J. Integr. Sci. Technol. 2025, 13(3), 1063

Journal of Integrated
SCIENCE& TECHNOLOGY

# Development of an SDG-driven Convolutional Neural Network (CNN) model for multi-class classification of Tomato Leaf diseases using combined public and local datasets

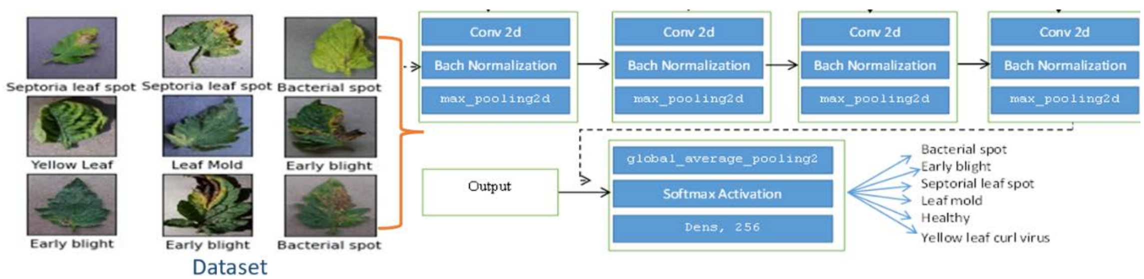
Mesfin Abebe,[1] Zewdu Tiumay,[1] Sudhir Kumar Mohapatra,[2] Srinivas Prasad,[3] Kumar Surjeet Chaudhury,[4] Prahallad Sahoo[5]

*[1]Adama Science and Technology University, Adama, Ethiopia, [2]Sri Sri University, Odisha, India. [3]GITAM University, Vishakhapatnam, Andhra Pradesh, India. [4]KIIT University, Bhubaneswar, Odisha, [5]GITA Autonomous College, Bhubaneswar, India.*

Submitted on: 17-Sep-2024, Accepted and Published on: 06-Jan-2025

Article

## ABSTRACT




Tomato plant is one of the most consumed vegetables globally. Tomato production has a great role in food security that secure the United Nations Sustainable Development Goals (SDGs). Early detection and classification of tomato leaf diseases is vital to mitigate yield loss. This study focuses on the development of SDG-driven Convolutional Neural Network model for the accurate multi-class classification of tomato leaf diseases using a combination of locally prepared and publicly available datasets. The combined data set is labeled with different classes such as: Bacterial Spot, Early Blight, Sectorial Leaf spot, Leaf mold, Yellow Leaf curl and healthy. The study developed custom made CNN, and VGG16 models applying various augmentation techniques, data splitting methods and hyperparameter tuning. The experimental results indicate that the custom-made CNN model outperform the pertained models with an accuracy of 99.8% using RGB image, augmentation techniques, 200 epochs, Adam, 0.0001 learning rate and a testing data set of 15% ratio. In conclusion, the study showed the use of AI-driven method to reduce the dependency on manual disease detection and improve the tomato production.

*Keywords: Sustainable Development Goals 2, SDG 2, Convolutional Neural Network, Tomato Leaf Diseases, Multi-Class Classification, Pretrained Models, Precision Agriculture.*

## INTRODUCTION

Tomatoes plant is one of the most widely consumed vegetable in the world after potatoes. Primarily, tomatoes are originated from Mexico and spread to Peru and the rest of the world.[1] Tomatoes are grown in various part of Ethiopian whichhave various agro-ecological conditions and it grows between the altitudes of 700 to 2000 meters above sea level.[2,3] The Ethiopian rift valley's areas are known for their high production of tomatoes, typically around the rift valley's lakes region and along side the Awash River valley. Although, tomatoes are grown throughout the year, they are highly grown in the winter and summer. The plant is not capable to survive a strong cold weather condition. In general, it may be cultivated at temperatures between 18°C and 27°C and it grow well at an average monthly temperatures range of 21°C to 23°C. The temperature and light intensity of the farm land has a significant effect on the pigmentation, fruit-sets, and nutritional value of the tomato fruits.[4]

Tomato fruits provide essential Minerals and Vitamins for human health life namely: vitamins C and E, B-carotene, lycopene, flavonoids, organic acids, phenolics, and chlorophyll. Furthermore, they are used to treat digestive disorders and purify blood with their therapeutic properties. As a result, recently scientists are studying the nutritional value of tomato in great detail.[5] In Ethiopia, tomato is one of the healthiest foods that is widely used for both home

*Sudhir Kumar Mohapatra, Faculty of Engineering & Technology, Sri Sri University, Cuttack, Odisha, India-754006
Tel: +91-9556878743; Email: sudhir.mohapatra@srisriuniversity.edu.in

Cite as: *J. Integr. Sci. Technol., 2025, 13(3), 1063.*
*URN:NBN:sciencein.jist.2025.v13.1063*
*DOI:10.62110/sciencein.jist.2025.v13.1063*

consumption and in the business centers. Ethiopia produces tomatoes for both domestic and international markets, including tomato paste, tomato juice, tomato catch-up, tomato soup, and whole peel-tomato. The tomato fruit international market and the market embraces the neighboring countries like Somalia and Djibouti.[6]


**Figure 1**: Sample tomato leaves withtheir diseases types

Figure 1 depicts the four most common diseases that usually affect the Ethiopian tomato plants. Ethiopian farmers are using their necked eyes to detect and identify the tomato diseases that requires knowledge and experience which is transferred from generation to generation. In contrast, just 1% of these farmers tried to get guidance from the domain experts since it is high expensive. This study propose a simplified approach of tomato disease detection using image processing and computer vision to support the farmers.[4]

In the southern Ethiopian regions near Gamo Gofa, the late blight tomato disease causes significant yield losses every year, ranging from 63.7 to 100%.[7-10] These tomato diseases turn into a severe problem because they can considerably reduce the quantity and quality of the tomato production. For example: the tomato disease losses a total estimated of $539.74 million in Georgia, USA in 2007. About 185 million USD of this total were used to control the diseases and the remaining sum represents the damage caused by the disease. Ethiopia's economy is dominated by agriculture sector, where the old-fashioned production methods are used to cultivate the land.[11-13] This has a significant effect on the agricultural production of the country. Thus, the detection and categorization of tomato leaf diseases is the main challenge for Ethiopian framers and the economy of the country.

