National College of Ireland

Plant Disease Detection on Wheat Plant using Deep Learning

MSc Research Project
Programme Name

Sayali Patil
Student ID: X20208162

School of Computing
National College of Ireland

Supervisor: Prashanth Nayak

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sayali Patil
Student ID:	x20208162
Programme:	MSc in Data Analytics **Year:** 2021- 2022
Module:	Research Project
Supervisor:	Prashanth Nayak
Submission Due Date:	15th August 2022
Project Title:	Plant Disease Detection on Wheat Plant Using Deep Learning
Word Count:	5193 **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: .....

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):	

Plant Disease Detection on Wheat Plant Using Deep Learning

Sayali Patil
X20208162

Abstract

Agriculture is a substantial source of revenue and a contributor to the national economy in several parts of the world. One of this industry's primary concerns is eliminating or reducing plant diseases. Diseases not only affect crop quality but also costs farmers money. Moreover, farmers occasionally need to consult professionals for advice, which is quite expensive. Therefore, to maintain crop quality and quantity, it is vital to diagnose the disease earlier to reduce pesticide use that affects the crop and the environment. The issues must be addressed by implementing innovative farming practices to aid in effective resolution. Recent advancements in machine learning techniques have significantly boosted plant disease detection studies. The current study aims to develop more precise methods for diagnosing and classifying wheat plant diseases. My experiments on the detection of plant disease using EfficientNetB0, a deep learning model show improved results in detecting plant diseases.

1 Introduction

Agriculture is essential because of the population's fast expansion and rising food consumption. Out of all the crops grown in the world, wheat is the most important crop and a source of food for people. (Genaev et al.,2021). Additionally, the wheat crop supplies 20% of the world's protein and nutritional goods (Singh, A. & Arora, M., 2020). Also, most planting areas are for wheat grain, and its production determines the population's food security in most countries

A good harvest benefits the farmers and contributes significantly to a country's economy. However, due to climatic changes, diseases, pests, and soil infertility, the crops are impacted; hence, it is very challenging to have high-yield production. More than 30% of crops are destroyed due to diseases (Genaev et al.,2021). Some of the common diseases affecting the wheat crop include crown and root rot, leaf rust, and wheat loose smut. Many researchers are currently focusing on and investigating to discover the best solution to these problems.

Digital revolution and the advancement of machine learning techniques have lately gained popularity in the diagnosis of plant diseases. Traditionally, experts try to identify the disease with their naked eyes based on their knowledge and experience. This activity, however, was time demanding and included a substantial risk of individual interpretation. The time-consuming process can be simplified by using

machine learning to develop an automated system for disease detection. Several machines learning algorithms, including SVM, Decision Tree, and Random Forest, have been utilized to successfully extract features and classify diseases. Rapid advances in research in this field have resulted in the development of advanced deep learning models.

The research aims to build a transfer learning model with trainable layers that can be combined with image augmentation to achieve a high accuracy level. The main contribution of the research is as follows:

- A deep learning model is utilized to accurately identify wheat plant diseases.
- To develop an EfficiNetB0 model and fine-tune the model using augmentation strategy along with multistage fine-tuning and identify the best-fitted model to provide solutions to the agriculture industry.

The remaining sections of the paper are structured as follows. The second section consists of related work, critically analyzing the existing system and the necessity for the current study. The third section describes the methodology adopted for this research. The fourth and fifth sections discuss the system's architecture and the steps necessary for successfully implementing deep learning models. The sixth section presents the evaluation and critical analysis of the implemented models in the project. Finally, the seventh section concludes the research with future scope.

1.1 Research Question

How effectively can a transfer learning model, EfficientNetB0, detect the disease in wheat plants?

This research focuses on how accurately the EfficientNetB0 model can identify the wheat plant's diseases.

2 Related Work

Many techniques for disease diagnosis utilizing Deep Learning and Machine Learning have been developed throughout the years. Several advanced versions of feature extraction and deep learning models were implemented. The studies conducted in this field, as well as its advantages and drawbacks, are highlighted

2.1 Detection of diseases using feature extraction and segmentation

There are many approaches used for the detection of diseases in plants. Many researchers have used CNN models and different segmentation and feature extraction methods to improve the model's performance for classifying plant diseases. C. K.

(2022) proposed a way to predict disease in cardamom plants. In this paper, the author used U2-Net (U square net) to eliminate the undesirable background from the image by extracting the main features. The proposed methodology was compared with the CNN and EfficientNet models, and it was found that the EfficientNetV2 model works best for the prediction. A combination of U2-Net and EfficientNet performed better for accurate time disease detection. In an extension of the above paper, the author Lee. (2017) used a hybrid feature extraction method to extract the features from lower to higher levels. The CNN model was utilized for the initial extraction from leaf images, and these images were further quantified using the D.N. technique. The experiment demonstrated that the hierarchical method retrieved the features more appropriately. The results obtained were satisfactorily good. However, this process requires millions of images and a more profound architecture for better results. The study by Panchal (2021) proposes a method for classifying diseases using leaves by performing image segmentation and feature extraction methods on the image. Several deep learning models were implemented on the PlantVillage dataset, and their performance was compared. CNN's poor performance was improved through the use of the transfer learning technique.

