
           2       0.97      0.89      0.93        66
           3       1.00      0.42      0.60        66

    accuracy                           0.81       253
   macro avg       0.88      0.81      0.80       253
weighted avg       0.87      0.81      0.79       253
```

Figure 6: DenseNet121 Scores

Figure 6 shows the scores for DenseNet121. The accuracy and F1 Score is as low as 80.6% for the DenseNet121 model as compared to the VGG16 model. The individual scores for precision and recall show no consistency as they range 64% to 100% which is a very wide range and does not make the model ideal.

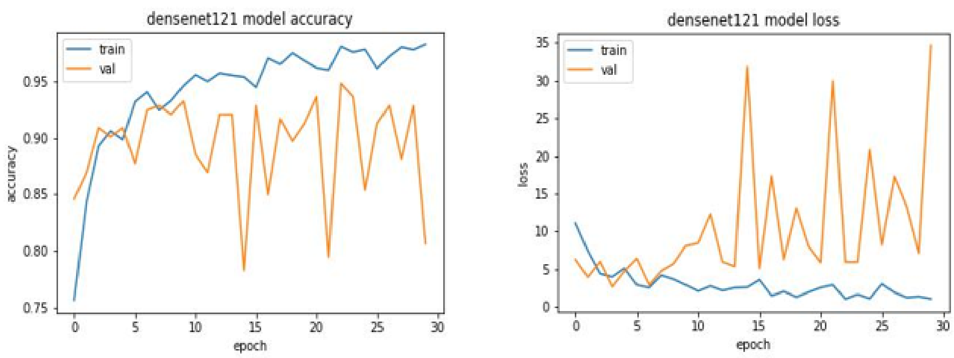

Figure 7: DenseNet121 Plots

It can be further seen from Figure 7 that the training accuracy of the model had an gradual rise throughout while the loss was very low by the end of the 30 epochs. This works in the favour of the DenseNet121 model. The validation accuracy and loss follow a very inconsistent pattern with frequent rise and fall in both the graphs. This makes the outcome of this model a bit unpredictable which dies not look good for the model.

6.3 Custom CNN model:

The CNN model built by the authors need to be finally evaluated to see how well it performs compared to the other two models.

```
Accuracy:0.9604743083003953
F1 Score:0.9604743083003953
Confusion Matrix:
[[43  0  0  0]
 [ 0 74  0  4]
 [ 2  1 63  0]
 [ 1  2  0 63]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      1.00      0.97        43
           1       0.96      0.95      0.95        78
           2       1.00      0.95      0.98        66
           3       0.94      0.95      0.95        66

