International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

# Enhancing Cotton Crop Health: A Data-Driven Approach for Disease Detection and Yield Optimization Through Tuned VGG-16 Model

**Samuel Chepuri*[1], Dr. Y. Ramadevi[2]**

**Submitted**: 29/01/2024 **Revised**: 07/03/2024 **Accepted**: 15/03/2024

**Abstract:** Cotton is a valuable cash crop. Timely disease detection and management can help increase crop yields and overall agricultural productivity. Healthy crops produce higher-quality cotton fibers, which are essential for the textile industry. Maintaining a healthy cotton crop contributes to food security and economic stability, especially in areas where cotton is a primary source of income. Traditional image processing techniques extract relevant features from the segmented leaf images. These features can include color histograms, texture descriptors, shape characteristics, and more. Feature extraction helps capture the distinctive patterns associated with healthy and diseased leaves. Cotton diseases can manifest in various ways, and their visual symptoms can vary based on factors such as disease stage, environmental conditions, and cotton variety. This variability can make it challenging to develop a one-size-fits-all image processing solution. The proposed model tunes the VGG-16 to perform the feature extraction and solves the problem of Variability in Disease Symptoms. Total 8 parameters are available for tuning the VGG-16 but the proposed model focuses on the learning rate, dropout rate and optimizer. These hyperparameters significantly impact the model's performance, convergence speed, and generalization ability. Without tuning the model has got 82.18% accuracy but after tuning the model has got 92.01%, which means that nearly 10% improvement in the designed process.

**Keywords:** *Cash Crop, Dropout Rate, Feature Extraction, Learning Rate, Optimizer, Tuning Parameters, Variability, VGG-16*

## 1. Introduction:

Cotton is a vital cash crop used in the textile industry. Diseases can significantly reduce cotton yield and fiber quality. Detecting and managing diseases can help maximize crop yield and maintain fiber quality. Image-based disease detection enables farmers to apply treatments, such as pesticides or fungicides, precisely where they are needed. This targeted approach reduces the unnecessary use of chemicals, lowering production costs and minimizing environmental impact. The below section discusses about the different deep learning models existing for the pre-trained approaches:

### 1.1. Meta Approaches in Deep Learning:

The dataset contained 2k pictures of cotton leaves, categorized into seven classes. By utilizing multiple models for cotton leaf disease detection, including a customized CNN, couple of pre- trained models & meta-learning methodologies. The custom CNN consisted of five conv, dropout, & max-pool. The SoftMax layer with 7 classes was used for disease diagnosis. The SoftMax layer with 7 classes was used for disease diagnosis. VGG16, a pre-trained methodology, was utilized with the top layers frozen this was also done with Resnet50 to acquire high accuracy. These methodologies used layered ensemble learning to combine to improve acquired accuracy. By combining multiple models, pre-trained networks, and ensemble learning techniques allowed for robust disease detection in cotton leaves. The dataset was carefully collected, annotated, and pre-processed to ensure the effectiveness of the models. Proper hyperparameter settings and model selection contributed to high accuracy in disease classification.

[1] *Research Scholar, Department of Computer Science and Engineering Osmania University, Hyderabad,Telangana, India-500017*
*ORCID ID: 0000-0002-4620-7977*
[2]*Professor, Department of AIML, CBIT, Hyderabad, Telangana, India.*
*ORCID ID: 0000-0001-8950-7761*
* *E-mail: drsamuelchepuri@gmail.com*

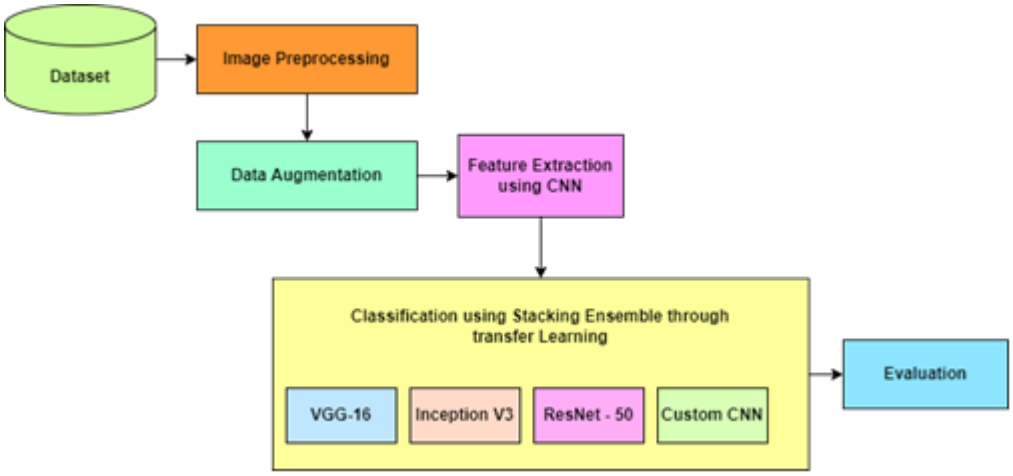


**Fig 1:** Working Model for Stacking Ensemble Pre-trained Model

### 1.2. Pre-trained Based Deep Learning Approach

The New Plant Village dataset, containing 1k images of couple of different leaves, is used for training and testing. The methodology focuses on identifying diseases in cotton plant leaves using classification methods. Image processing techniques help prepare the data for classification. The methodology utilizes CNNs for disease identification. A combination of pre-trained models from ImageNet and the DenseNet-121 component is employed to leverage techniques. The methodology outlines various steps for cotton leaf disease detection, the pre-processing of images, division, and extraction of features, choosing features, and classification using multiple techniques. Features such as histograms and wavelets are extracted from segmented images. Various five ML classification methodologies are applied to classify the selected features and distinguish healthy and diseased images. DenseNet-121, a well-performing image classification model, is utilized for decrease errors & increases the performances. It uses a unique approach to connect layers directly to each other, enhancing feature propagation. The diseased images are analyzed using several classification techniques.