2.2 Reviews of the existing system for the detection of diseases

The study by Shruthi et al. (2019) reviews various algorithms in the existing system to detect diseases. Algorithms like SVM, KNN, ANN, Fuzzy classifier, and Deep Learning, a comparative study on these classification algorithms were discussed. The CNN model detected more diseases with greater accuracy than the others. In an extension of the above paper, the author Kartikeyan, P and Shrivastava G. (2021) elaborates briefly on the five significant steps to be taken for disease detection and algorithms with appropriate usage of graphical representation. Additionally, the author conducted a thorough study of the algorithms for better understanding, accompanied by comparative analysis. They concluded that the SVM works better than ANN and KNN. However, external software is required for feature extraction to achieve good prediction accuracy using SVM and KNN, which is time-consuming and tedious. In contrast to the above paper, Raina S. and Gupta A. (2021) discusses the various existing deep learning models and segmentation methods implemented for disease detection by the researchers. Though CNN requires less human effort for pre-processing the data, it requires a huge dataset for processing and training the neural network. Also, the author critically analyses all the models by describing their advantages and disadvantages. Poornappriya T.S, and Gopinath R. (2022), studied the identification of rice plant diseases using deep learning, image processing, and deep learning methods. The author discusses various image processing techniques for improving image quality. A lot of successful models were implemented to detect plant diseases. However, the author illustrates the gap remained despite several significant developments. Deep learning and machine learning models need to be advanced to detect plant diseases during the complete cycle of diseases and predict the disease in all field conditions.

2.3 Deep Learning Methods

A technique to identify five fungi-related diseases, including stem rust, leaf rust, powdery mildew, Septoria, and yellow rust was proposed by Genaev et al. (2021). The dataset was created using data redundancy techniques based on the image hashing technique. Then, the most efficient model was selected by finding the EfficientNetB0's optimum parameters during training. A high level of accuracy was attained by augmenting the data and implementing the transfer image style. Furthermore, a chatbot on the Telegram messenger was created to help users identify the disease in the wheat plant. This paper is an improvement of the above; a novel approach was proposed by (Goyal et al. 2021). This methodology uses the CNN model, where the weights in the model were modified during the backward run. The dataset used in the model has 12000 images, including ten types of diseases occurring on the wheat plant. According to the author, the proposed methodology gave better results than the transfer learning models like VGG16 (Deep learning model) and ResNet50 (Deep learning model) by 7 % and 15.92%, respectively. However, achieving good results using the CNN model requires a vast dataset. The study by Zhou et al. (2021) developed a more accurate deep learning model for detecting tomato leaf disease. The author used Residual Block and DenseNet to create the model. The images were normalized, and the CNN used a dense residual layer. The model, however, can only detect disease in tomato plants. The study by Islam et al. (2021) has focused on the prediction of whether a person has diabetes or not using images of the retina. The research revealed that retinal images provided more information for diabetes prognostic indicators than clinical data. The novel approach of using the multistage fine-tuning model was utilized for detecting the diseases and achieved an accuracy of 84%. The multistage fine-tuning technique can be used to detect plant diseases where the model can be trained incrementally to accurately detect the wheat plant's disease.

2.4 Machine Learning model

Methods for rice leaf disease detection are implemented by Ahmed et al. (2019). In this paper, the author has used machine learning algorithms like KNN, Decision trees, NaïveBayes, and Logistic Regression. Image filtering and correlation approaches were utilized for extracting the relevant image features. The accuracy of the model was improved by the cross-validation technique. After evaluating the results, it was determined that the decision tree achieved the best accuracy of 97%. In Ramesh et al. (2018), the author used Random Forest to detect the disease. Before training, the image features were extracted using the Histogram of an Oriented Gradient. The model was trained on a small dataset of 160 papaya leaf images and achieved 70% accuracy. However, the accuracy could have improved by augmenting the data or having more images. Similarly, Ganatra N. and Patel A. (2020) used the Random Forest model for leaf disease detection. The features extracted from leaves were color, texture, shape, and vein to improve prediction accuracy. When compared to other machine learning algorithms, random forest outperformed the others on the dataset. Kulkarni (2021)

proposed a method using Random Forest to detect the 20 diseases of 5 plants. The accuracy achieved by the model was 93%. The shape features were extracted using Morphological transform, and color features were extracted using bitwise and operation. After reviewing the paper, it was seen that Random Forest and Decision Tree gave better accuracy than other machine learning algorithms for disease detection.

2.5 Conclusion

Extensive research has been conducted on the application of machine learning and deep learning for identifying plant diseases. The current study intends to solve the inadequacies of previous research in this field by utilizing transfer learning models and data augmentation. Out of many EfficentNet models, the EfficienNetB0 model will be used in this study, along with multistage fine-tuning of the model, to identify diseases affecting wheat plants. The EfficientNet models outperform the other transfer learning models discovered after rigorously analyzing the publications. The study will assist in enhancing the model's effectiveness for detecting the diseases of wheat plants by utilizing multistage fine-tuning used for diagnosing diabetes.

3 Research Methodology

It is critical to select an appropriate methodology for the project before commencing the implementation to get good outcomes. Knowledge Discovery in Databases (KDD) is a technique for extracting knowledge from data by studying and predicting the data. This approach will be used in the present study to identify diseases in wheat plants. The data analysis procedure splits the project into eight stages. In the first stage, the data will be selected according to the project requirements: wheat plant images. After that, the data will be imported and pre-processed by scaling down the image size. The data will be used for training the model, and the performance of the model was evaluated on the test. In this section, each component of the KDD process is explained briefly.