Most plant diseases are caused by bacteria, fungi, viruses, and Chromista. Hence, it is critical to identify and diagnose the plant leaf to treat and to stop the spread of the diseases. Traditionally, domain expert knowledge serves as the basis for the diagnosis of various plant diseases. Recently, computer vision and machine learning[14] are started to use as the method for the detection of different plant characteristics in agricultural domain. As a result, it reduces the misunderstanding and assist the domain expert in plant diseases detection and classification, which brings a long-standing impact on the agricultural sector and community.

The Sustainable Development Goal 2 (SDG 2) aims to end hunger, achieve food security and improved nutrition, and promote sustainable agriculture. This study aligns with SDG 2 by addressing the significant agricultural challenge of tomato leaf diseases that can cause substantial production losses and threaten on food security. This study tries to enhance the ability of the Ethiopian farmers to detect and classify tomato leaf diseases effectively by develop a CNN-based machine learning model. This can lead to increase tomato yields, improve food availability, and greater economic stability for the farming communities. Furthermore, the use of latest technologies in the agricultural sector promotes sustainable practices, ensuring long-term productivity and resilience in food supply systems.

These days most of the countries are suffering with a significant economic-losses as a result of tomato diseases.[15-17] Therefore, a system that automatically detects these diseases is highly demanded to overcome the economic impact of the diseases. This automatic approach should work by taking the pictures of the plant leaves with the help of digital camera, so that deep learning and image processing techniques can be used to identify the disease type and to provide appropriate remedy. In general, this study tries to address the following research questions such as: which image processing techniques, which image analysis techniques and which deep learning algorithm.

## LITERATURE REVIEW

Usama Mokhtar[18] offers an effective technique to determine whether a tomato leaf is infected or healthy. The leaf image is initially pre-processed to minimize the backdrop by using the erosion technique to remove noise. The performance of a support vector machine (SVM) classifier is assessed using N-fold cross-validation technique and different kernel functions. The SVM classifier has achieved a 99.83% accuracy. However, this classification does not support the identification of the diseases' types. In contrast, different segmentation, feature extraction, and classification algorithms are presented by S.D. Khiladi.[19] Various segmentation algorithms, such as K-Means clustering, have been used to obtain the infected region and Artificial Neural Networks (ANN) is used to diagnose the diseases types.

A color-based method to identify infection on a tomato leaf, including early blight disease, was presented by Juan F. Molina et al.[20] The approach uses color descriptors to classify tomato leaves into different classes. According to the author, the color structure descriptor offered superior accuracy versus other techniques. Likewise, Usama Mokhtar[21] describes an image processing method for identifying tomato leaf disease. During the image acquisition stage, digital pictures of tomato leaves with two class of disease: early blight and powdery mildeware gathered. Several image enhancement methods are used such as: smoothness, noise removal, scaling, image isolation, and background removal. The author offered the Gabor wavelet transformation and the support vector machine to detect the class of the tomato disease.

Sachin D[22] discussed disease detection, where the initial stage of acquiring a digital image is pre-processing to eliminate noise. Besides, pre-processing equalizes the histogram to turn RGB images into grayscale images. A boundary and spot detection algorithms are used in image segmentation to locate leaf infection areas. A fuzzy rule-based method for classifying tomato leaf diseases based on color features was developed by K Muthukannan et al.[23,24] First, preprocessing with the gradient operator is employed to reduce noise or tiny functions in the image. In the final, two crucial features - the mean and standard deviation - are taken from a cropped image with various image size samples. They believe that that fuzzy rule-based classification performs well. The experimental results demonstrate that the suggested strategy requires minimal computational effort to detect leaf diseases.

A total of 54,306 images from the Plant Village website were trained using GoogleNet and AlexNet models by David Hughes et al. [25]. GoogleNet regularly outperforms AlexNet with a training accuracy of 99.35%. However, the accuracy drops to 31.4% when it is tested with images that were taken in a different setting from the images that were used to train the model. Mosisa Desalegn[26] authors employed a deep learning technique to detect infected leaves in wheat production. The study collected data from three Ethiopia agricultural research institutes: Kolomsa Agricultural Research Institute, Bishoftu Agricultural Research Institute, and Ambo Agricultural Research Institute. The study used CNN algorithm to classify the diseases and it achieve a high accuracy of 99.76. However, due to the storage of dataset, the study focuses on only two classes only: healthy and infected.

Generally, this study tried to address some of the gaps of the previous studies and to integrate locally prepared dataset. The locally collected dataset integrated with the publicly available data to ensure the consistency**,** diversity**,** and robustness of the classification model to overcome the limitation of the existing models that are trained on single data set collected from one site. Furthermore, the study considered five diseases types and healthy tomato images. The approach combines the SDG-driven goals into the development of the tomato disease detection model to improve food security and to reduce the impact of pesticide on the environment.

## METHODOLOGY

### 3.1 Image Acquisition and Dataset Preparation

The study requires the preparation of locally collected tomato dataset and combining it with the publicly available once to create more divers and representative dataset. The local image dataset is gathered from several tomato farms of the Ethiopia regions: Oromia and Amhara research institutes. The local dataset is captured using a digital camera and the different classes are labeled with the domain experts. The images are gathered following the standards and procedures of the publicly available data set to make it compatible with the local dataset during the integration process. The dataset has a total 7020 images that includes five tomato diseases which are common in Ethiopia: *Bacterial Spot*, *Early Blight*, *Healthy*, *Septoria Leaf Spot*, *Leaf mold* and *Yellow leaf curl* and healthy leaf as shown in Table 1.