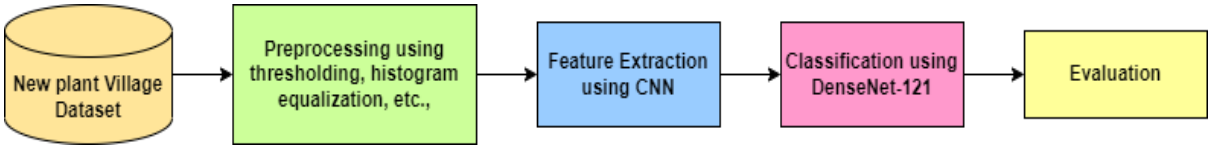


**Fig 2:** DenseNet-121 Approach

### 1.3. CNN integrated ML Approach

The integration of CNN with ML methodology for cotton leaves disease detection is a promising approach that addresses several important needs in agriculture and crop management. Early detection of diseases in cotton leaves is crucial for preventing crop loss. This early detection enables timely intervention, reducing the spread of diseases and crop damage. ML models can analyze the data generated by CNNs to determine the severity and type of disease. This information can be used to precisely target affected areas with the appropriate treatment, reducing the use of pesticides and saving resources. By detecting diseases early and precisely, farmers can allocate resources more efficiently. Integrated systems can minimize the utilization of fertilizers and chemicals to lessen the harmful impacts of farming. CNNs excel at processing large volumes of image data quickly and accurately. This makes them suitable for monitoring vast fields of cotton crops.

### 1.4. Limitations of Feature Extraction using CNN

Features extracted by CNNs are often tailored to the specific dataset and task they were trained on. Transferability to different tasks or domains can be limited, especially if the source and target domains differ significantly. CNN-based feature extraction produces fixed-size feature vectors. Training and using CNNs for feature extraction can be computationally intensive, particularly for deep networks. CNN models, especially deep ones, have a large memory footprint due to their numerous layers and parameters. CNNs typically have a limited receptive field, which means they may miss contextual information that is crucial for understanding the overall scene. When using pre-trained CNN models for feature extraction, there is a risk of overfitting to the source dataset, especially if the

target dataset is small or significantly different. Fine-tuning or regularization techniques are often needed to mitigate this issue.

Visual Geometry Group architectures have a simple and uniform structure, with small convolution filters (3x3) and max-pooling layers used throughout the network. This uniformity makes it easier to understand and work with the architecture, and it can also facilitate model design and modification. 8 Models are available in VGG. Out of these 8 only VGG-16 & VGG-19 are popularly used. But other models are suffering from limitations which are presented in table 1.

**Table 1:** Limitations of VGG Versions

| Variants | Limitations |
|---|---|
| VGG11 | VGG-11, due to its use of small convolutional filters (3x3) and max-pooling layers, may have limitations in capturing global contextual information in images. |
| VGG13 | VGG-13 strikes a balance between model depth and performance, which may not be optimal for all tasks. For tasks requiring more intricate feature representations, deeper models might be more suitable, while for lightweight applications, shallower architectures might suffice |
| VGG-M | The deployment of VGG-M models, particularly for real-time or edge device applications, may still require substantial computational resources, depending on the specific use case and hardware limitations. |

The proposed model utilizes the pre-trained model VGG-16 for feature extraction because it is a versatile and powerful tool for feature extraction in computer vision tasks, thanks to its rich hierarchical features, pre-trained model availability, and compatibility with popular deep learning frameworks.VGG-16 has a deep architecture with 16 layers, which allows it to learn rich hierarchical features from input images. This means it can capture both low-level features like edges and textures and high-level features like object parts and complex patterns. The traditional architecture of VGG-16 is presented in the figure 3