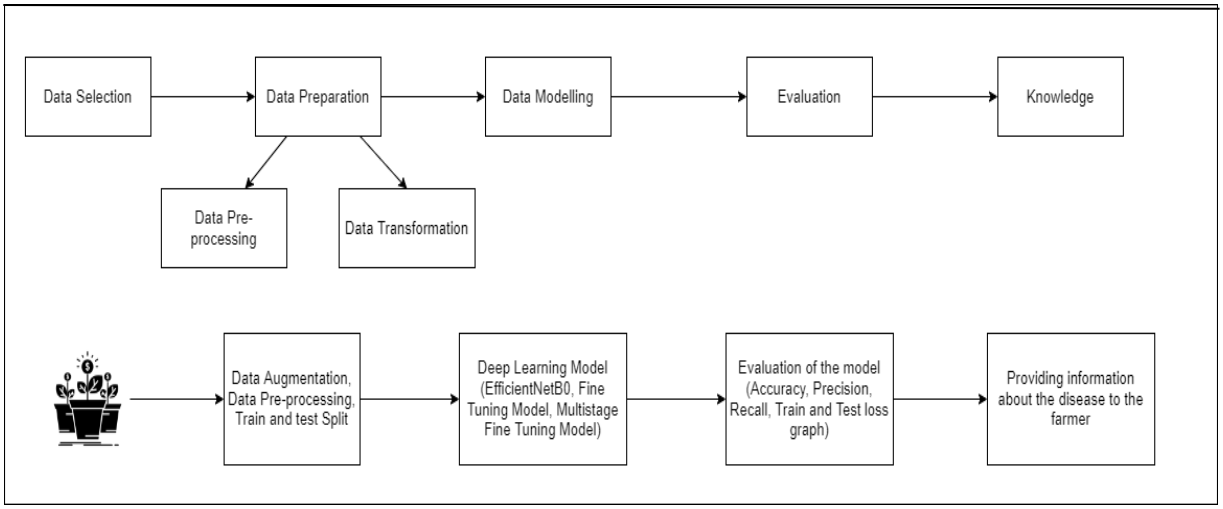

Figure 1: KDD Process

3.1 Data Selection

It is difficult to find a publicly accessible dataset for agricultural research. The majority of the time, researchers create their datasets by visiting farms and clicking pictures of plants with a camera; these are kept private. This process is very time-consuming and mundane. Due to extensive research being conducted on the agriculture industry, there are few publicly accessible data available. This research makes use of such public data referred to as the Large Wheat Disease Classification dataset (LWDCD2020) used by (Goyal et al. 2021). The dataset contains 3,683 images of wheat plants categorized into four categories: leaf rust, wheat loose smut, crown and root rot, and healthy plant images.

3.2 Data Pre-Processing

Pre-processing the data is essential since it can improve the prediction's accuracy. Initially, the data were saved in four folders labeled with the name of the disease. Then, the data was read iteratively along with the label and kept in a list format. Following data reading, the color of the images was transformed from RGB to BGR for Keras compatibility. The images were scaled down to a resolution that was suited for the algorithms, and uniformity was provided so that classification algorithms could function effectively. The number of images of each disease is almost the same. The data was then divided into train and test sets.

3.3 Data Transformation

Data transformation was used to prevent model overfitting, and it is crucial to transform the data appropriately to obtain accurate results. The images and labels were initially converted into a NumPy array. On the training dataset, augmentation was performed using the ImageDataGenerator module of Keras. The following parameters were included while augmenting the data.

Rotation range - Rotating the image by the number of degrees specified.

Zoom range – Images are zoomed in based on the value.

width shift range – Image shifting vertically having a value between 0 and 1

height shift range - Image shifting horizontally having a value between 0 and 1

shear range – Image is shifted in a given degree of angle

horizontal flip - the image is flipped along the horizontal axis

fill mode – points outside the input are filled with the "nearest" mode

3.4 Data Mining

A deep learning model was developed and optimized to produce the optimal model in terms of accuracy and precision. In this study, the transfer learning CNN model EfficientNet was utilized. The ImageNet dataset was utilized to train this model, and the pre-trained weights were utilized to retrain the model on the wheat plant dataset. In this study, only the last layers of the model were retrained. The original and augmented datasets were utilized for training the model. In addition, the model was trained on both datasets using the multistage fine-tuning technique. The models were implemented to determine if they outperform the existing models.

3.5 Evaluation

The model's performance was evaluated using the evaluation metrics. The models were evaluated according to their precision, recall, f1 score, and accuracy. A confusion matrix was created to determine the number of correctly identified images of plant diseases. The best model for the project was determined based on these metrics by comparing the implemented models.

4 Design Specification

The architecture has two levels, which are referred to as the client tier and the business logic tier. Python was selected as the programming language due to its easy usability and the availability of libraries such as Keras and matplotlib for data reading, processing, modeling, and visualization. Jupyter notebook was used as the IDE for the script implementation. After reading and pre-processing the data, the business logic tier was responsible for implementing the transfer learning Efficient model. The models were compared to determine the optimal model, and then the matplotlib was used to visualize the results on the client tier.

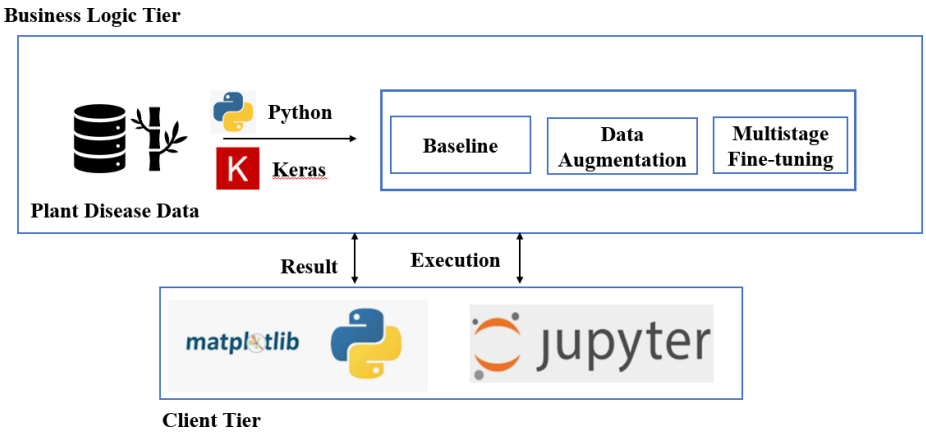

Figure 2: Design

4.1 Convolutional Neural Network

CNN is a deep learning model used for image processing and classification. CNN's complexity increases with each successive layer. These layers are used to extract the image's primary features. Unlike machine learning algorithms, feature extraction does not require external software or algorithms. Various characteristics, such as shape, edge, and color, are extracted from the image while it is being processed through the layers. Following feature extraction, these features are used for classifying the images with the most appropriate label. The CNN model has three layers: the convolutional layer, the fully connected layer, and the average pooling layer. These layers are briefly described below:

