CSE 299 Junior Design

Final Report

Plant Disease Classification with ML & Android

Submitted to: Dr. Tanzilur Rahman (TNR)

Section: 11

Spring 2021

Group 3

Md Yearat Hossain – 1712275042

Md-Raihanul Hoque Shanto - 1812746042

Md. Imanur Rahman Emon – 1813181642

Plant Disease Classification with ML & Android

Abstract: Agriculture has been associated with us from the very beginning of our socialization and modernization. Our modern world is rooted from agriculture. People learned how to grow plants, how to farm animals, how to use the materials of the environment to aid themselves. Day by day they invented many things for the benefit of themselves. We humans invented more techniques and developed sophisticated machines for us. This development of science touched every aspect of our life, hence the agricultural field too. We studied, explored and invented ways to produce more with less resources to fulfill the demands of ours. Though we have gathered our knowledge, it is not always possible to apply the knowledge properly. There are many obstacles behind these situations. Maybe we have the right tool but not always on the right hand.

Computer science is probably one of the newest fields of study. With the help of computer science, we could solve many complex problems which were unthinkable once. This field of study is advancing so rapidly that every year we get to see more and more sophisticated technologies. Among these advancements, Artificial Intelligence or AI has shown us huge promises. Nowadays with the help of AI we are able to solve problems like detecting issues or predicting something with decent accuracy, in fact in some cases they surpassed humans.

This technology is being used in every sector and agriculture is also one of them. With the help of AI, we can track diseases on crops, predict production rates and many more. Scientists had been working to develop modern systems to aid the people associated with agriculture and farming to aid them in various aspects. Most of the time the farmers are not aware of the disease they face in their plants and also it is not always possible to get help from the authorities in rural places. Now, with the advancement of technology and AI, we can build systems which can aid the farmers remotely without causing much trouble. So, in our project we intend to develop a system that can detect diseases in plants with the help of AI and android.

Table of Contents:

1. Introduction:

Bangladesh is an agricultural country. Almost 70% to 80% of the population is somehow associated with agriculture. Among all, rice, jute and potato are the most produced crops in Bangladesh. These are produced across the entire country and from the cultivation to transportation, many people and industries are associated with these. But these plants can face several diseases which can disrupt the production. In the case of rice, leaf blast, hispa, brown spots are common. And potatoes may face different types of blights like early and late blights. But our farmers are not qualified or privileged enough to detect these diseases by themselves. And it is not always possible to contact proper authorities immediately in rural areas. Due to these issues, many farmers across the country face less production issues every year. And as we discussed above this can cause great impacts on all the persons and industries associated with it. Maybe a farmer can easily cure the disease but first place he has to be able to identify the disease. Also, in rural places across the country the internet can be an issue.

So, we intend to develop a system which can help the farmers to detect the diseases in plants with the help of machine learning and android. Our application will be able to take pictures from the phone and detect plant diseases without accessing the internet. The diseases that we are willing to work on can be detected from the appearance of the leaves. In the case of rice, Leaf blast, Hispa, Brown spot they all leave various markings on the rice leaves thus we can distinguish them from healthy rice leaves. But to identify the specific disease someone must have that prior knowledge about the diseases. Our goal is to build such a system which will detect the specific disease from the leaf pictures.

1.1. Machine Learning:

Machine learning is a field of Artificial Intelligence where we let a machine learn something by itself based on the data provided. For example, in our project, we will provide some data on healthy and infected rice leaves and with the help of some ML algorithms, our computer program will learn how to detect which rice leaf is good and which has a disease.

The core concept of machine learning is to find features from provided data and establish a relation between them. The entire thing happens with the help of various machine learning algorithms on mathematical models. First the machine assigns certain values to link the features of data and then with the help of loss functions it measures the accuracy. Then again depending on the result the machine sets new values for solving the problem.

1.2. Android:

Android is the most popular operating system in the world. There are more than 2.5 billion android users in the whole world.**[1]** Because of android, almost all types of people have the opportunity to use smart phones. Nowadays, android powered devices are really good at performing basic to advanced tasks. With the development of faster chips and better cameras, any budget android phone can capture good pictures.

2. Background Studies:

For our project we chose the following 4 research papers to understand the concept properly.