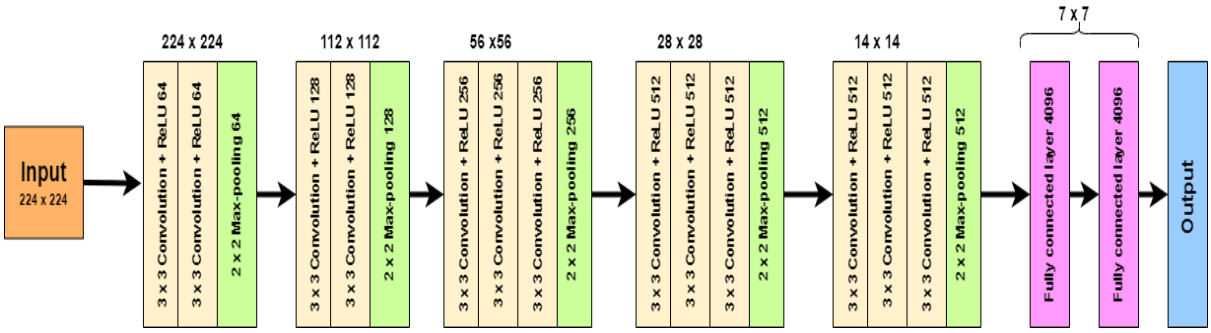


**Fig 3:** Traditional Architecture of VGG-16

## 2. Literature Survey:

Sandeep Kumar et al [1] has focused on CNN methodology for detection of cotton diseased leaves. A comparative study was conducted on frameworks for model creation, particularly for an iOS project focused on image classification. This involves supervised learning, training a model to recognize target classes based on labelled images. It includes single-class & multiclass tasks. Feature extraction involves obtaining relevant data elements from images. Convolutional layers in CNNs perform this function, creating feature maps to represent image patterns. Activation functions, like ReLU, enhance nonlinearity in feature maps after creation. Pooling layers compress and abstract image information, making networks more proficient at identifying relevant features, thus preventing overfitting. Flattening represents the final CNN layer, converting feature maps into a sequential vector for processing. The machine learning workflow comprises four phases: Data Set Preparation, TensorFlow Model Creation, Model Training, and Model Evaluation.

Abu Sarwar Zasmani et al [2] has implemented DL, ResNet50 methodology for detection of cotton diseased leaves. The benchmark dataset of plant leaf disease images is obtained from PlantVillage, consisting of labelled images of various crops. The proposed approach includes four parts: handcrafted feature extraction, deep feature extraction, feature fusion and selection using PCA, and LDA-based classification. Four types of handcrafted features MGH, LBP, HOG,

& GLCM are extracted from input images. A modified ResNet50 model is employed for deep feature extraction using transfer learning. Handcrafted features and deep features are concatenated for generating a hybrid vector features for improved classification. PCA is utilized to decrease the dimensionality related to vectors of hybrid features while maintaining important discriminant features. The reduced feature set is used for developing the PCA-LDA classification model, where LDA maximizes the separation between classes. The trained PCA-LDA model is applied to an independent dataset for prediction. The model's performance is evaluated using standard quality measures.

C. Murugamani et al [3] has proposed a SVM methodology for detection of cotton diseased leaves. The text discusses the potential applications of WSNs in agriculture, enabling real-time monitoring of various factors such as climate, soil conditions, pest detection, and more. Improved methods of managing farming equipment and automation using WSNs are explored, facilitating better control and management of large-scale farms. WSNs aid in controlled pesticide and fertilizer usage, reducing expenses and improving crop quality. The proposed system involves various components, including nine attribute valves for data collection and control. An IoT cloud is utilized for data exchange between devices, enabling storage, management, and analysis of data collected from sensor nodes. IoT devices like Raspberry Pi and sensors facilitate data acquisition and analysis. The system involves preprocessing, segmentation, & extraction of features of images for disease detection. Various classification methods are discussed, including RF, NB, & SVM along with performance measurement metrics.

Lavika Goel et al [4] has focused on a SVM methodology for detection of cotton diseased leaves. Plant diseases can be classified into biotic diseases caused by living agents & abiotic diseases caused by non-infectious agents. Biotic diseases can spread among plants and damage various plant parts. Plant diseases can be categorized based on the infected plant type, infected organ, and the type of phytopathogen causing the disease. Plant disease recognition models use machine learning and IP methodologies is utilized for detection of diseases regarding various plant parts. K-NN is based on the similarity between data points and assigns class labels based on the nearest neighbours. ANN simulates the brain's functionality and learns from data to predict unknown values. SVM creates decision boundaries to separate classes in multi-dimensional space. utilizing CNNs and combining SVM with deep features, achieving high accuracy in identifying plant diseases. Transfer learning and modifications to model parameters have enhanced classification results for various crops.

Ahmed Elaraby et al [5] has focused on a DCN, AlexNet, & PSO methodology for detection of cotton diseased leaves. The proposed framework focuses on using deep learning for plant disease detection. This model is pre-trained on a large dataset. The modified AlexNet CNN is employed due to its strong feature extraction capabilities. Data augmentation is applied to the training dataset to artificially expand it and enhance generalization. Deep neural networks are employed for feature extraction. PSO helps in choosing the most significant features. The classification phase involves splitting the dataset into training and test sets. Transfer learning is applied, and the optimal set of features chosen by PSO is used to train the classifier. This process helps categorize images into different disease categories. The hyperparameters of the AlexNet model are optimized using PSO techniques. Control parameters, such as those governing PSO optimization, are determined to achieve the best outcomes.

Serosh Karim Noon et al [6] has initialized SPP, YOLOX methodologies for detection of cotton diseased leaves. The proposed dataset consists of cotton plant images captured in regions. The images captured in the field contained background information. The algorithm estimates color distribution using Gaussian Mixture Model and applies graph cuts to separate foreground and background. By improving the images the data is training performance and reduce overfitting. The dataset size increased to 1k images after augmentation. In order to overcome the difficulties of illness development symptoms and the coexistence of multiple conditions on a single leaf, the suggested approach must take these issues into account. YOLOX, an anchor-free object detection model, is utilized for detecting disease symptoms and severity levels. The model's architecture includes CSP Darknet for feature extraction, FPN for feature fusion, and a focus module for feature enhancement. The proposed YOLOX model is adapted to identify several disease symptoms particularly on each leave to identify disease typical condition.