Convolutional Layer: Feature extraction on the images is performed by this layer. The basic features are extracted at the start, and a high level of feature extraction is done as the image proceeds through the layers.

Pooling Layer: In this layer, the dimension of the images is reduced to avoid the usage of high computation power and reduction of learning parameters. In the current study, Average pooling was used for extracting the average features from the images.

Fully connected layer: This layer appears in the CNN's last layers, also called as Dense layer. The Dense layer is utilized to classify the output and assign features to a specific label

Dropout layer: The dropout layer prevents model overfitting by dropping a few of the neurons from the layers.

Softmax layer: This is the last layer of the CNN model and is used for generating output probabilities.

4.2 Transfer Learning

Training a CNN can be difficult as it requires a large amount of data and a deeper neural network, resulting in high computational power and ample training time. Transfer learning is advantageous for achieving good results and faster training on small datasets. Transfer learning is a technique that allows the knowledge of a previously trained model to be applied to a different dataset. The pre-trained models are trained on the ImageNet dataset, containing millions of images and can classify up to one thousand categories. The current study will utilize EfficientNet to train the wheat plant images. Due to its training on a vast dataset, the model is recognized for producing accurate results for image classification. The final layers are retrained on a dataset of wheat plants. The model architecture is described below:

- The EfficientNetB0 has 3x3 and 5x5 convolutional layers stacked on each other. The model has seven convolutional layers and uses one average pooling layer. A fully connected layer has 512 neurons along with the Softmax layer for

classifying the images. A dropout layer (0.6) was added to regularise the model.

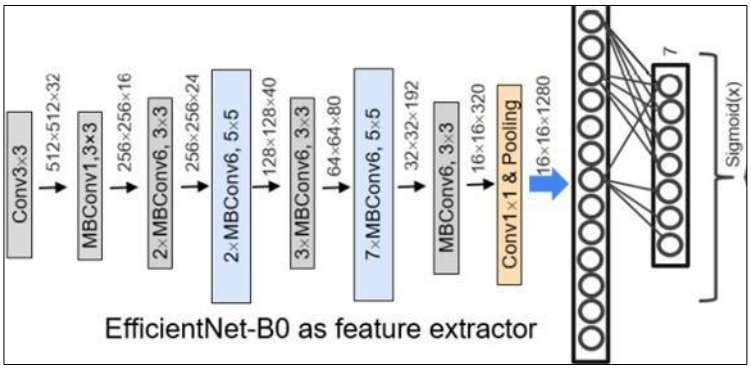

Figure 3 EfficientNetB0 (Genaev et al., 2021)

5 Implementation

This section describes the various steps taken for implementing the classification model used to classify wheat plant diseases using image data.

5.1 Setting Up the Environment

The project was executed using the 64-bit Windows 10 operating system, and Python was the preferred programming language. Keras, OpenCV, OS, Pickle, Matplotlib, and TensorFlow were the libraries installed for implementation. The environment supports GPU for quicker image processing and model training. The computer's folder served as a storage and location for the data and a point of access. Libraries like OpenCV, O.S., pickle, and NumPy were used for pre-processing. The implementation of the model was performed using Keras and TensorFlow. After completion of the performance, the results were presented using the matplotlib library[1]. The IDE used for programming was Jupyter Notebook.

5.2 Data Handling

The data underwent pre-processing and model-appropriate transformations. Initially, images and labels were read from the folders and resized to 22x224 pixels as per the model specifications. The image and the label were converted to arrays and then divided into train and test sets in the proportion of 75:25. After splitting the data, the train set was augmented using ImageDataGenerator with the parameters described in section 3.3. The training set was used for training the model, while the test set was used for evaluating the performance of the model unseen data.

[1] Matplotlib — Visualization with Python

5.3 Classification Model

The EfficientNet model leads to improved performance and a shorter training period. The model receives the data following pre-processing and data augmentation. The advantage of using a CNN model is that the model itself extracts the features from the images while training. Various layers were defined for feature extraction and classification during model training. An Average Pooling layer was defined as having a size of 5x5 for extraction of features. In the dense layer, Rectified Linear Unit (ReLU), a non-linear activation function was utilized to increase the model's speed. This function returns zero if the output is negative, whereas it returns the value if the result is positive. The data is converted into a one-dimensional array using the flattened layer. The Dropout (0.6) layer was applied for preventing the model from overfitting. The last layer of the model contains the classification function called Softmax used for classification. The Adam optimizer was utilized, and the learning parameter was set to 0.01. A training batch size of 64 and 20 epochs was chosen for training.

EfficientNetB0 Layers	Value
Flatten Layer	(None, 1280)
Average Pooling Layer	(None, 1,1,1280)
Dense Layer	(None, 512)
Dropout Layer	(None, 512)
Dense Layer	(None, 4)
Total Parameters	4,707,495
Trainable Parameters	657,924
Non-trainable Layers	4,049,571

Figure 4: Efficient Model Layers

Additionally, the model was trained on both the original dataset and on augmented data to test the improved model's performance. The model was trained using a multistage fine-tuning technique. Multistage fine-tuning refers to the fact that the model is trained incrementally.