**Table 1**: Tomato leaves images used for the model development

| ***Disease Type*** | ***Bacterial Spot*** | ***Early Blight*** | ***Healthy*** | ***Septorial Leaf spot*** | ***Leaf mold*** | ***Yellow Leaf curl*** |
|---|---|---|---|---|---|---|
| Public dataset | 950 | 950 | 950 | 950 | 950 | 950 |
| Local dataset | 220 | 220 | 220 | 220 | 220 | 220 |
| ***Total*** | ***1170*** | ***1170*** | ***1770*** | ***1170*** | ***1170*** | ***1172*** |

The image dataset includes tomato diseases that are commonly occurred and severely affect the country economy. Images with the same disease types are organized and stored into different folders to do the preprocessing task. This facilitates to easily integrate new images into the dataset in the future. The approach enables to continuously enhance and increase the size and the diversity of the data set. Figure 2 gives an example of the image dataset used in this study to show its diversity and arrangement.

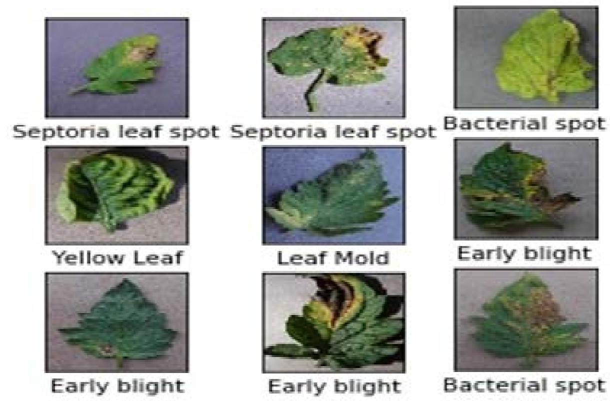



**Figure 1**: Sample images from the dataset

### 3.2 Image Preprocessing

The locally collected images and the Plant Village datasets have various image size of height and width. The images are resized to 256 by 256 pixels to bring all the images to equal size and reduce the computational and memory requirement of the training and the inference process. Since the CNN models require an input image of fixed size due to the architectural constraint. The dataset is normalized using Z-score normalization which subtract the mean from each pixel and dividing the results by the standard deviation that speeds up the convergence of the training process. Thus, each input parameter, or pixel maintained the same data distribution. Following the normalization, the dataset is augmented to increase the dataset size and to improve the model performance by simulating environmental variability. Data augmentation is the process of adding extra training data points to a dataset by producing additional data points from the existing dataset [27]. It avoids the over-fitting issue and aids in the network's capability to learn more sophisticated features from the data. Several data augmentation techniques are applied including rotation, zooming, flipping, shearing, and width and height shifting.

The infected part of the tomato plant is recognized using segmentation technique. In image processing, this leaf disease region is treated as an intersecting area. The infected area of the image is retrieved following the segmentation. In order to identify the different types of leaf disease, additional feathers are then removed based on the symptoms of the diseases. The study will discuss the most popular optimization procedure in the upcoming sections and why it is essential to optimize the model. The weights and biases will not update until the study figured out the gradient descent for the entire data set. The backpropagation stage will be extremely slow if the data set has thousands or hundred thousands of images in the dataset.

It is possible to update as soon as a small number of images have been utilized for learning, rather than waiting until all computations have been completed. Hence, the entire set of data can be divided into smaller segments, or mini-batches. Additionally, the gradient descent (GD) can be computed for a mini-batch at each interaction rather than the entire data set. Most importantly, we will update all weights and biases using the values gained from the single batch at each step, also known as an epoch. We have the stochastic gradient descent when we use just one image for each mini-batch.

The Stochastic gradient descent (SGD's) drawback is that it doesn't insurance to a global minimum; instead, there is a lot of noise in the cost variance over time, even while the average is falling and this in place of GD's standard edition. We can try a number of optimizations to increase the network's performance without significantly lowering its accuracy. Although there are many distinct optimization algorithms,[28,29] GD with momentum, RMS prop, and Adam are the most popular and the ones used in this study.

## Experimental results

CNN, a deep learning method, was selected based on a variety of computer vision research studies, particularly those that focused on image classification. CNNs have an intriguing approach to process an adaptive image. The algorithms are employed in training, testing, feature extraction, classification, and accuracy assessment of the model. Unlike traditional machine learning algorithms, CNNs use raw data directly without additional pre-processing or feature extraction step. Inside the CNN framework, the feature extraction and classification stages take place seamlessly.

The primary benefit of using the CNN method for tomato leaf disease recognition and classification is the end-to-end automation which brings better performance than classical machine learning algorithms.[30] The classical machine learning algorithm requires more handcrafted algorithms since it must construct different algorithms for different situations. However, the CNN model simplifies all the difficulties for the detection of fungus and bacteria in tomato plants.[31-35]

### 4.1 Tomato Leaves Diseases Recognition

Training and testing are the two major stages of the machine learning process that should be done to come up with the classification model. The CNN algorithm takes normalized images for the training and validation steps during the training phase. Additional images are created to help the CNN model to fit using the data augmentation methods that are mentioned above. The proposed approach of detection and classification is depicted in Figure 3.

Each model is built on the integrated dataset which include the public and local tomato leaf images that has six classes. The dataset was split into training, validation and testing sets to start the model building process. The model generates performance measures using the original validation data and the enhanced data for the training. The model building is the process of extracting important features from each image to classify them using the filtered-out features by the CNN model layers. Using the validation dataset, which is used to increase the model performance through model training process. Upon evaluating the model performance, the model with the highest accuracy is preserved and used as a predictive model. Then, the unseen data from the training phase is fed into the predictive model to conduct the testing phase. Finally, the model provides a class prediction or the likelihood level that an image corresponds to a particular class that was specified during training.