2.1. <u>Rice Disease Identification and Classification by Integrating Support Vector Machine with Deep Convolutional Neural Network.[2]</u>

Published by: Dept of Electrical and Electronic Engineering.
Hajee Mohammad Danesh Science and Technology University.

In this research, they have used a deep convolutional neural network InceptionV3 with support vector machine algorithm to detect 9 different classes of rice disease. The dataset contained high-resolution pictures taken with professional cameras. The dataset is prepared by themselves. They used Transfer learning and achieved 97.5% accuracy.

2.2. <u>Rice Crop Disease Identification and Classifier.[3]</u>

Published by: Dept of CSE, Rajiv Gandhi College of Engineering, India

Here they also used a DCNN with SVM algorithm but they emphasized more on image processing. Here the images were taken with high-end mobile cameras as their intention was to deploy the system into smartphones. Their system uses a cloud-based solution where a photo is uploaded to the cloud and the cloud sends the prediction results back to the device

2.3. <u>Rice Disease Classification using Convolutional Neural Networks.[4]</u>

Published by: Dept of CSE, Rochester Institute of Technology.

This study is more focused on machine learning models. Here they trained the dataset on various small to complex pre-trained CNN models and showed the differences in training time, accuracy, and performance across various popular models like AlexNet, GoogLeNet, ResNet-50, Inception-v3, ShuffleNet and MobileNet-v2.

2.4. <u>Disease Detection in Plum with CNN under True Field Conditions[5]</u>

Published by: Dept of CSE, Islamia College, Pakistan.

In this study the researchers solved disease detection in plum by using various popular deep learning models. The dataset contained high resolution pictures of both plums and leaves. The main goal of this research was to demonstrate how much increased performance we can achieve by using augmented data. Here they generated more data by applying contrast, saturation and brightness on the dataset. Then they compared the results with the performance of the raw data.

* None of the papers mentioned their dataset as public or provided any relevant resource.

3. Methodology:

Our goal is to build a system which can detect plant diseases remotely from a smartphone. To reach our goal we have to go through various processes.

First, we need to choose a dataset which we will use for our machine learning project. Then we need to study and process the dataset.

Then, we need to select a machine learning model to train with the dataset.

Then, after testing and evaluating we need to deploy the model into android as an android application.

So we divided our project into 3 different sections:

1. Dataset: Exploration, Preparation and Augmentation.
2. Model: Exploration, Training and Evaluation.
3. Android Application: Development, Deployment and Testing.