5.3.1 Early Stopping

The early stopping strategy was employed to determine the model that suited the dataset the best. The main purpose of using early stopping was to avoid overfitting the model. Early stopping defines the minimum delta (min_delta), patience, and monitor parameters. The smallest change necessary to qualify as an improvement in the performance being measured is known as the min_delta and the value was set to 0.001. The usage of patience results in training halting. In this case, the model will terminate training if no progress is shown within the epochs. The value for patience was given as 3. Validation loss was the metric that was tracked throughout the training. The best model was stored in a file with a given directory.

5.3.1 Baseline

The original, pre-processed data was used for training the baseline model. Due to the use of early stopping, the initial training of the model was terminated at the eleventh epoch. However, the model did not converge properly and did not show stability. Thus, more epochs were used to train the model to achieve better results. Early stopping was not used during the second training as more epochs were needed for training the model. For evaluating the performance of the model, plots of the train and test data were generated. The parameters defined during the training of the model are given in the table below:

Parameters	Value
Image Size	224x224
Batch Size	64
Early Stopping Patience	3
Learning Rate	0.001
Early Stopping Monitor	Validation Accuracy
Optimizer	Adam
Epoch	15

Table 1: Parameter List

5.3.2 Data Augmentation

The model was trained on the augmented dataset to improve its performance. The train data was augmented and used for training, while the test data was used to evaluate the performance of the model on unseen data. The augmented train data was saved in an object with the name "training." The instance of the “training” object was fed to the model for training. This model did not use early stopping during training because the model was not converging properly due to the early halting of the training.

Parameters	Value
Image Size	224x224
Batch Size	64
Learning Rate	0.001
Epoch	15
Optimizer	Adam

Table 2: Parameter List

5.5.3 Multistage Fine Tuning

In this model technique, augmented data was initially used for training the model. Upon completion of the initial training, the model was retrained using the original dataset. This technique was used to familiarize the model with noise; therefore, it was initially trained on a noisy dataset and then on the original dataset, so that it can accurately predict the disease. Due to the use of early stopping during the initial training, the model stopped training at the tenth and seventh epochs, respectively to avoid overfitting.

Parameters	Value
Image Size	224x224
Batch Size	64
Early Stopping Patience	2
Learning Rate	0.001
Early Stopping Monitor	Validation loss
Optimizer	Adam
Epoch	15

Table 3: Parameter List

6 Evaluation

The performance of the models was evaluated based on the evaluation metrics. This section evaluates the models on precision, f1 score, and recall accuracy. Additionally, the accuracy and loss plots on the train and test sets were plotted for analysis. A confusion matrix was generated to check the accurate number of results predicted for each label. The evaluation of the model based on these metrics will determine the optimum model for the project.

6.1 Baseline model

The use of early stopping terminated training at the 8th epoch, where it was evident from the plot that the model had not converged and could be trained for additional epochs. After training the model for a greater number of epochs, from figures 5 and 6, both plots can be observed to be more stable. The loss and accuracy plots revealed that accuracy is steadily increasing and converging since the 4th epoch, while loss is decreasing and converging slightly since the 10th epoch. The difference between train and test loss widens after the 6th epoch but remains constant after the 10th epoch. This training did not utilize early stopping because the model required additional training epochs.

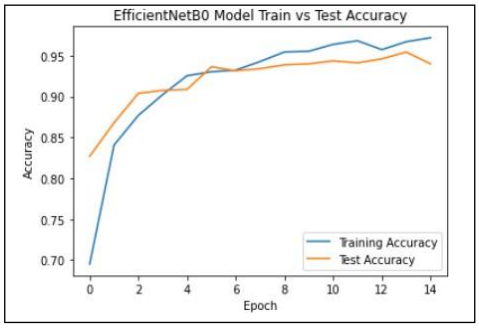

Figure 5: Train Vs Test Accuracy

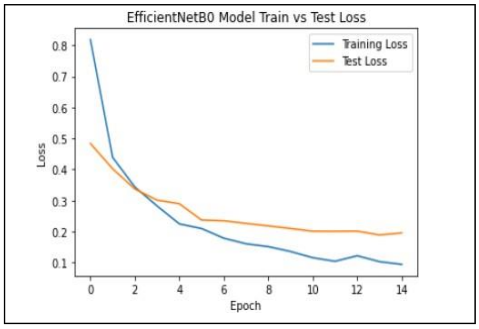

Figure 6: Train Vs Test Loss

6.2 Data Augmentation

From figs 7 and 8, the model achieved good results, however, the accuracy and loss plots show significant fluctuations between train and test after the 4th epoch,

demonstrating that the plots are slightly unstable. The plot demonstrates that after the 12th epoch, test loss increases while test accuracy decreases. The train and test accuracy overlap, but this is not evident in the loss plot.