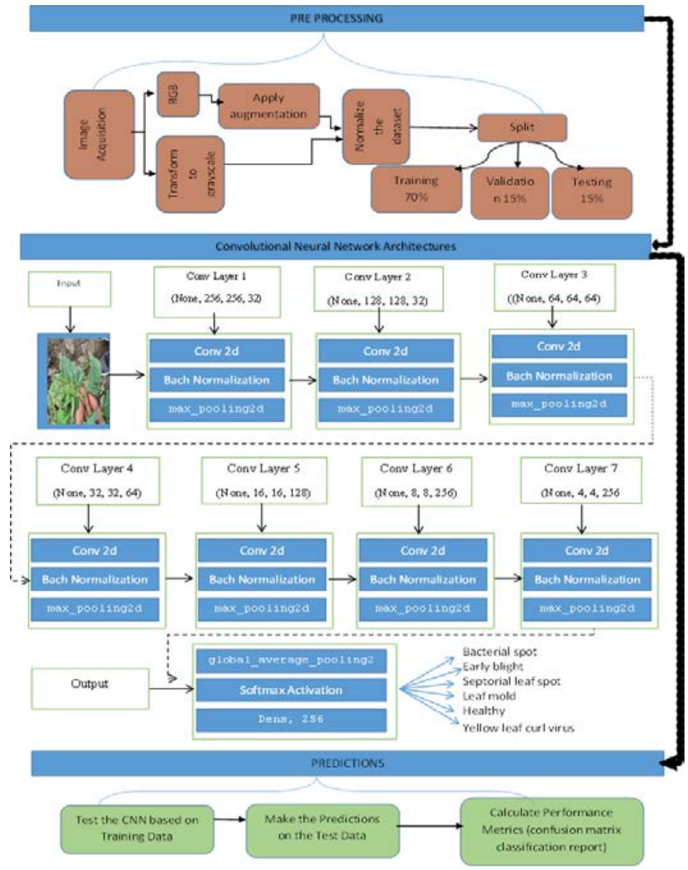



**Figure 3**: the Proposed approach CNN Model of tomato leaf disease classification

### 4.2 Classification with CNN and VGG16

CNN pre-trained models required large amounts of images to train, usually it involves thousands of classes in a large-scale image classification problem. The models have a stronger ability to generalize an object since they are trained on thousand and millions of images that need to be classified. Consequently, though the problem is distinct from the ones in the original challenge, the features that the models have learnt can be applied to a wide range of additional real-world problems. Transfer learning is the process of using once models that are trained on a different dataset to classify new classification problem. Instead of building a massive CNN model from scratch, anyone can apply these pre-trained models to train new model with their dataset.

It is sometimes impractical and computationally costly to train these designs from scratch due to limitation of computational resources. In this study, our dataset is used to refine and to re-train the pre-trained models such as: VGG16 beside to the custom-made CNN model. The majority of the models that our dataset trained on have two key components. Most pre-trained CNN models have two core components: a classifier which uses the extracted features by the convolution base to classify the input image and a convolution base that contains pooling and convolution layers for extracting important features from the input images. The convolution base, often referred to as a knowledge base that holds the features which are acquired during training. The convolution layers closest to the input capture general features that are common to all of the provided images. In contrast, convolution layers nearby the classifier only store model-specific characteristics and they have good generalization capability.

This section explain the application of the CNN algorithm to categorize the tomato plants leaf disease into different classes: bacterial spot, early blight, healthy, leaf mold, sectorial leaf spot, and yellow leaf. All the experimental details including each experiment's results and a brief discussion of them is presented hereafter.

### 4.3 VGG16 Pre-Trained Model

The VGG model is the simple in its design since it has three 3x3 convolution layers that are stacked one on top of another to increase the layer depth. This model comes in two forms: VGG19 and VGG16. The VGG19 includes 19 weight layers in the network. The model provides 1000 classes of ImageNet dataset that has a total of 14 million images. The model stacked convolutions layer which has a 3x3 receptive field to processes the input images. To reduce the spatial size of the convolution layers max-pooling of window size 2x2 is used after every three successive convolution layers. For the classification, there are three completely connected layers that are followed by a stack of the Conv layers. This is done using the VGG16 architecture and five max-pooling layers in both cases. With a SoftMax activation function, the last layer has 1000 channel depth, which is equal to the number of classes present in the ImageNet dataset, while the first two layers have 4096 channel depth.

In this study, a 127x127 down sampled RGB image is used to do the experiment. The model is fine-tuned to offer six different output to fit it with our dataset. This spatial dimension of the image minimizes model complexity to 15,894,849 parameters. The training is adjusted by using only the network's Conv base of VGG16 model due to the computational resource limitation. In multiple experiments, the model is trained various Conv blocks to determine the best pre-trained model. The model exhibits significant overfitting when it is trained using the entire network's Conv base with only the fully connected layer altered. Overfitting may occur since the model weights are trained on millions of images however the dataset has only 7020 images.

Consequently, data augmentation methods are used to increase the number of images and update few of the network's weights. In this study, we have enhanced the original data with augmentation for the experiment and freeze some of the model's layers (Conv blocks). The dataset increased to 56,160 images after applying eight different types of augmentation techniques. We conducted multiple experiments and found that freezing the first three Conv blocks is the best option in contrast to freezing the first two or the first fourth Conv blocks. The network is trained using different hyperparameters such as: epoch, batch size, activation function, loss function, optimization algorithm, and learning rate. The model accuracy achieved is indicated in Table 2 by using three different data splitting ratio.

Table 2 shows that the proposed model performs well with various ratios of training and testing dataset ratios. The results indicate that a ratio of 80% for training, 15% for validation, and 5% for testing yields better performance. When an 80:20 ratio is applied, 80% of the dataset is used for training and 5% for testing and the remaining 15% is for validation.

**Table 2**: Results of VGG16 pre-trained models using different data splitting ratios

| *Training Ratio* | *Test Ratio* | *Accuracy of the Model* | | | *Error rate of the Model* | | |
|---|---|---|---|---|---|---|---|
| | | *Training* | *Validation* | *Testing* | *Training* | *Validation* | *Testing* |
| **80** | **20** | **98.49 %** | **98.48 %** | **97.86 %** | **4.30 %** | **5.70%** | **6.25 %** |
| 70 | 30 | 93.80 % | 95.40% | 97.50 % | 20.40 % | 18.29% | 22.00 % |
| 60 | 40 | 95.30 % | 96.50% | 96.60 % | 14.10 % | 10.10% | 7.20 % |

The experiment employing various learning rates to arrive the outcome of the proposed model is shown in the next table using the accuracy metrics for the train, validation, and test data individually expressed as a percentage. The following result shows that larger learning rates are associated with lower accuracy when compared to smaller learning rates. Thus, under the suggested paradigm, a learning rate of 0.001 is considered ideal.