3.1. System Diagram:

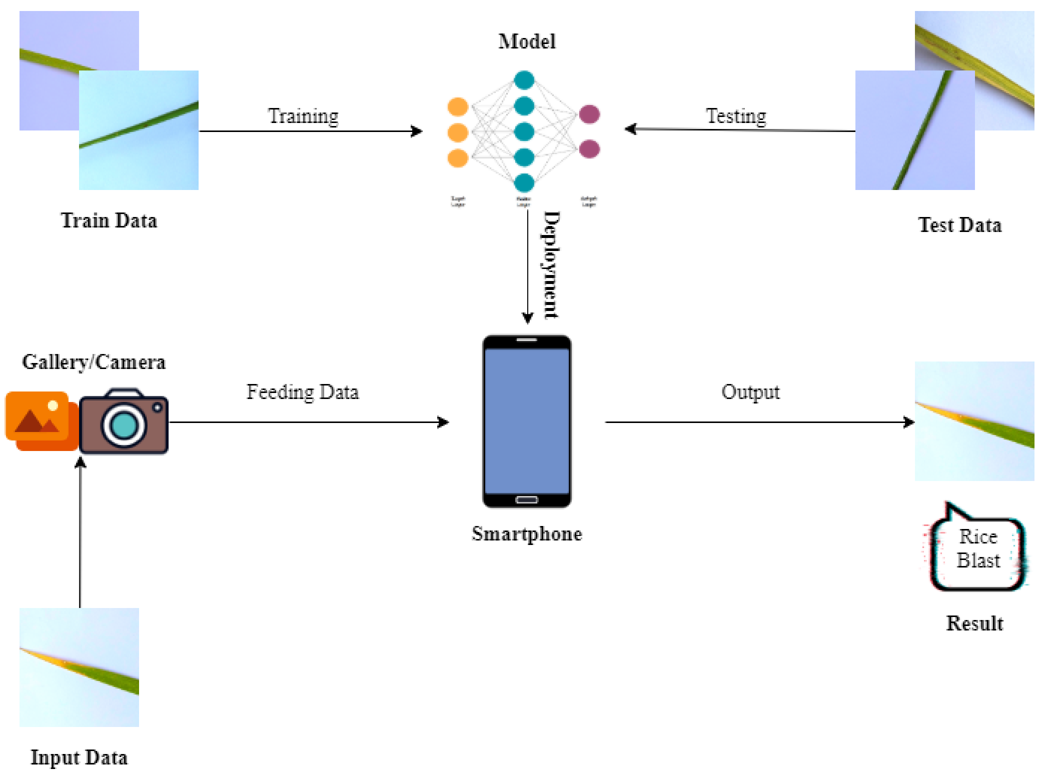

Figure 1. System Diagram

3.2. Workplan:

Week 1 – Project Description Presentation
Week 2 – Dataset Exploration
Week 3 – Dataset Process and Augmentation
Week 4 – Model Exploration
Week 5 – Model Training and Evaluation
Week 6 – Model Testing and Tweaking
Week 7 – Android Application Development
Week 8 – Further Trial and Testing
Week 9 – Final Demonstration

3.3. Software & Technologies Requirement:

a. **TensorFlow & Keras:**

TensorFlow and Keras are python frameworks for developing machine learning and deep learning related tasks. These contain all the necessary algorithms and features we need to build, train, process machine learning and deep learning models.

b. **Google Colab:**

Google Collaboratory or Colab is a cloud-based platform for machine learning related works. Colab provides us all the necessary modules, software and hardware we need for training models. Though Colab can be slower compared to local machines, it provides all the tools we need.

c. **Android Application: Flutter:**

Flutter is a popular app development framework made by Google. Flutter is made for rapid UI based application development. Instead of using Java, Flutter uses a programming language called Dart. Flutter can be integrated into android studio easily to develop android applications.

3.4. Dataset Exploration:

Dataset is one of the main components of the machine learning process. The machine learning models are trained on datasets. A proper dataset can help to build an ML project with much more efficiency and lesser hassle. Due to the development of internet technologies, we have access to more data nowadays but it is not always easy to find a properly maintained dataset. Dataset exploration can be a tricky process but can not be avoided.

We chose Kaggle as our dataset resource as it is the most popular public data science platform. We worked with 2 different datasets.

a. Dataset 1: Rice Diseases Image Dataset by Huy Minh Do[6]:

This dataset contains 3,355 images of 4 different classes. Among these 3 are for different rice diseases and one for the healthy leaves. These are separated into 4 different labeled directories, BrownSpot, Healthy, Hispa, LeafBlast. The pictures have variable pixels in dimension, use RGB color space, and were taken using a Xiaomi Redmi Note 4 smartphone.

Healthy Brown Spot Hispa LeafBlast

Figure 2. Rice Diseases Image Dataset

b. Dataset 2: PlantVillage Dataset by Abdallah Ali[7]:

PlantVillage dataset contains 50,000 expertly curated images on healthy and infected leaves of various crop plants. These plants include Apple, Corn, Strawberry, Pepper, Potato, Tomato, etc. The dataset has 3 subfolders of 38 different classes, one contains all colorful pictures, one contains all greyscale pictures and one contains all colorful segmented pictures. We took only the potato leaves dataset from the entire dataset. There were 2152 pictures of 3 classes.

Early Blight Healthy Late Blight

Figure 3. PlantVillage (Potato) Dataset

3.5. Dataset Process and Augmentation:

Dataset process is an important task in training machine learning models. A dataset can have several issues like corrupt files, garbage values, impossible data combinations, and missing values. Dataset processing can take a short to a long time depending on the present condition of data and how we need the data for the model.

At first we visualized the entire dataset and found no corrupt or irrelevant data. Then we renamed all the data with their parent folder names which gave them proper labels. In the case of the rice disease dataset, the images had variable resolutions from 1300x1300 pixels to 1800x1800 pixels and we resized them all into 400x400 pixel images. But in the case of the Potato dataset we did not need to downsize or resize the resolution as all the images had 256x256 pixel resolution. Here labeling and resizing the data helped us to organize and visualize the data in a more efficient way when needed. We split the dataset into 80:20 ratio for training and validation purposes.