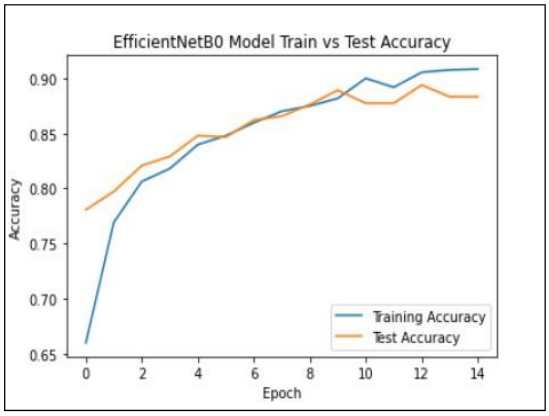

Figure 7: Train Vs Test Accuracy

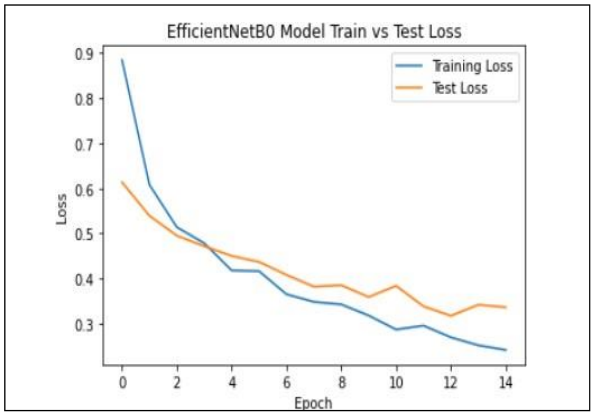

Figure 8: Train Vs Test Loss

6.3 Multistage Fine-tuning

Figs 9 and 10 show the plots of the model trained on the augmented dataset. The test accuracy and loss are fluctuating after 3rd epoch. Furthermore, the training accuracy is continuously rising throughout the training, whereas the training loss is descending smoothly after the 4th epoch. Also, the test loss increases, whereas test accuracy decreases after the 12th epoch. To bring more stability, the model was retrained on the original dataset. Fig 11 and 12 show plots of the trained model on the original dataset. The difference between the train and test in both plots is increasing from the beginning of training. On the train set, the model works well, but not on the dataset. As we can see in fig, the training loss is descending, however, the test loss is increasing from the 5th epoch. Also, the testing accuracy is fluctuating throughout the training. Early stoppingwas used for halting the training of the model from further overfitting.

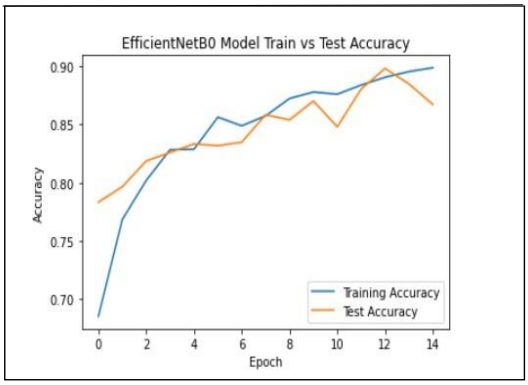

Figure 9: Train Vs Test Accuracy

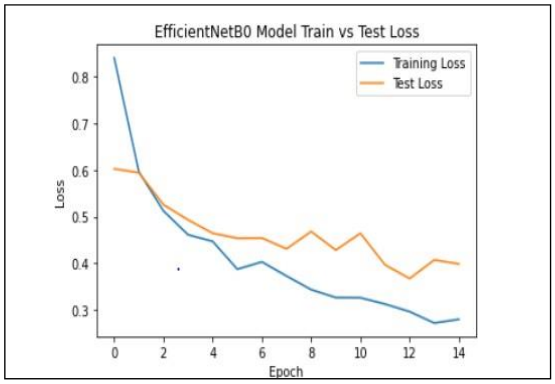

Figure 10: Train Vs Test Loss

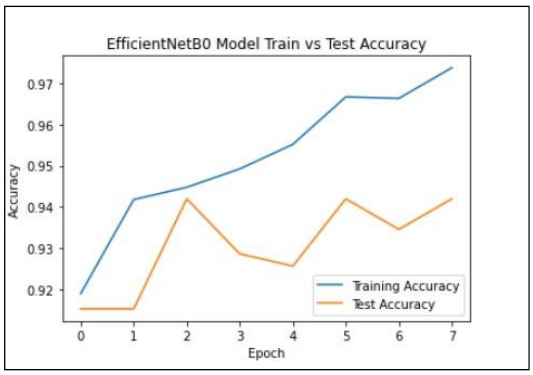

Figure 11: Train Vs Test Accuracy

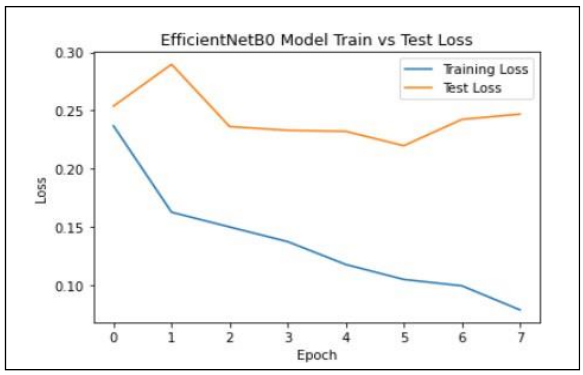

Figure 12: Train Vs Test Loss

6.4 Model comparison

We can conclude from the precision, f1 score, accuracy, and recall values that the baseline model provides better outcomes. We don't want to overlook a wheat plant affected by the disease. This indicates that we want to identify as many infected wheat plants as possible. Therefore, we require a high recall value for this study. After comparing the precision and recall values, the baseline model gave the highest recall value of 97%, and also gave an accuracy of 96%. Moreover, fig 13, 14, and 15 comparison of the confusion matrix reveals that the baseline model classified diseases with greater precision than the other two models. In addition, the plots for this model exhibit greater stability, as discussed in section 6.1. Thus, the baseline model is the optimal model for this research.