K. Indumathy et al [7] has introduced a AFMRCNN methodology for detection of cotton diseased leaves. The proposed methodology involves image processing, augmentation, enhancement, and subsequent classification using the AFMRCNN model. Pixel augmentation techniques are used for data preprocessing, creating artificial images that are different from the original yet suitable for training. The AFMRCNN model performs classification between normal and infected leaves based on feature extraction from enhanced images. Image augmentation

encompasses various steps to modify data for training. It involves creating new images through transformations while maintaining their suitability for training. Histogram equalization and normalization are employed to adjust image contrast. It uses convolutional neural networks to extract features and performs background and foreground classification with regression boundary boxes. The training process of AFMRCNN involves inputting images with pixel-level transformations, extracting features through convolutional neural networks, and classifying regions based on disease presence.

Bhagya M. Patil et al [8] has proposed a Chan Vese methodology for detection of cotton diseased leaves. These images were captured near Kalaburgi in Karnataka. Bilateral filtering is utilized for edge-preserving smoothing. The process employs linear Gaussian smoothing, with the addition of a weighting term that considers both spatial distance and intensity difference between pixels. The methodology involves processing images through bilateral filtering, leading to the removal of noise and smoothening. The suggested technique won't re-initialize for leaf segmentation; instead, it combines the Chan-Vese approach and the level set method. The chanvese method introduces curve evolution without edges, aiming to find continuous image edges without dependence on image gradient. By forcing the level set function to stay near to the signed proximity function, the level set technique without re-initialization does away with the necessity for re-initialization.

**Table 2:** Comparative Analysis of Existing Approaches

| **Author** | **Algorithm** | **Merits** | **Demerits** | **Accuracy** |
|---|---|---|---|---|
| Sandeep Kumar et al | CNN | But utilizing minimum layer the time complexity is very less. | At particular extent the diseases are identified. | 90% |
| Safdar Ali et al | DL, ResNet50, | This can be used for automation. | This works only for cotton crop. | 98.2% |
| C. Murugamani et al | SVM | By selecting vector the model is easily developed. | The device should be connected to network all the time. | 98.34 % |
| Lavika Goel et al | SVM | The method has appropriate leaf prediction. | The model only identifiesthe particular diseases. | 99.9% |
| Ahmed Elaraby et al | DCN, AlexNet, PSO | Any kind of leaf disease can be detected. | Time complexity is high when the data is large. | 98.8% |
| Serosh Karim Noon et al | SPP, YOLOX | This method is efficient. | Validation was not proven accurately. | 73.1% |
| K. Indumathy et al | AFMRCNN | This was introduced for DM techniques. | The accuracy was different for each disease. | 99% |
| Bhagya M. Patil et al | Chan Vese | No need of re-initialization. | Prediction time should be decreased. | 88.9% |

## 3. Proposed Methodology:

Deep learning models, such as VGG-16, have the potential to detect diseases at their early stages when visual symptoms might not yet be apparent to human observers. This early detection allows for timely intervention and treatment.Deep learning models can be deployed on various platforms, including edge devices and drones, allowing for scalable and remote monitoring of large cotton fields. The proposed methodology initially starts with data augmentation as discussed in the below section

### 3.1. Data Augmentation:

The data augmentation process employed in this context encompasses a variety of transformations to enhance the diversity and robustness of the dataset. These transformations include rotation within a range of 90 degrees, adjustments to brightness levels within the specified range of 0.1 to 0.7, as well as shifts in both width and height with a maximum range of 0.5 units. Additionally, horizontal and vertical flipping are applied to the images, which further contributes to dataset diversity. It's worth noting that a validation split of 15% is also integrated into the process, allowing for the creation of a validation subset to assess model performance during training. Lastly, a pre-processing function labelled "pre-process_input" is incorporated. This strategy introduces a range of transformations and enhancements to the dataset, helping to mitigate overfitting and improve the generalization capabilities of machine learning models. Figure 4 presents the workflow of the augmentation process.