Augmentation is generating more variants of existing data by applying shifting, rotation, flipping, cropping, adding noise etc. We applied some basic data augmentation techniques following the Keras image augmentation documentation.**[8]**

3.6. Model Exploration:

For our project we chose to work with the MobileNetV2**[9]** model. MobileNetV2 is based on depth wise convolution and inverted residual blocks hence good for use in mobile devices. Convolution is a resource hungry task but in this particular model the main convolution is separated in multiple convolutions hence called the depth wise convolution. Also, it is based on inverted residual blocks. It keeps the data flow through the model compressed but expands the data inside the blocks to run convolutions.

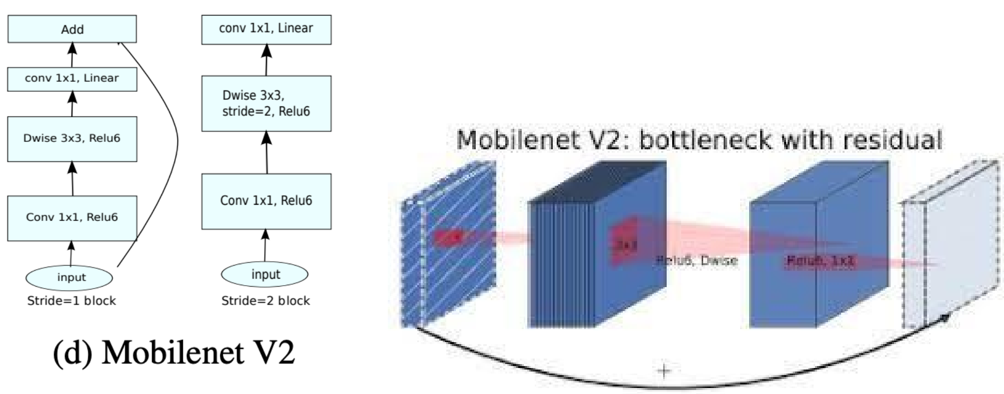

Figure 4. MobileNet V2 Summary

3.7. Model Training and Evaluation:

Model training is the part where we finally train our model with the dataset we have chosen. During this process machine learning algorithms help the model to learn from the data provided. The model gets trained and tested simultaneously at the same time.

We used Categorical Cross Entropy**[10]** to calculate the loss. It is used to measure loss in multiclass detection problems. As optimizer for the model, we used Adam**[11]** optimizer which can work as a replacement algorithm for SGD algorithm. The optimizer acts as the core algorithm which helps the model to learn. In the very last layer of our model we implemented a

SoftMax**[12]** function. This function is used for multiclass classification problems and it returns probabilities of each class.

We applied Transfer Learning in our training process. In transfer learning the model is already pre trained with similar kinds of data. Our model is pre-trained with imagenet**[13]** dataset hence good for detecting images. We used 224x224 sized images and the number of images per batch is 64. The data is splitted into 80:20 ratio for train and validation. We also kept some data for testing the application which were completely unseen by the model.

3.8. Android Application:

With the help of tflite converter we converted our model into a tflite format model. The converter provided us a model.tflite and a labels.txt file.

Then we designed and implemented our application with basic ui and functionality. We used Flutter as our android development framework and Android Studio as IDE. During the development process we tested everything on a virtual machine replication of the Pixel 2 device. Our application has two features, it can either capture a photo using the camera or it will access the file manager or gallery. Once the picture is provided the application returns the prediction result in a snap. The application automatically stores the prediction results in it's cache so that if anyone provides it the exact same image file again the application will directly return the result from cache, instead of running the prediction again.

Minimum Application Requirement:

Android 5, 3GB Ram, 80mb storage, camera and storage permission.

4. Results:

For the Rice Disease Dataset, we trained the model for 10 epochs and achieved 60% training accuracy and 52% validation accuracy which were totally unacceptable. To identify the problem we trained the same dataset with ResNet 152**[14]** which is a powerful model and achieved 62% training accuracy and 56% validation accuracy.