**Table 3**: the proposed VGG16 model with different learning rate

| *Learning Rate* | *Accuracy of the Model* | | | *Error rate of the Model* | | |
|---|---|---|---|---|---|---|
| | *Training* | *Validation* | *Testing* | *Training* | *Validation* | *Testing* |
| 0.1 | 94.60% | 91.40% | 74.70 % | 21.70% | 20.10% | 60.40 % |
| 0.01 | 94.15% | 95.42% | 97.78 % | 16.70% | 13.50% | 8.20% |
| **0.001** | **98.49 %** | **98.48%** | **97.86 %** | **4.30%** | **5.70%** | **6.25 %** |

The following table presents the experiment results for the proposed model using several activation functions. The classification accuracies are stated as a percentage for the train, validation, and test data separately as shown in Table 4.

**Table 4**: the proposed VGG16 model with different activation function

| *Activation Function* | *Accuracy of the Model* | | | *Error rate of the Model* | | |
|---|---|---|---|---|---|---|
| | *Training* | *Validation* | *Testing* | *Training* | *Validation* | *Testing* |
| SoftMax | 94.60 % | 95.20% | 97.10 % | 15.20 % | 12.30% | 8.60% |
| **Sigmoid** | **98.80 %** | **99.14 %** | **98.49 %** | **0.0004%** | **0.0057 %** | **0.0065%** |