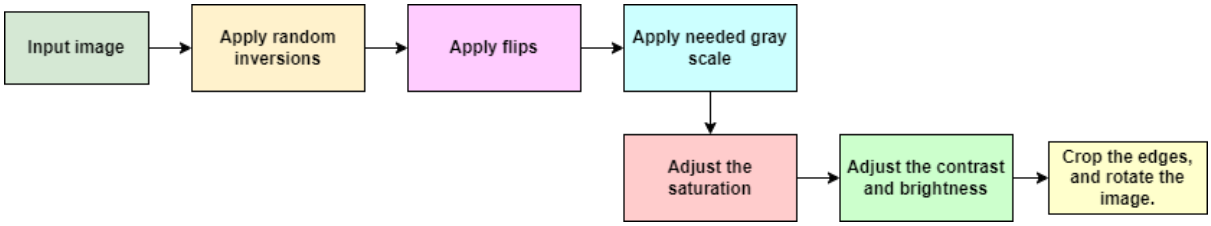


**Fig 4:** Augmentation Process Workflow

### 3.2. Feature extraction using VGG:

Feature extraction using the VGG architecture involves utilizing a pre-trained VGG model to extract meaningful features from images. These models have learned to recognize a wide range of visual patterns and objects in images during their training. In feature extraction, we typically remove the fully connected layers of the VGG model. After removing the top layers, the remaining part of the VGG model consists of convolutional and pooling layers. These layers are responsible for capturing low-level to high-level features in images. The feature maps extracted from the VGG model represent different levels of abstraction in the input image. While more sophisticated characteristics and object components are captured by higher levels, simpler elements like edges & textures are captured by lower layers. Depending on the specific layer from which you extract features, the dimensionality of the extracted features will vary. Deeper layers have more channels in their feature maps and capture more abstract information. Figure 5 exhibits the feature extraction process using VGG-16

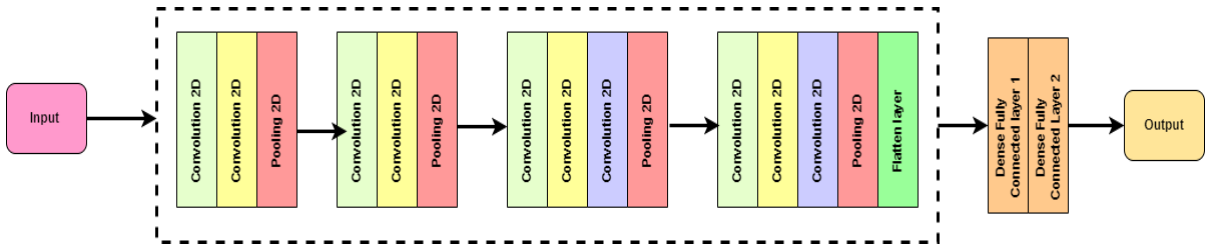


**Fig 5:** Feature Extraction using VGG-16

### 3.3. Hyper Tuning VGG:

The VGG model is a CNN that has been shown to be very effective for image classification tasks. There are many different ways to perform hyperparameter tuning for the VGG model. One common approach is to use a grid search. A gs is a thorough investigation of a hyperparameter space, where all possible combinations of hyperparameter values are evaluated. A random search randomly samples hyperparameter combinations from the hyperparameter space. Although it costs less to compute than a grid search, this method might not yield the optimal hyperparameter values. In Bayesian optimisation, the optimal hyperparameter values are found using a probabilistic model. Although it could be more effective than a grid search or random search, this calls for deeper understanding of the hyperparameter field beforehand. The model's learning rate determines how rapidly it learns. The quantity of epochs determines how frequently a model is going to be trained using the data. The batch size specifies how many data samples will be used to iteratively update the model's parameters. The dropout rate is the probability that a neuron will be dropped out during training. These are some hyperparameters which are tuned on the VGG methodologies. Table 3.1 discusses the parameters required for tuning the VGG-16

**Table 3.1:** Parameters for tuning the VGG as a table with description

| Parameter | Description |
|---|---|
| Learning Rate | It determines the step size for gradient updates during training. It's a critical hyperparameter, and a n appropriate value is crucial for convergence. |
| Batch Size | It defines the number of training examples processed in each forward and backward pass. |
| Number of Epochs | The number of training epochs specifies how many times the entire dataset is processed during training. |
| Weight Decay | In order to deter using excessive weight values, a penalty term known as L2 regularisation is added to the loss function. |
| Dropout Rate | It is a regularisation strategy that, during every training cycle, arbitrarily removes a subset of neurons. |
| Optimizer | It chooses the model load update method for training. The convergence behaviour and training efficiency may be impacted by optimizer selection. |
| Data Augmentation | It involves applying random transformations to training images. It helps the model generalize better by exposing it to a variety of input variations. |
| Model Architecture | Depending on the specific task, it need to modify the VGG design. This can include adding or removing layers, changing the number of output units in the final FC layer for classification tasks, or incorporating custom layers for specialized tasks. |

### 3.4. Proposed Model Customized Parameters:

The proposed model considers three important parameters namely learning rate, dropout rate, and optimizer. These parameters are tuned using the popular Grid Search approach because it is a transparent and deterministic method. It doesn't require knowledge of the model's underlying behavior and doesn't rely on probabilistic modeling.Grid search by training and evaluating the neural network on all possible combinations of hyperparameters from the grid. After evaluating all combinations, select the hyper parameters that resulted in the best performance according to the chosen evaluation metric. Table 3.2 determines the metrics that are customized in the proposed model of VGG-16

**Table 3.2:** Metrics of VGG-16 for Tuning

| Parameter Name | Importance | Need of Tuning | Possible Values |
|---|---|---|---|
| Learning Rate | The learning rate determines the step size at which the model's parameters are updated during training. | If the learning rate is too high, the model may fail to converge, while if it's too low, training may be excessively slow, and the model may get stuck in local minima. | 0.00001 to 1 with an interval of 10 |
| Dropout Rate | Dropout is a regularizatio n technique used to prevent overfitting by randomly dropping a fraction of neurons during training. The dropout rate determines the probability of dropping neurons. | If the dropout rate is too low, it may not provide effective regularizatio n, while if it's too high, it may hinder model convergence and learning. | 0.2 to 0.5 |
| Optimizer | The optimizer determines how the model's weights are updated based on the gradients of the loss function | The choice of optimizer can significantly impact the model's training speed and final performance | SGD, NAG, AdaDelta , RMSProp , ADAM, Adagrad |