Then we tried tweaking various parameters like epochs, dropouts, image size, batch size but none of them seem to work.

Then we tried some different datasets just to make sure that model was working right. As we found the model was working properly we decided to work more on the dataset. Then following one of our background studies we applied saturation, contrast, brightness and sharpness enhancement on the entire dataset. But this time the model overfitted with 97% training accuracy and 48% validation accuracy.

Then we trained our model with the Potato dataset without any enhancement and achieved 96% accuracy in both training and validation section, which was decent enough to be deployed.

Model	Dataset	Train Accuracy	Validation Accuracy
MobileNet V2	Rice Disease	60%	52%
ResNet 152	Rice Disease	62%	56%
MobileNet V2	Rice Disease (Enhanced)	97%	48%
MobileNet V2	Potato (PlantVillage)	96%	96%

Table 1. Training Results

Study	**Model**	**Dataset**	**Train Accuracy**	**Validation Accuracy**
1	Inception V3 with SVM	Rice	97.5%	93%
2	Inception V3 with SVM	Rice	Not Mentioned	Not Mentioned
3	MobileNet V2	Rice	62.5%	62.5%
4	Inception V3	Plum	86.81%	Not Mentioned
Our Project (Potato)	MobileNet V2	Potato	96%	96%

Table 2. Results compared to background studies
(The best result from each study are taken)

On the application end, the model predicted only one false result on late blight but accurately predicted the rest of the test set.

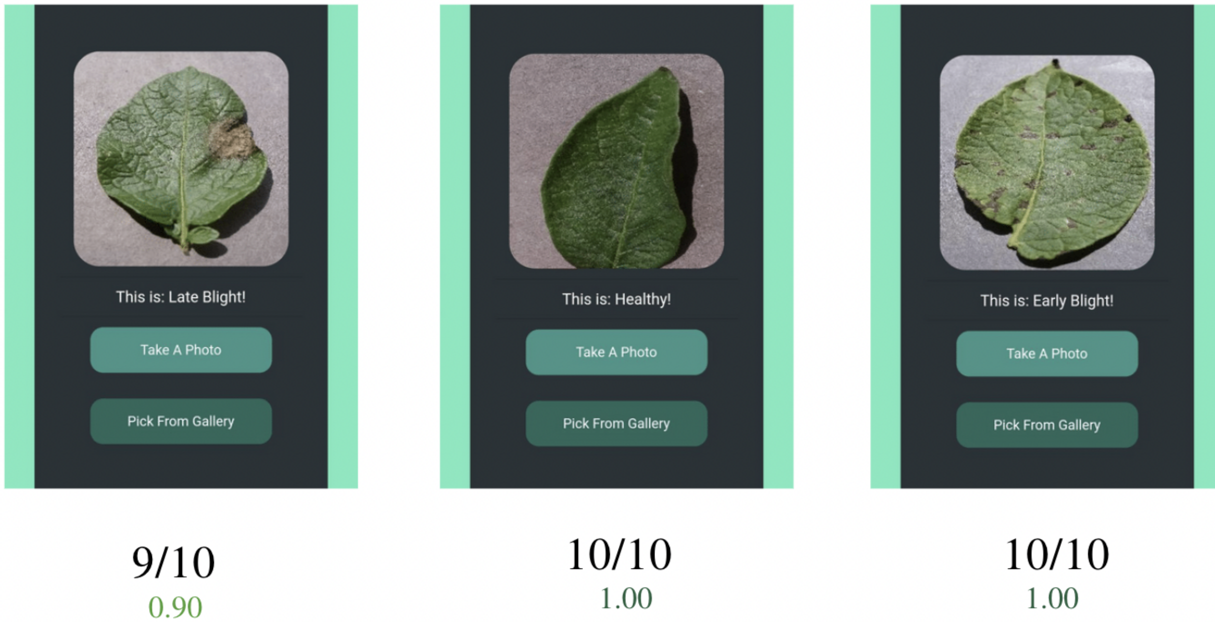

Figure 5. Application Accuracy Test

We tested our application on various types of android devices to check compatibility issues. As a result we found that the application generally will not get installed on those devices which have ram less than 3 GBs.