    accuracy                           0.96       253
   macro avg       0.96      0.96      0.96       253
weighted avg       0.96      0.96      0.96       253
```

Figure 8: Custom CNN Scores

As seen in Figure 8, the accuracy and F1 score are 96% which is really good accuracy score for a custom CNN model as the model is run through many iterations and made better by making adjustments. The precision and recall score range from a 93% to a 100% in different instances which shows that the model is performing very well under all scenarios.

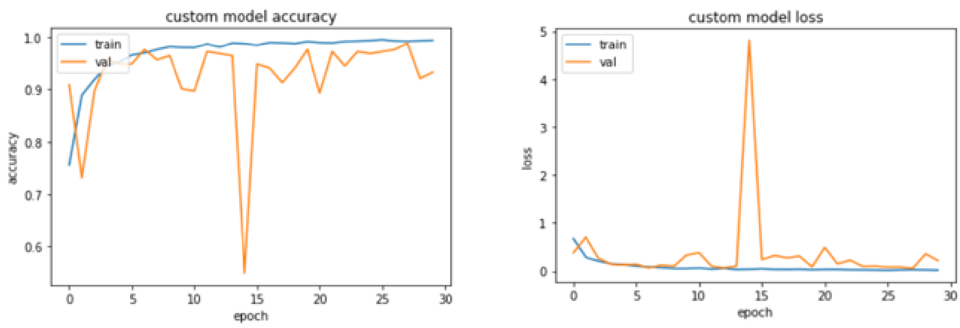

Figure 9: Custom CNN Plots

Figure 9 shows the plots for the CNN model accuracy and loss. The train accuracy and loss plot shows a gradual increase in a very consistent manner which shows that the model learnt better over time. The validation plot was quite consistent too except during the 14^{th} epoch where the accuracy dropped drastically and the model loss increased drastically too.

6.4 Computational Time:

In terms of selecting the best model, the scores and accuracy are the most important factor but the computational time and power required for these models is quite important. Not everyone would be able to afford high computational power or have enough time. Among the three models, VGG16 and DenseNet121 took as much as twice the amount of time as compared to the custom CNN model. This is because of the vast number of layers and parameters that these pre-trained models contain. The computational power also required for these pre-trained models is quite high as compared to the custom CNN model.

6.5 Discussion

The above subsections gave details about how the models performed in different aspects. All the necessary metrics such as the accuracy, F1 score, precision, recall and the history plots were observed. Just by looking at the accuracy score, VGG16 was the best model slightly outperforming the CNN model by 0.79%. Apart from the accuracy when the other metrics were taken into consideration, it could be seen that the precision and recall of the CNN model was better than the VGG16 model with a 100% accuracy in some instances.

Taking all the model history plots into consideration, it is can be said that the consistency obtained by the CNN model over the course of 30 epochs was more than that of the pre-trained models. This shows that the CNN model was learning over the course of the 30 epochs and was only getting better. Apart from that, the computational time and power taken by the CNN model is less than half of that of the pre-trained models.

So finally only comparing the VGG16 model, which has the best accuracy among the pre-trained models and the CNN model, it can be said that apart from the slightly lower accuracy, the CNN model is more efficient.

7 Conclusion and Future Work

It is necessary to have solutions in place when diseases might affect an entire field of plants or crops. The main aim of this paper was to build a CNN model good enough to automatically detect and classify diseased plants from the healthy ones. The model built could identify diseased plants with an accuracy of 96.04% which is a fairly good score and also took less computational time and power as compared to pre-trained models. Compared to VGG16 and DenseNet121 pre-trained models, it could be said that the CNN model performed the best.

As for future scope, this model could be integrated into a system where drones could be used to fly around the entirety of a field capturing the images of the plants and immediately classify the plants. The model could rather be improvised more to get better accuracy while classifying the images in real-time.

References

Alyas, R. M. and Mohammed, A. S. (2022). Detection of plant diseases using image processing with machine learning, *2022 2nd International Conference on Computing and Machine Intelligence (ICMI)*, pp. 1–6.

Bing-jin, L., Jian, Y., Yan-jun, L., Liang, P. and Guo-xiong, L. (2020). Research and practice of x-ray chest film disease classification based on densenet, *2020 International Conference on Artificial Intelligence and Education (ICAIE)*, pp. 241–244.

Deepa, N, R. and Shetty, C. (2021). A machine learning technique for identification of plant diseases in leaves, *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pp. 481–484.

Dutta, L. and Rana, A. K. (2021). Disease detection using transfer learning in coffee plants, *2021 2nd Global Conference for Advancement in Technology (GCAT)*, pp. 1–4.

Gobalakrishnan, N., Pradeep, K., Raman, C. J., Ali, L. J. and Gopinath, M. P. (2020). A systematic review on image processing and machine learning techniques for detecting plant diseases, *2020 International Conference on Communication and Signal Processing (ICCSP)*, pp. 0465–0468.

Guan, X. (2021). A novel method of plant leaf disease detection based on deep learning and convolutional neural network, *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pp. 816–819.

K, K. J., M, S., G, D., R, I. and T, Y. (2022). Identification of maize plant leaf disease detection using convnet model, *2022 8th International Conference on Smart Structures and Systems (ICSSS)*, pp. 1–3.

Kobayashi, K., Tsuji, J. and Noto, M. (2018). Evaluation of data augmentation for image-based plant-disease detection, *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2206–2211.

Lakshmanarao, A., Babu, M. R. and Kiran, T. S. R. (2021). Plant disease prediction and classification using deep learning convnets, *2021 International Conference on Artificial Intelligence and Machine Vision (AIMV)*, pp. 1–6.

Militante, S. V., Gerardo, B. D. and Dionisio, N. V. (2019). Plant leaf detection and disease recognition using deep learning, *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*, pp. 579–582.

Nagi, R. and Tripathy, S. S. (2021). Grapevine leaf disease identification using transfer learning, *2021 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE)*, pp. 43–46.

Oh, S., Lee, D.-Y., Gongora-Canul, C., Cruz-Sancan, A., Ashapure, A., Fernandez, M., Telenko, D., Jung, J. and Cruz, C. (2020). Estimation of visual rating of tar spot disease of corn using unmanned aerial systems (uas) data and machine learning techniques, *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pp. 4882–4885.

Pajjuri, N., Kumar, U. and Thottolil, R. (2022). Comparative evaluation of the convolutional neural network based transfer learning models for classification of plant disease, *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pp. 1–6.

Rubini, P. and Kavitha, P. (2021). Deep learning model for early prediction of plant disease, *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pp. 1104–1107.

Sardogan, M., Tuncer, A. and Ozen, Y. (2018). Plant leaf disease detection and classification based on cnn with lvq algorithm, *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pp. 382–385.

Sehgal, A. and Mathur, S. (2019). Plant disease classification using soft computing supervised machine learning, *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, pp. 75–80.

Sharma, P., Hans, P. and Gupta, S. C. (2020). Classification of plant leaf diseases using machine learning and image preprocessing techniques, *2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pp. 480–484.

Tao, J., Gu, Y., Sun, J., Bie, Y. and Wang, H. (2021). Research on vgg16 convolutional neural network feature classification algorithm based on transfer learning, *2021 2nd China International SAR Symposium (CISS)*, pp. 1–3.

V, P. K., Rao, E. G., Anitha, G. and Kumar, G. K. (2021). Plant disease detection using convolutional neural networks, *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1473–1476.

Vishwakarma, P. K. and Jain, N. (2022). A review of data analysis methodology, *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pp. 356–360.

Wongbongkotpaisan, J. and Phumeechanya, S. (2021). Plant leaf disease classification using local-based image augmentation and convolutional neural network, *2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 1023–1027.