## 4. Results & Discussion:

```
Epoch 1/10
51/51 [==============================] - 541s 10s/step - loss: 13.9829 - accuracy: 0.4948 - val_loss: 0.8705 - val_accuracy: 0.6875
Epoch 2/10
51/51 [==============================] - 42s 810ms/step - loss: 0.8956 - accuracy: 0.6466 - val_loss: 0.8894 - val_accuracy: 0.6389
Epoch 3/10
51/51 [==============================] - 40s 788ms/step - loss: 0.8182 - accuracy: 0.6822 - val_loss: 0.7810 - val_accuracy: 0.6979
Epoch 4/10
51/51 [==============================] - 45s 885ms/step - loss: 0.7390 - accuracy: 0.7124 - val_loss: 0.7976 - val_accuracy: 0.7014
Epoch 5/10
51/51 [==============================] - 42s 832ms/step - loss: 0.6532 - accuracy: 0.7333 - val_loss: 0.6298 - val_accuracy: 0.7535
Epoch 6/10
51/51 [==============================] - 42s 824ms/step - loss: 0.6044 - accuracy: 0.7769 - val_loss: 0.6932 - val_accuracy: 0.7500
Epoch 7/10
51/51 [==============================] - 42s 818ms/step - loss: 0.5757 - accuracy: 0.7744 - val_loss: 0.6564 - val_accuracy: 0.7465
Epoch 8/10
51/51 [==============================] - 40s 789ms/step - loss: 0.5471 - accuracy: 0.7837 - val_loss: 0.7260 - val_accuracy: 0.7049
Epoch 9/10
51/51 [==============================] - 40s 795ms/step - loss: 0.5501 - accuracy: 0.7923 - val_loss: 0.6746 - val_accuracy: 0.7500
Epoch 10/10
51/51 [==============================] - 40s 781ms/step - loss: 0.5354 - accuracy: 0.7990 - val_loss: 0.5938 - val_accuracy: 0.7500
```

**Fig 6:** Epochs on Training & Validation Accuracy

A hyperparameter called "epochs" determines how many times the learning method will run over the whole training dataset. At the end of one epoch, the parameters of the internal model have been updated for every collection in the training dataset. A piece of the dataset is used to train the neural network in a few batches that make up each epoch. We use the term "iteration" to describe the process of moving over a batch of training data. Figure 6 shows that the accuracy is quite low prior to tweaking the settings. Additionally, the loss values of the function are somewhat greater than anticipated.