### 4.4 Proposed CNN Model on Different Types of Images

The CNN model used for this study is a standard architecture consisting of multiple convolutional layers followed by fully connected layers as shown in Figure 3. The architecture is designed to extract important features from the input images and classify them into disease categories. Images with a single color channel, or grayscale, are unable to contain sufficient information to extract from the image data. The model is evaluated on the same data set by varying its parameters, which might impact the model's effectiveness and the learning rate. The learning rate 0.001 for the first result and 0.00001 for the second is the only characteristic that differs between the two results.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 256, 256, 32)      896
_________________________________________________________________
norm_1 (BatchNormalization)  (None, 256, 256, 32)      128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 128, 128, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 128, 128, 32)      9248
_________________________________________________________________
norm_2 (BatchNormalization)  (None, 128, 128, 32)      128
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 64, 64, 64)        51264
_________________________________________________________________
norm_3 (BatchNormalization)  (None, 64, 64, 64)        256
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 32, 32, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 32, 32, 64)        36928
_________________________________________________________________
norm_4 (BatchNormalization)  (None, 32, 32, 64)        256
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 16, 16, 128)       73856
_________________________________________________________________
norm_5 (BatchNormalization)  (None, 16, 16, 128)       512
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 8, 8, 128)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 8, 8, 256)         33024
_________________________________________________________________
norm_6 (BatchNormalization)  (None, 8, 8, 256)         1024
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 4, 4, 256)         0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 4, 4, 256)         590080
_________________________________________________________________
norm_7 (BatchNormalization)  (None, 4, 4, 256)         1024
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 2, 2, 256)         0
_________________________________________________________________
global_average_pooling2d_1   (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 1542
=================================================================
Total params: 800,166
Trainable params: 798,502
Non-trainable params: 1,664
```

These results have accuracy of 90.45% and 93.55%, correspondingly. This demonstrates that in result two, the learning rate, which was equal to 0.001, has dropped to 0.00001. When the learning rate drops, the model's accuracy drops in tandem.

**Table 5**: Summary of the proposed CNN model (TomdiseasesNet)

| ***No*** | ***Dataset type*** | ***Epochs*** | ***Learning Rate*** | ***Test Ratio*** | ***Accuracy*** | ***Loos*** |
|---|---|---|---|---|---|---|
| 1 | Grayscale | 50 | Adam(0.001) | 20% | 93.34% | 15.00% |
| | | 30 | Adam(0.00001) | 15% | 90.45% | 24.00% |
| | | 100 | Adam(0.001) | 5% | 94.25% | 14.00% |
| | | 200 | Adam(0.001) | 20% | 95.62% | 12.00% |
| 2 | RGB | 50 | Adam(0.001) | 20% | 95.81% | 0.026% |
| | | 30 | Adam(0.001) | 30% | 93.47% | 0.013% |
| | | 100 | Adam(0.0001) | 15% | 98.43% | 0.011% |
| | | 200 | Adam(0.0001) | 15% | 99.18% | 0.0003% |
| 3 | Augmented | **50** | Adam(0.001) | 20% | 98.81% | 0.006% |
| | | **30** | Adam(0.001) | 30% | 99.11% | 0.0043% |
| | | **100** | Adam(0.0001) | 15% | 99.43% | 0.0041% |
| | | **200** | **Adam(0.0001)** | **15%** | **99.88%** | **0.0003%** |

The best results are obtained from testing the suggested model on RGB images using various learning parameters are shown in the above table with their impact on the model. As shown in the table, changing any parameter will have an impact on the accuracy and training time required for the model to produce the desired results.

## Discussion

Figure 4 and Figure 5 show the classification accuracy and loss graphs loss of the VGG16 pre-trained model that we trained with by making some changes to the original pre-trained model to allow the model to classify well in our dataset. The training accuracy is roughly 88% in the first epoch and gradually climbs to 95% by the tenth epoch. Between epochs 10 and 20, the training accuracy of the model exceeds 97%. The graph shows that the accuracy increases in the first few epochs which is due to the dataset. In general, as seen in the accompany figures, the validation accuracy line is nearly parallel to the training accuracy line, while the validation loss line is also parallel to the training loss. Although validation accuracy and validation loss are lowering, they are not growing, and validation accuracy is increasing rather than falling.

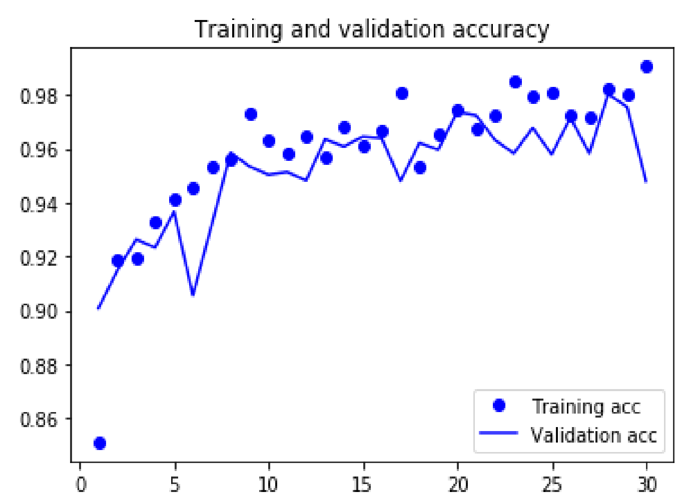



**Figure 4**: Training and validation accuracy for VGG16

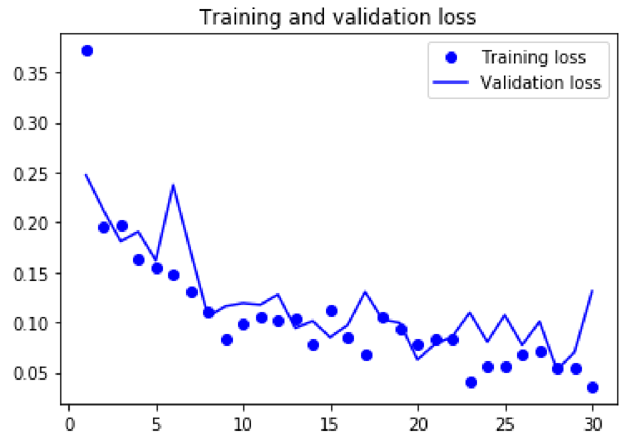



**Figure 5**: Training and validation loss for VGG16

At the beginning of the training, the training accuracy line's value is 90%, and the validation accuracy line's value is approximately 91%. Both values then increase until the fifth epoch. Both lines cross 99.18% after epoch 5 and climb extremely slowly. From the first epoch to 200thepoch, the training loss and validation loss curves in Figure 5 plot both drop linearly from 0.20 to 0.0250. Following the 10th epoch, the training loss line and validation both passed 0.050, which is the lowest result in the training and drastically drops from the start, but failed to pass 0.025, the smallest value.

