

**Agripreneurs : Motivating individuals to become entrepreneurs
through agriculture**

Project ID – 2021-090

R.S.W.M.R.L. Rangalla

IT18110562

Bachelor of Science (Hons) in Information Technology
Specializing in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

August 2021

**Agripreneurs : Motivating individuals to become entrepreneurs
through agriculture**

Project ID – 2021-090

R.S.W.M.R.L. Rangalla

IT18110562

Dissertation submitted in partial fulfillment of the requirements for the Bachelor of Science
(honors) degree in IT (Specializing in Software Engineering)

Department of Software Engineering

Sri Lanka Institute of Information Technology

August 2022

Declaration

We hereby declare that this project work entitled “**Agripreneurs: Motivating Individuals to Become Entrepreneurs through Agriculture**” is our own work and this thesis does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, we hereby grant to Sri Lanka Institute of Information Technology, the non-exclusive right to reproduce and distribute my Thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature:

R. Schanck

.....

Date: 13/10/2021

The above candidate has carried out this research thesis for the Degree of Bachelor of Science (honors) Information Technology (Specializing in Software Engineering) under my supervision.

Signature of the supervisor:

Date:

Abstract

Entrepreneurship is one of the most discussed domains in the 21st century mainly because it not only allows individuals to break boundaries to become exceedingly successful but also because it allows a country's economy to develop with very little effort from the governing side. However, due to many high-risk factors, in developing countries like Srilanka this domain have had reached its peak mainly because of the collapsing economy. On the other hand, agriculture is one of the sectors in Srilanka which provides a high number of employment opportunities for individuals.

Creating a bridge between the agriculture and the entrepreneurship domain through modern technology is being achieved. This is achieved by the introduction of a motivational gamification model within the system which allows users to not only engaged with the application but also with other users using the application. An IoT module that will analyze soil data which includes humidity, temperature. An integrated auction platform that consists of a machine learning model to analyze user behavior. A robust image processing model to assist users in uncontrollable situations such as plant diseases.

The goal of this research is to allow employed or semi-employed individuals to make a passive source of income, mainly because due to the pandemic situation many individuals are facing low-income issues.

The focused component involved in aiding users of the agriprenuers application with an extra hand on growing their crops through a robust machine learning model. Many farmers throughout the agriculture domain face many difficulties when growing crops, even engaging for years. Through modern machine learning algorithms, image classification and image processing techniques finding defects in crops has been efficient than ever before.

Keywords—Machine Learning, React Native, BSON, MongoDB, Image Processing, Random Forest, Gamification, Azure, Epoch.

Acknowledgment

I would like to express our appreciation and full-hearted gratitude to our supervisor **Mrs. Dhammika Perera** and co-supervisor **Ms. Janani Thamalaseelan** who guide us to success this research and for sharing the experiences and expertise with the project matters. We extend our gratitude to the Sri Lanka Institute of Information Technology (SLIIT) for giving an opportunity to carry out this research project. A special thanks to the agriculture department for providing datasets and our friends and families, who help us and gave suggestions for our research. We would like to appreciate the guidance given by other supervisors as well as the members of the panel.

Table of Contents

Declaration	i
Abstract.....	ii
Acknowledgment	iii
Table of Contents.....	iv
List of Tables.....	vi
List of Figures.....	vii
List of Abbreviation	viii
1. INTRODUCTION.....	1
1.1. Background Literature.....	3
1.1.1. Image procesing for plant diseases using basic functions from OpenCV	Error!
Bookmark not defined.	
1.1.2. Image processing done with K Means Clustering	3
1.1.3. Disease detection done by noise reduction and converting RGB to HSV	3
1.1.4. Utilizing Otsu method and Genetic Algorithms	3
1.1.5. Detecting diesased tomato leaves by Kmeans clustering	4
1.1.6. Using ANN for disease detection with Gray Level Co-occurrence Matrix	4
1.2. Research Gap.....	5
1.3. Research Problems.....	6
1.4 RESEARCH OBJECTIVES.....	9
1.4.1 Main Objectives.....	9
1.4.2 Specific Objectives.....	10
2 METHODOLOGY.....	11