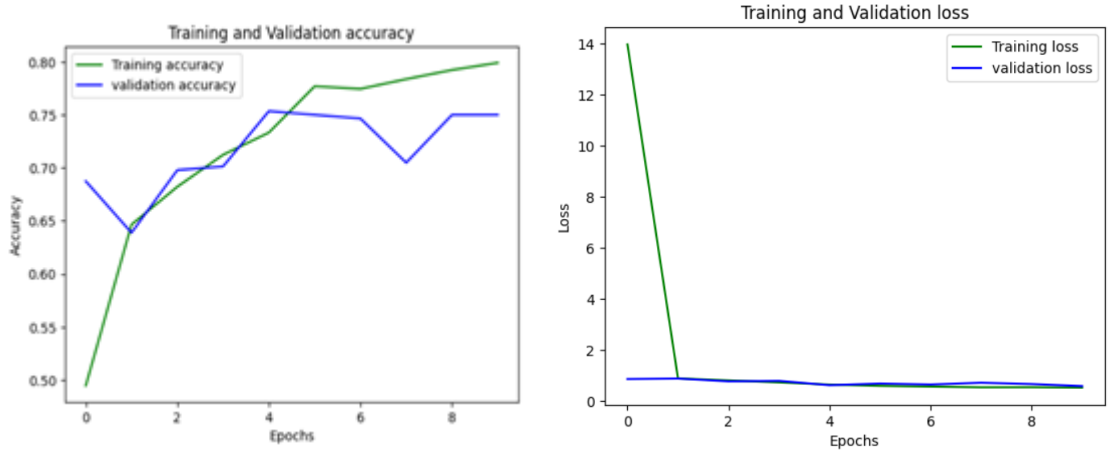


**Fig 7:** Accuracy & Loss of Traditional Approach

Learning curve graphs are widely used to determine network convergence for neural network model models. Usually, graphs showing loss vs. epoch or accuracy vs. epoch are displayed. We expect accuracy to increase and loss to decrease as the overall number of training epochs increases. However, we believe that accuracy and loss will eventually normalise. As usual, splitting the information set into training and validation sets is advised. Thus, we may generate progression graphs for various collections. These graphs let us to determine if the model has learnt too much, too little, or whether it matches the learning set. The training and validation accuracy according to epochs are shown in figure 7 above. It demonstrates that the validation accuracy has displayed oscillations while the training accuracy has steadily grown from a very low level. The training and validation loss curves are displayed against epochs in figure 7. By giving us a clearer knowledge of how learning performance fluctuates over the span of the number of epochs, these loss curves assist us in identifying any learning problems that may lead to an underfit or overfit model. The training loss decreases steadily, drops suddenly, and then continues to diminish until the last period. With slight variations, the validation loss has generally

```
Epoch 36/40
51/51 [==============================] - ETA: 0s - loss: 0.2262 - accuracy: 0.9213
Epoch 36: val_loss did not improve from 0.17775
51/51 [==============================] - 44s 854ms/step - loss: 0.2262 - accuracy: 0.9213 - val_loss: 0.2041 - val_accuracy: 0.9236
Epoch 37/40
51/51 [==============================] - ETA: 0s - loss: 0.2039 - accuracy: 0.9305
Epoch 37: val_loss did not improve from 0.17775
51/51 [==============================] - 38s 753ms/step - loss: 0.2039 - accuracy: 0.9305 - val_loss: 0.1981 - val_accuracy: 0.9306
Epoch 38/40
51/51 [==============================] - ETA: 0s - loss: 0.2042 - accuracy: 0.9250
Epoch 38: val_loss did not improve from 0.17775
51/51 [==============================] - 43s 836ms/step - loss: 0.2042 - accuracy: 0.9250 - val_loss: 0.3002 - val_accuracy: 0.8993
Epoch 39/40
51/51 [==============================] - ETA: 0s - loss: 0.1984 - accuracy: 0.9336
Epoch 39: val_loss did not improve from 0.17775
51/51 [==============================] - 42s 816ms/step - loss: 0.1984 - accuracy: 0.9336 - val_loss: 0.2195 - val_accuracy: 0.9306
Epoch 40/40
51/51 [==============================] - ETA: 0s - loss: 0.1969 - accuracy: 0.9398
Epoch 40: val_loss improved from 0.17775 to 0.16391, saving model to tl_model_v2.weights.best.hdf5
51/51 [==============================] - 57s 1s/step - loss: 0.1969 - accuracy: 0.9398 - val_loss: 0.1639 - val_accuracy: 0.9375
```

**Fig 8:** Accuracy of Proposed Model

The period training following tuning is shown in figure 8 above. The dataset's intrinsic perplexity determines the appropriate number of epochs. Using all of the training data, a neural network undergoes training for one cycle throughout an epoch. Throughout an era, each piece of knowledge is used precisely once. If a pass moves both forward and backward, it counts as one pass. Following VGG parameter adjustment, accuracy has risen. It increased to 93% in a row. Minimal values also applied to the loss. 40 epochs of training have been completed.

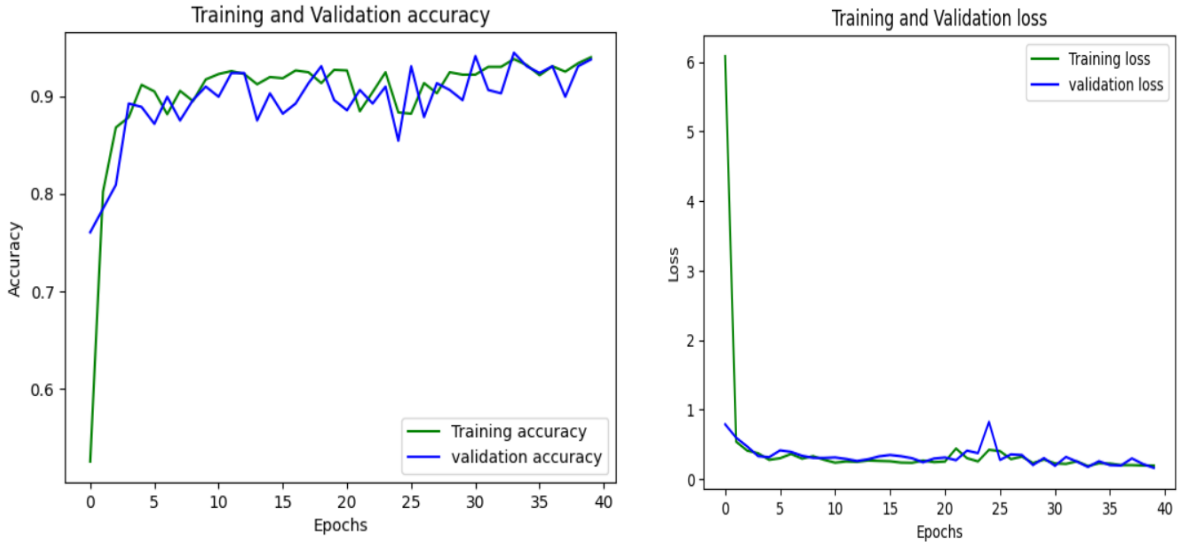


**Fig 9:** Graphical Analysis of Proposed Model

The learning curve derived from the training dataset provides a sense of how effectively the model is picking up new information. The validation learning curve, which is derived from a hold-out validation dataset, provides insight into how effectively the model generalises. The training loss curve drops sharply and continues to rise until the very final epoch. The graph shows that the proposed model's loss is relatively lower.