The validation accuracy and validation loss are finally in line with the training accuracy and training loss, respectively. On the other hand, the training loss and validation loss curves are almost linear. The training accuracy and validation loss curves are almost linear. The validation curves indicate that the proposed model does not exhibit overfitting, as evidenced by the growing validation accuracy and decreasing validation loss shown in Figure 6 and Figure 7. Moreover, and perhaps most crucially, there is no significant difference between training and validation accuracy or between training and validation loss. Since the validation set's loss is only marginally more than the training set's loss, we can thus conclude that our model's generalization ability has improved significantly.

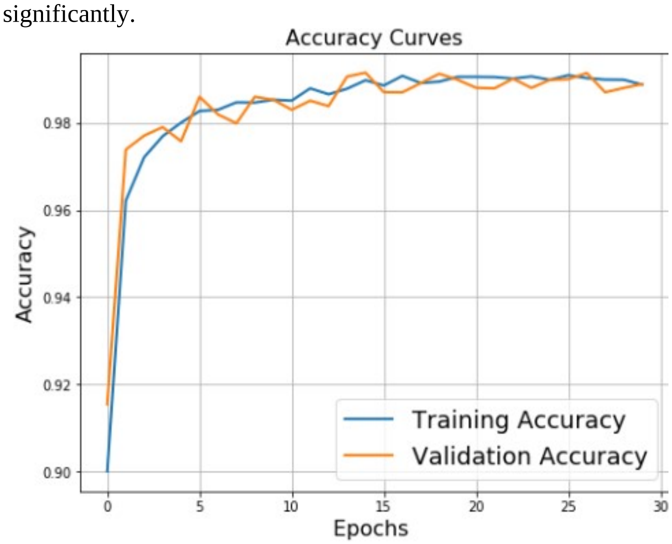



**Figure 6**: Training and validation accuracy for the proposed CNN Model

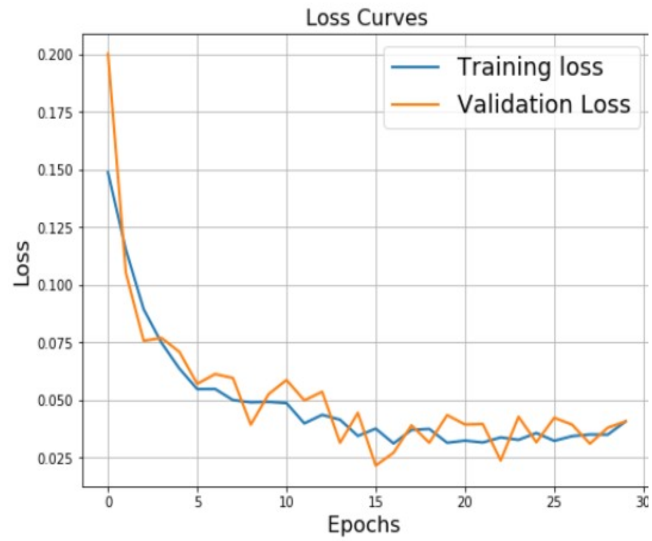



**Figure 7**: Training and validation loss for the proposed CNN Model

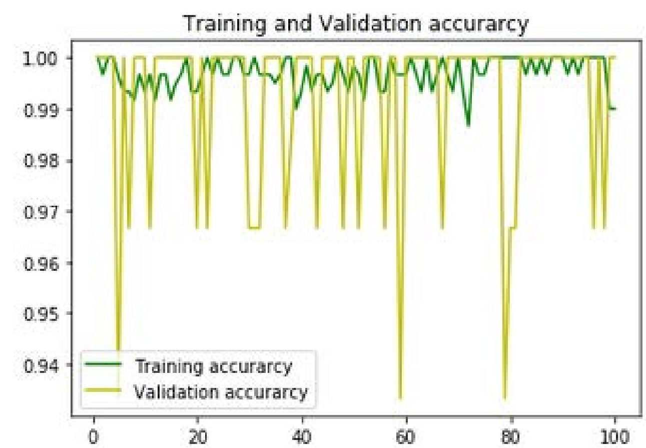



**Figure 8**: Training and validation accuracy with RGB image

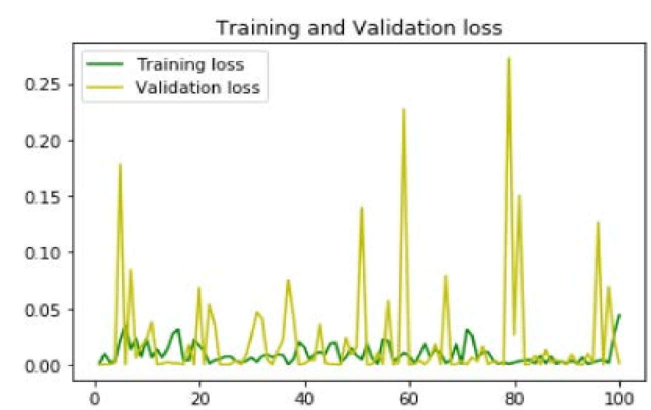



**Figure 9**: Training and validation loss with RGB image

Figure 8 and Figure 9 displays the outcome of the proposed approach experiment utilizing the classification accuracy metrics for the train, validation, and test data independently expressed as a percentage. The confusion matrix above demonstrates how well the model performs when tested using the previously trained or known pictures. However, the model misclassified three yellow leaves, one sectorial leaf spot, and two early blight images out of the total of 1595 test data points. These six misclassified images are part of the testing data.

The Loss chart demonstrates that the model loss30% at the beginning and progressively decreases to practically zero. On the other side, the accuracy increased slowly to 99.97%. The training and validation data do not significantly differ from one another in either graph.

## CONCLUSION AND FUTURE WORK

These days tomato infection diseases such as: tomato bacterial spot, tomato early blight, tomato sectorial leaf spot, tomato leaf mold, and tomato yellow leaf curl cause some problems for tomato production. These diseases can cause a huge reduction in production of unimproved local cultivars and tomato yield quality. Furthermore, the dearth of diagnostic instruments severely hinders the development and standard of living of people in underdeveloped nations like Ethiopia. As a result, it is critical to identify the illness early on using accessible and reasonably priced technology. We have developed and put into practice a deep learning strategy that uses an image processing algorithm and a convolutional neural network to enable early disease diagnosis. Using tomato leaf images as an input, we have proposed a TomdiseasesNet model to identify and categorize various tomato leaf illnesses and healthy leaves.

The study considered several significant aspects that may have an impact on the architecture of CNN model to come up with suitable solution. In order to run the experiments for the TomdiseasesNet model, the following three types of enhanced datasets are used: RGB, grayscale, and augmented with various augmentation techniques like brightness range, fill mode, rotation, zoom, width shift, height shift, shear range, and vertical and horizontal flips. TomdiseasesNet model's accuracy is 95.62%. 0.0001 learning rate with a data set of grayscale images. When the model is trained on the RGB data set, the outcome improves. After the enhanced image dataset is archived, it finally climbed to an accuracy of 99.18%, showing better performance than the other archived 99.68% with 200 epochs and a learning rate of 0.0001. The development of deep learning models for tomato disease detection directly supports several Sustainable Development Goals(SDGs),mainly SDG 2: Zero Hunger and SDG 12: Responsible Consumption and Production. By allowing early and precise detection of diseases in tomato crops, the models help smallholder farmers mitigate crop losses and enhance food security.

In order to advance this study, particularly in Ethiopia, it is imperative that farmers, investors, and both public and private institutions be involved in assisting Ethiopia's agricultural industry in transitioning to a modern agricultural period. In future, the farmers, investors, governmental organizations, or private businesses will have easy access to assistance in building and deploying a deep learning model onto a web server using Django, Flask, and other frameworks. Additionally, a single leaf may include several disease type; in this case, how would one categorize such a leaf should be addressed in the future. The neural network system can be further enhanced to account for increased environmental diversity and dataset scalability. More images of various tomato plant diseases need to be gathered for future study to make the model more Universal that local. Furthermore, it need to make the model explainable and small size (optimized) to deploy it on Edge devices in the future studies.

## REFERENCES


1. L. D. Research experiences in tomato production; Ethiopia Agricultural Research Organization, Addis Ababa.
2. T. Balemi. Response of tomato cultivars differing in growth habit to nitrogen and phosphorus fertilizers and spacing on vertisol in Ethiopia. *Acta Agric. Slov.* 2008, 91 (1), 103–119.
3. K. Bekele, H. Yaynu. Tuber yield loss assessment of potato cultivars with different levels of resistance to late blight. In *In: Eshetu Bekele E, Abdulahi A, Yemane A (Eds.) Proceedings, 3rd Annual Conference of Crop Protection Society of Ethiopia.*; Crop Protection Society of Ethiopia, Addis Ababa, Ethiopia, 1996; Vol. 7, pp 1145–1150.