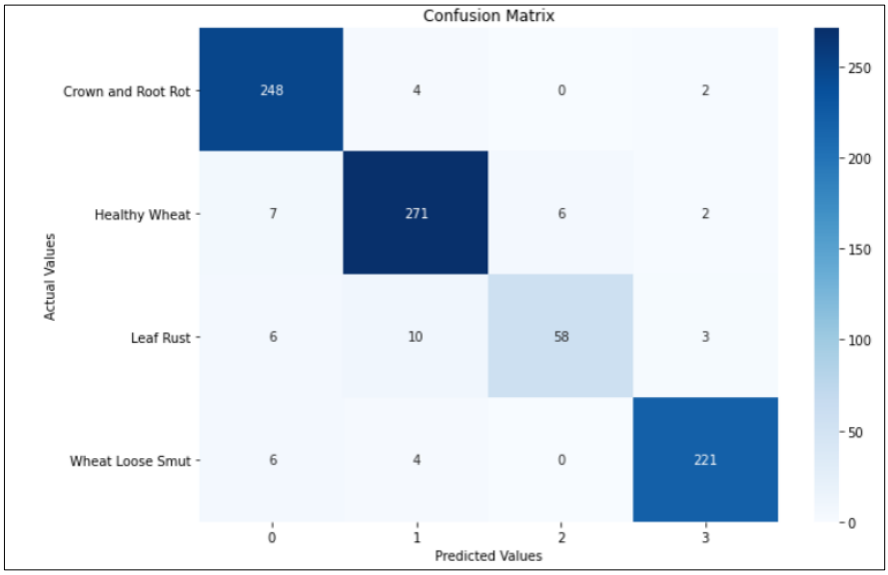

Figure 13: Confusion Matrix (Baseline)

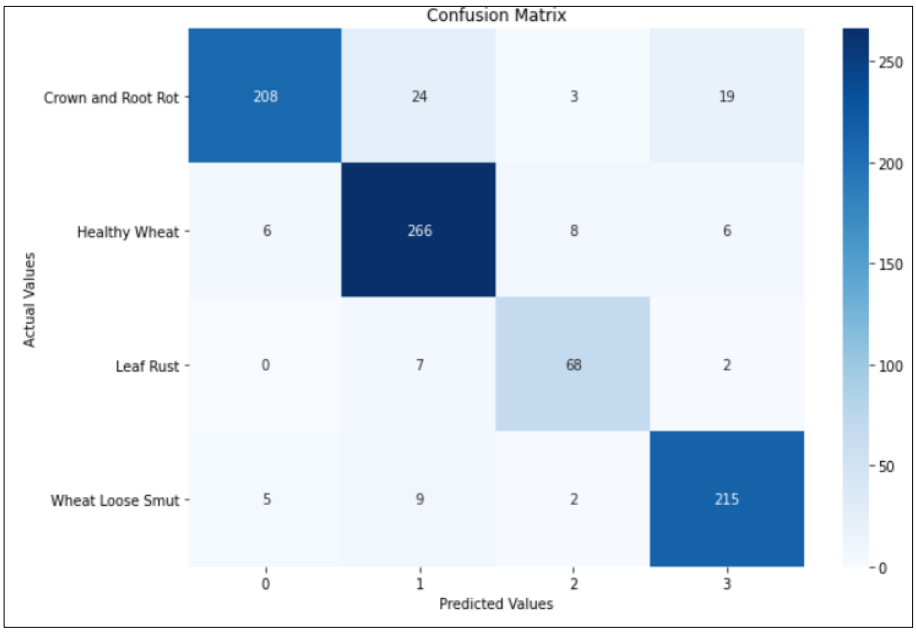

Figure 14: Confusion Matrix (Data Augmentation)

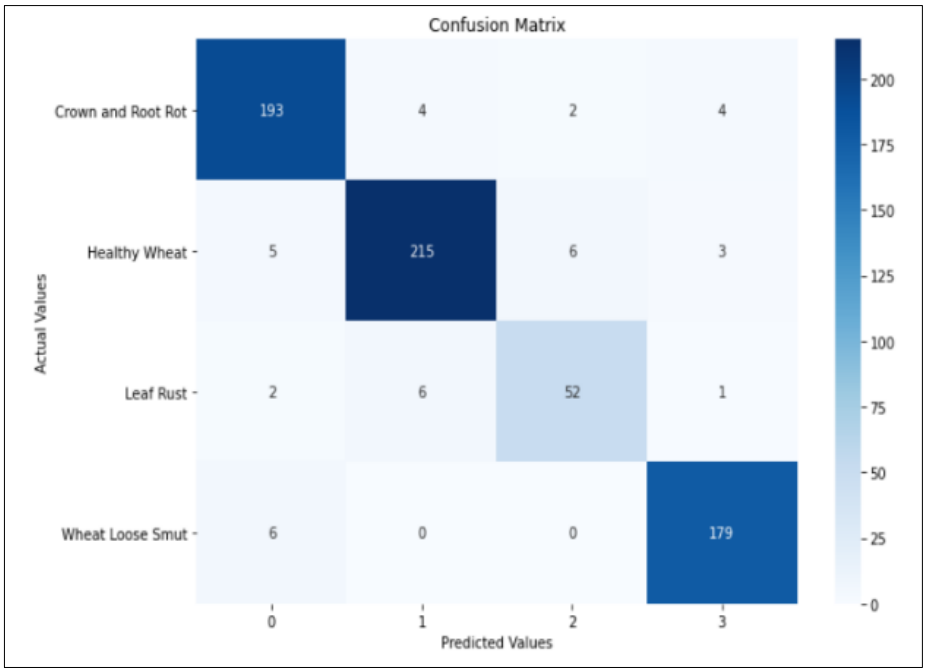

Figure 13: Confusion Matrix (Multistage Fine-tuning)

Evaluation Metrics	Accuracy	Precision	Recall	F1 Score
Baseline	96%	95%	97%	96%
Data Augmentation	88%	95%	84%	89%
Multistage fine-tuning	94%	94%	95%	94%

Table 4: Evaluation Matrix

6.5 Discussion

The study's primary objective was to build a model for accurately identifying plant diseases. After a comprehensive study of the literature review, the limitations of the existing systems were identified. It revealed that existing models required either a larger dataset, a large amount of computational power, or external algorithms for feature extraction. The transfer learning model was the solution to these problems which was used for the current study. In the current study, before training the model, the necessary pre-processing such as resizing, changing the color of the images, and converting the data into a NumPy array was performed. Data augmentation, validation, and fine-tuning were performed for regularization and enhancement of the model.