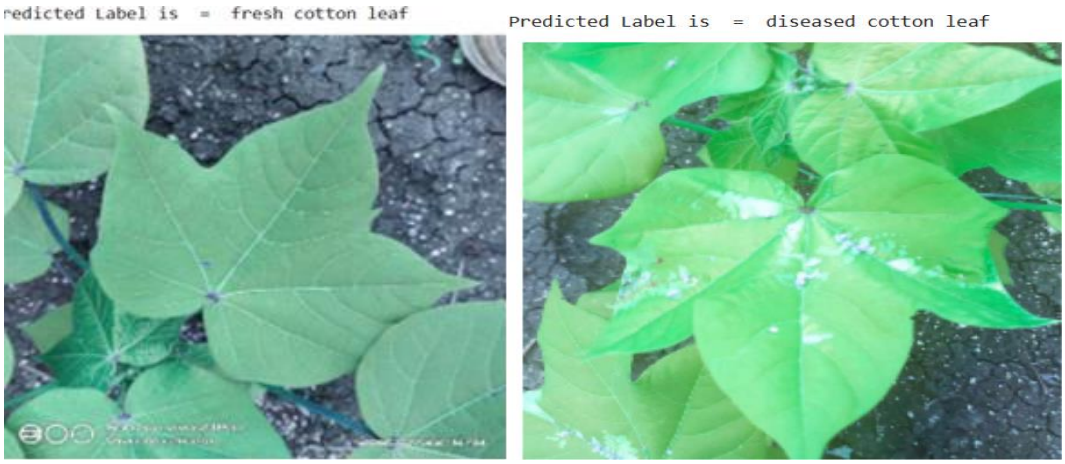


**Fig 10:** Prediction using Hyper Tuned Approach

The findings of the suggested VGG-16 model are depicted in the above images. In the images above, the model is applied to the non-diseased leaf image and successfully predicts the fresh cotton leaf. A sick leaf is shown in the second image so that the approach can analyse the characteristics and recognise the leaf and its illness. The class designation given is correct.

## 5. Conclusion:

In conclusion, the recommended method for detecting cotton leaf disease using deep learning models, in particular VGG-16, has demonstrated tremendous promise in addressing major difficulties in crop management and agriculture. With the use of pre-trained models, meta-learning techniques, and image-based sickness detection techniques, we were able to achieve a respectable accuracy rate of 93% after 51 epochs. By precisely targeting treatment on the affected areas, optimizing crop yield, and maintaining fiber quality while using fewer unnecessary chemicals, this technique lessens the impact on the environment. The use of VGG-16 for feature extraction and suitable hyperparameter tweaking with Grid Search has significantly improved disease classification. Future research may focus on enhancing the model's robustness through the use of more diverse datasets and the examination of transfer learning techniques to adapt the model to different crop diseases.

## References:

[1] Kumar, S., Ratan, R., & Desai, J. v. (2022). Cotton Disease Detection Using TensorFlow Machine Learning Technique. Advances in Multimedia, 2022. https://doi.org/10.1155/2022/1812025

[2] Ali, S., Hassan, M., Kim, J. Y., Farid, M. I., Sanaullah, M., & Mufti, H. (2022). FF-PCA-LDA: Intelligent Feature Fusion Based PCA-LDA Classification System for Plant Leaf Diseases. Applied Sciences (Switzerland), 12(7). https://doi.org/10.3390/app12073514

[3] Murugamani, C., Shitharth, S., Hemalatha, S., Kshirsagar, P. R., Riyazuddin, K., Naveed, Q. N., Islam, S., Mazher Ali, S. P., & Batu, A. (2022). Machine Learning Technique for Precision Agriculture Applications in 5G-Based Internet of Things. Wireless Communications and Mobile Computing, 2022. https://doi.org/10.1155/2022/6534238

[4] Goel, L., & Nagpal, J. (2023). A Systematic Review of Recent Machine Learning Techniques for Plant Disease Identification and Classification. In IETE Technical Review (Institution of Electronics and Telecommunication Engineers, India) (Vol. 40, Issue 3, pp. 423–439). Taylor and Francis Ltd. https://doi.org/10.1080/02564602.2022.2121772

[5] Elaraby, A., Hamdy, W., &Alruwaili, M. (2022). Optimization of deep learning model for plant disease detection using particle swarm optimizer. Computers, Materials and Continua, 71(2), 4019–4031. https://doi.org/10.32604/cmc.2022.022161

[6] Noon, S. K., Amjad, M., Qureshi, M. A., & Mannan, A. (2022). Handling Severity Levels of Multiple Co-Occurring Cotton Plant Diseases using Improved YOLOX Model. IEEE Access. https://doi.org/10.1109/ACCESS.2022.3232751

[7] Indumathy, K., &Devisuganya, S. (2023). Cotton Plant Disease Classification Using Data Mining Technique and Augmented Fast RCNN Algorithm. Journal of Survey in Fisheries Sciences, 10(3S), 6365-6378.

[8] Patil, B. M., &Burkpalli, V. (2022). Segmentation of cotton leaf images using a modified chanvese method. Multimedia Tools and Applications, 81(11), 15419–15437. https://doi.org/10.1007/s11042-022-12436-8