4. M. Worku, S. Sahe. Review on Disease Management Practice of Tomato Wilt Caused Fusarium oxysporum in Case of Ethiopia. *J. Plant Pathol. Microbiol.* 2018, 9 (11).
5. D. Regassa, W. Tigre, A.S.-J. of P. Breeding, U. 2016. Tomato (Lycopersicon esculentum Mill.) varieties evaluation in Borana zone, Yabello district, southern Ethiopia. *Plant Breed. Crop Sci.* **2016**, 8 (10), 206–210..
6. A. Masho, M. Alemayehu, G. Alemayehu. Performance evaluation of tomato varaieties for irrigation production system in Mecha District of west Gojiam Zone, Amhara Region, Ethiopia. *Proceeding of the 3rd Annul National Conference on Agriculture and Environmental Management for Sustainable Development*. Amhara Region, Ethiopia 2016, pp 142 – 157.
7. A.D. Mengistu, D.M. Alemayehu, S.G. Mengistu. Ethiopian Coffee Plant Diseases Recognition Based on Imaging and Machine Learning Techniques. *Int. J. Database Theory Appl.* 2016, 9 (4), 79–88.
8. K.R. Gavhale, U. Gawande, K.O. Hajari. Unhealthy region of citrus leaf detection using image processing techniques. In *2014 International Conference for Convergence of Technology, I2CT 2014*; 2014.
9. K. Khairnar, R. Dagade. Disease Detection and Diagnosis on Plant using Image Processing A Review. *Int. J. Comput. Appl.* **2014**, 108 (13), 36–38.
10. T.N. Tete, S. Kamlu. Plant Disease Detection Using Different Algorithms. In *Proceedings of the Second International Conference on Research in Intelligent and Computing in Engineering*; **2017**; Vol. 10, pp 103–106.
11. D. Kaur, Y.K. Various image segmentation techniques: a review. *Int. J. Comput. Sci. Mob. Comput.* **2014**, 3 (5), 809–814.
12. N.M. Zaitoun, M.J. Aqel. Survey on Image Segmentation Techniques. *Procedia Comput. Sci.* **2015**, 65, 797–806.
13. A.U. Ruby, B.N. Chaithanya, T.J. Swasthika Jain, et al. An effective feature descriptor method to classify plant leaf diseases using Extreme Gradient Boost. *J. Integr. Sci. Technol.* **2022**, 10 (1), 44–52.
14. Y.A. Thakare, K.H. Walse. A review of Deep learning image captioning approaches. *J. Integr. Sci. Technol.* **2024**, 12 (1), 712.
15. G. Kumar, P.K. Bhatia. A detailed review of feature extraction in image processing systems. In *International Conference on Advanced Computing and Communication Technologies, ACCT*; 2014; pp 5–12.
16. Y. LeCun, Y. Bengio, G. Hinton. Deep learning. *Nature* **2015**, 521 (7553), 436–444.
17. H. Durmus, E.O. Gunes, M. Kirci. Disease detection on the leaves of the tomato plants by using deep learning. In *2017 6th International Conference on Agro-Geoinformatics, Agro-Geoinformatics 2017*; IEEE, 2017; pp 1–5.
18. U. Mokhtar, N. El Bendary, A.E. Hassenian, et al. SVM-Based Detection of Tomato Leaves Diseases; Springer, **2015**; pp 641–652.
19. A. Mishra, P. Chaurasia, V. Arya, F.J.G. Peñalvo. Plant Disease Detection using Image Processing. In *Lecture Notes in Networks and Systems*; IEEE, **2023**; Vol. 599 LNNS, pp 227–235.
20. J.F. Molina, R. Gil, C. Bojacá, F. Gómez, H. Franco. Automatic detection of early blight infection on tomato crops using a color based classification strategy. In *2014 19th Symposium on Image, Signal Processing and Artificial Vision, STSIVA 2014*; IEEE, 2015; pp 1–5.
21. U. Mokhtar, M.A.S. Ali, A.E. Hassenian, H. Hefny. Tomato leaves diseases detection approach based on Support Vector Machines. In *2015 11th*

*International Computer Engineering Conference: Today Information Society What's Next?, ICENCO 2015*; 2016; pp 246–250.
22. S.D. Khirade, A.B. Patil. Plant Disease Detection Using Image Processing. In *2015 International Conference on Computing Communication Control and Automation*; IEEE, **2015**; Vol. 599 LNNS, pp 768–771.
23. L.P. Muthukannan. Fuzzy inference system-based unhealthy region classification in plant leaf images. *Comput. Inf. Eng.* **2016**, 2103–2107.
24. P. Mitkal, P. Pawar, M. Nagane, et al. Leaf Disease Detection and Prevention Using Image Processing using Matlab. *Int. J. Recent Trends Eng. Res.* **2016**, 2 (2), 26–30.
25. S.P. Mohanty, D.P. Hughes, M. Salathé. Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **2016**, 7 (September), 1419.
26. M. Dessalegn. Applying a Deep Learning Approach for Wheat Rust Disease Detection. *Agric. Res. Rev.* **2019**, 99 (September), 1–70.
27. N. Ketkar, J. Moolayil. Deep Learning with Python; Manning Publications, **2021**.
28. S. Ruder. An overview of gradient descent optimization algorithms. **2016**.
29. H. Rajoriya, R. Sadiwala. Deep compressive sensing and reconstruction algorithm in wireless Internet of Things. *J. Integr. Sci. Technol.* **2023**, 11 (2), 487.
30. R. Ray Chaudhari, S. Jain, S. Gupta. Agricultural Machine learning platform: Enhancing crop suggestion and crop yield estimates. *J. Integr. Sci. Technol.* **2025**, 13 (1), 1017.
31. K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. 2015.
32. S.K. Mohapatra, A. Belete, A. Hussen, et al. Coffee leaf diseases classification using deep learning approach. In *Machine Learning Algorithms Using Scikit and TensorFlow Environments*; IGI Global, **2023**; pp 91–111.
33. N.T. Sinshaw, B.G. Assefa, S.K. Mohapatra. Transfer Learning and Data Augmentation Based CNN Model for Potato Late Blight Disease Detection. In *2021 International Conference on Information and Communication Technology for Development for Africa, ICT4DA 2021*; IEEE, **2021**; pp 30–35.
34. S.K. Mohapatra, S. Prasad, S.C. Nayak. Wheat Rust Disease Detection Using Deep Learning. *Data Sci. Data Anal.* **2021**, 191, 191–202.
35. N.T. Sinshaw, B.G. Assefa, S.K. Mohapatra, A.M. Beyene. Applications of Computer Vision on Automatic Potato Plant Disease Detection: A Systematic Literature Review. *Comput. Intell. Neurosci.* **2022**, 2022 (1), 7186687.