The results demonstrated that the transfer learning model performed well on the small dataset. However, some difficulties were faced during the development. The primary issue was the lack of versatile data that could have enhanced the model. In addition, the model has only been trained on four types of diseases. If the wheat plant is affected by a disease of a different class or by multiple diseases, the model may perform poorly. Also, there were slightly fewer images of the leaf rust disease due to which some biases were included. In addition, training the final few layers of the transfer learning model is time-consuming and requires a powerful GPU. Due to the extensive amount of time required for training, the image dimensions were reduced for faster processing. Training on larger dimensions and higher-quality images can lead to better results.

7 Conclusion & Future Work

The preceding section illustrates the advantages of using transfer learning models for the identification of wheat plant diseases, thereby achieving the outlined project goals. After comparing the implemented models, the baseline model produced the best results and performed well on the test set. In the baseline model, the plots showed stability and achieved a recall value of 97%. Also, the baseline model classified the images more precisely than the other two models.

The research can be expanded by developing models to identify the pests affecting the wheat plants. The models are trained on four types of diseases and thus will not be able to detect other types of diseases or multiple diseases; however, this can be improved

in the future. The models can be deployed on smartphones, allowing farmers to identify diseases by simply clicking images of affected plants. Moreover, along with diseases, farmers can also be provided with solutions. This application will be cost-effective and of great assistance to the agriculture industry.

8 Acknowledgment

I would like to take this chance to express my appreciation and respect to my mentor Prashanth Nayak for his guidance, insightful feedback, and consistent encouragement throughout this project. His informative suggestions were of great support throughout my research.

References

Ahmed, K., Shahidi, T.R., Alam, S.M.I. and Momen, S., 2019, December. Rice leaf disease detection using machine learning techniques. In *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)* (pp. 1-5). IEEE.

C. K., S., C. D., J. and Patil, N., 2022. Cardamom Plant Disease Detection Approach Using EfficientNetV2. *IEEE Access*, 10, pp.789-804.

Ganatra, N. and Patel, A., 2020. A multiclass plant leaf disease detection using image processing and machine learning techniques. Int. J. Emerg. Technol, 11(2), pp.1082-1086.

Genaev, M.A., Skolotneva, E.S., Gultyaeva, E.I., Orlova, E.A., Bechtold, N.P. and Afonnikov, D.A., 2021. Image-based wheat fungi diseases identification by deep learning. *Plants*, *10*(8), p.1500.

Goyal, L., Sharma, C.M., Singh, A. and Singh, P.K., 2021. Leaf and spike wheat disease detection & classification using an improved deep convolutional architecture. *Informatics in Medicine Unlocked*, *25*, p.100642.

Islam, M.T., Al-Absi, H.R., Ruagh, E.A. and Alam, T., 2021. DiaNet: A deep learning based architecture to diagnose diabetes using retinal images only. *IEEE Access*, *9*, pp.15686-15695.

Kartikeyan, P. and Shrivastava, G., 2021. Review on emerging trends in detection of plant diseases using image processing with machine learning. *International Journal of Computer Application*, *975*, p.8887.

Kulkarni, P., Karwande, A., Kolhe, T., Kamble, S., Joshi, A. and Wyawahare, M., 2021. Plant disease detection using image processing and machine learning. arXiv preprint arXiv:2106.10698.

Lee, S., Chan, C., Mayo, S. and Remagnino, P., 2017. How deep learning extracts and learns leaf features for plant classification. *Pattern Recognition*, 71, pp.1-13.

Panchal, A., Patel, S., Bagyalakshmi, K., Kumar, P., Khan, I. and Soni, M., 2021. Image-based Plant Diseases Detection using Deep Learning. *Materials Today: Proceedings*,

Poornappriya, T.S. and Gopinath, R., 2022. Rice plant disease identification using artificial intelligence approaches.

Ramesh, S., Hebbar, R., Niveditha, M., Pooja, R., Shashank, N. and Vinod, P.V., 2018, April. Plant disease detection using machine learning. In 2018 International conference on design innovations for 3Cs compute communicate control (ICDI3C) (pp. 41-45). IEEE.

Raina, S. and Gupta, A., 2021, March. A study on various techniques for plant leaf disease detection using leaf image. In *2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS)* (pp. 900-905). IEEE.

Shruthi, U., Nagaveni, V. and Raghavendra, B.K., 2019, March. A review on machine learning classification techniques for plant disease detection. In *2019 5th International conference on advanced computing & communication systems (ICACCS)* (pp. 281-284). IEEE.

Singh, A. and Arora, M., 2020, September. CNN based detection of healthy and unhealthy wheat crop. In 2020 International Conference on Smart Electronics and Communication (ICOSEC) (pp. 121-125). IEEE.

Zhou, C., Zhou, S., Xing, J. and Song, J., 2021. Tomato leaf disease identification by restructured deep residual dense network. *IEEE Access*, *9*, pp.28822-28831.