Device	CPU	Ram	Android	Price
Samsung S20 (2020)	Exynos 990(7nm)	12 GB	11	80K
Google Pixel 2(2017)	Snapdragon 835(10 nm)	4 GB	8	60K
Realmi 5i(2020)	Snapdragon 665 (11 nm)	3 GB	9	16K
Vivo Y17(2019)	Helio P35 (12nm)	4 GB	9	21K
Huawei Nova 3i(2018)	Kirin 710 (12 nm)	4 GB	8	22K
Redmi 4(2016)	Snapdragon 430 (28 nm)	2 GB	6	15K
Samsung J2(2015)	Exynos 3475 Quad (28 nm)	1 GB	5	15K
Redmi Go	Cortex-A53	1 GB	8	6K

Table 3. Application Compatibility Test
(Red marked devices does not work)

Depending on how many features a photo has, the app took a different amount of time in each prediction. Inference refers to the process of executing a TFLite model on-device in order to make predictions based on input data. The range of inference was 58 - 80 units.

Tested on Samsung S20 Exynos 990 (7nm)

Figure 6. Inference Test (latency)

The peak memory usage for the application never exceeded 300mbs in our tests. It takes only 32mb of ram when it sits idle.
The application always detected with proper accuracy across all the devices and never crashed. But we noticed some latency depending on the storage type of the device we were testing. For example the application ran way faster on UFS 3.0 storage devices than the standard storage enabled devices.

5. Discussion:

Though we tried to build a rice disease detection system, our main goal was to implement the idea that we can detect plant diseases with our phones without accessing the internet. Throughout the development of this project we faced various challenges and learned various workarounds.

We mainly struggled with our dataset. The development of the model and application was not that challenging but working with data was. We faced various performance issues due to our first dataset (Rice Dataset) and we tried various techniques to fix that issue. We tried to enhance the image quality by applying various image processing techniques but the model overfitted this time. Then we tried to remove similar looking data belonging to different classes. As a result we lost a good amount of data which we could not balance even after applying image augmentation. With this new version of enhanced hand picked data the model always went to an overfitting situation.

We believe we faced issues with the dataset because the dataset was not made properly. The dataset had some indistinguishible data which might have led the model to it’s failure. Then as we concluded that the previous dataset (Rice Dataset) had issues, we had to continue our work with our second dataset which was the potato leaves dataset.

Initially we trained our model on Google Colab but eventually we had to move onto our local machine due to some factors. Google Colab has limited space and runtime functionality. But as we had to retrain our model several times, we could not continue with Google Colab anymore. The free version of Colab lets us use only 12 GB of memory which was not flexible enough for our project. Colab sessions run on Nvidia Tesla K40 gpus but on our local machine we could use Nvidia RTX 2060 gpu using which we got significantly faster training results on our local machine. Also our local machine had an Intel i7 8700K (6 core, 4.60 GHz) CPU , 16 GB of ram and 512 GB of SSD along with the GPU.

GPU	Clock Speed	Performance	Training Time (10 Epochs)
Nvidia Tesla K40	745MHz	4.29 TFLOPS	43.51 minutes
Nvidia RTX 2060	1365MHz	6.45 TFLOPS	8 minutes

Table 4. Colab vs Local Machine GPU Performance **[15]**

With our potato leaves dataset we achieved satisfactory results and we moved onto the development of our android application. With the conversion technique of Tensorflow Lite, we could easily convert our large model into a smaller one. TFlite conversion removes less important weights from the model and reduces the precision of values to make the model smaller and less complex. This is why we could deploy our model into a mobile application. But this has drawbacks too. As TFlite models are less complex than larger models, the power and functionality of the models are limited for now.

As we used Flutter, we could rapidly build our application without much complication. The development pipeline of flutter is different from development of Java. In flutter we could easily deploy our converted model and implement our app functionalities with the help of various flutter packages. Because of these packages we did not need to code the entire functionality from the ground. Also we could easily design our user interface with the help of various flutter ui building toolkits.

As we intended to help our farming community with our application, we had to make sure that the application properly works across all types of devices. Mainly we wanted to test our application on mid to low spec mobile devices. To perform such a task we had to collect several types of android devices ranging from old to new, low budget to high budget and test our application manually onto them. But this was definitely not the most efficient way to test the application. There are some online platforms where they let people test applications on a large variety of devices but we could not manage one for free. But our application did well on almost all types of modern devices. This was possible to achieve due to the power and advancements of modern mobile devices.

So in conclusion, we can say that the dataset can be the most important thing in AI development. If the dataset has issues, no matter how sophisticated algorithms we invent, or how many high performance machines we use, we won't get proper results. Also, industries and institutions should make the data as much as public as possible to aid the researchers and developers. AI can shape our future, we hope that the development of AI will help us and everything around us, and we must avoid the bad practices of it.

6. References:

1."Android Statistics (2021)," *Business of Apps*, 13-May-2021. [Online]. Available: https://www.businessofapps.com/data/android-statistics/.

2.M. J. Hasan, S. Mahbub, and S. Alom, "Rice Disease Identification and Classification by Integrating Support Vector Machine With Deep Convolutional Neural Network," *Academia.edu*. [Online]. Available: https://www.academia.edu/43974637/Rice_Disease_Identification_and_Classification_by_Integrating_Support_Vector_Machine_With_Deep_Convolutional_Neural_Network.

3.Vishakha Nagrale, "Rice Crop Disease Identification and Classifier," *International Journal of Computer Sciences and Engineering*. [Online]. Available: https://www.academia.edu/42123981/Rice_Crop_Disease_Identification_and_Classifier.

4.J. Pereira, "Rice Disease Classification using Convolutional Neural Networks," *Academia.edu*. [Online]. Available: https://www.academia.edu/41821127/Rice_Disease_Classification_using_Convolutional_Neural_Networks.

5.J. Ahmad, B. Jan, H. Farman, W. Ahmad, and A. Ullah, "Disease Detection in Plum Using Convolutional Neural Network under True Field Conditions," *MDPI*, 28-Sep-2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/19/5569.

6.H. M. Do, "Rice Diseases Image Dataset," *Kaggle*, 15-Nov-2019. Available: https://www.kaggle.com/minhhuy2810/rice-diseases-image-dataset.

7.A. Ali, "PlantVillage Dataset," *Kaggle*, 01-Sep-2019. Available: https://www.kaggle.com/abdallahalidev/plantvillage-dataset.

8.K. Team, "Keras documentation: Image data preprocessing," *Keras*. [Online]. Available: https://keras.io/api/preprocessing/image/.

9.M. Sandler, *MobileNetV2: Inverted Residuals and Linear Bottlenecks*, 13-Jan-2018. [Online]. Available: https://www.citationmachine.net/ieee/cite-a-website/custom.

10.“Categorical crossentropy loss function: Peltarion Platform,” *Peltarion*. [Online]. Available: https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy.

11.K. Pykes, “Adam Optimization Algorithm,” *Medium*, 06-Jun-2020. [Online]. Available: https://towardsdatascience.com/adam-optimization-algorithm-1cdc9b12724a.

12.“Softmax Function,” *DeepAI*, 17-May-2019. [Online]. Available: https://deepai.org/machine-learning-glossary-and-terms/softmax-layer.

13.*ImageNet*. [Online]. Available: https://www.image-net.org/about.php.

14.PyTorch, “ResNet-152,” *Kaggle*, 13-Dec-2017. [Online]. Available: https://www.kaggle.com/pytorch/resnet152.

15.“Nvidia GeForce RTX 2060 vs Nvidia Tesla K40: What is the difference?,” *VERSUS*. [Online]. Available: https://versus.com/en/nvidia-geforce-rtx-2060-vs-nvidia-tesla-k40.

Group Contribution:

1. Md Yearat Hossain - 1712275042
   - Model Implementation
   - Android Application Development
   - Data Enhancement Processing
   - Android Application Testing (virtual machine) and Inference Testing
2. Md-Raihanul Hoque Shanto - 1812746042
   - Slide Preparation (All)
   - Data Exploration (Rice Diseases)
   - Data Processing
   - Training with Handpicked data
3. Md. Imanur Rahman Emon - 1813181642
   - Background Study (Exploration, Analysis)
   - Real World Android Application Testing on Various Devices
   - Data Exploration (PlantVillage: Potato Leaves)
   - Raw data collection: He was able to collect some raw rice data from the field but the amount and variation were not enough for our project.

For any queries contact: yearat.hossain@northsouth.edu