Component Overview	11
2.1.1 Resources Needed	15
2.1.1.1 Software Boundaries	15
2.1.1.2 Hardware Boundaries	16
2.1.1.3 Communication Boundaries	17
2.1.1.4 Memory constraints	17
2.1.1.5 Operations	17
2.1.1.6 Site adaptation requirements	18
2.1.1.7 Feasibility Study	18
2.2 Commercialization	Error! Bookmark not defined.
2.2.1 Basic User	20
2.2.2 Premium User	23
2.2.3 Advertisements	23
2.3 Testing and Implementation	26
2.3.1 Testing	29
2.3.2 Implementation	29
3 RESULTS & DISCUSSIONS	38
3.1. Results	Error! Bookmark not defined.
4.2. Research Findings	40
4.3. Discussion	40
5. CONCLUSION	41
REFERENCES	42
APPENDICES	43

List of Tables

Table 1 Comparison of with agripreneurs.....	6
Table 2 Front end Test case1	20
Table 3 Front end Test case2	21
Table 4 Front end Test case3	21
Table 5 Front end Test case4	22
Table 6 Back-end Test case5	22

List of Figures

Figure 1 Bare Land availability.....	7
Figure 2 Overall System Diagram.....	12
Figure 3 Flow diagram of the image processing component.....	13
Figure 4 Detailed flow of the model.....	14
Figure 5 Algorithm Comparison.....	28
Figure 6 Mongo Database.....	32
Figure 7 Initial Screen.....	34
Figure 8 Capture Image Screen.....	35
Figure 9 Upload image from device Screen.....	36
Figure 10 Prediction Screen with alert.....	37

List of Abbreviation

<u>Abbreviation</u>	<u>Description</u>
BSON	Binary JSON
GDP	Gross Domestic Product
SQL	Structured Query Language
ML	Machine Learning
IOT	Internet of Things
IOS	Iphone Operating System
IDE	Integrated Development Environment

1. INTRODUCTION

Entrepreneurship is a broad domain in the modern, continuously technologically advancing world. The world is continuously facing new challenges mainly due to the limited non-renewable resources and other circumstances created by human itself, one of them are global warming, this has vastly increased opportunities in the entrepreneurship domain over the past years. However, the entrepreneurship domain still remains as one of the highest risk domains mainly due to its extremely high uncertainty levels. Furthermore, in a country like Srilanka, this uncertainty level has been skyrocketing in the past years.

Srilanka, which is a developing country has been failing in its economic development which has led to many consequences such as the rupee depreciating, high taxes and import bans to name a few. One of the key sectors, which in Srilanka, that has of high dominance for the past century was agriculture. However, due to various consequences such as climate changes and low controlled prices, the agriculture domain in Srilanka has been gradually failing. This has led to the import of various different vegetables, grains/rice and even fruits from foreign countries such as India to fill the deficit in food in the society.

Moreover, when considering the statistics of the agriculture domain in Srilanka, the agriculture GDP of the country has been declining over the past years [1]. In a developing country where the majority of citizens are considered poor, consists of 75% of agricultural/farmer families that depend on agriculture as their primary source of income. A significant portion of 25% are industrial level farmers. Since there is no mediation between these two farmers categories, the revenue from the agriculture domain goes down drastically. Hence making the agriculture domain technologically advanced so that not only the branded farmers could assist in this revenue but also the everyday person that mostly has ample idle time in their hands are considered.

Bridging the agricultural and the entrepreneurship domain through modern technology is a task that has being sought out to be accomplished in the past years. Due to the pandemic situation the past years, this

Agriculture creates an economic future for developing countries like Sri Lanka, the demand of involvement of modern technologies in agriculture sector is higher. A substantial percentage of the inhabitants of the country depend on the agriculture. But nowadays percentage of total agriculture GDP has been dropping [1]. In Sri Lanka 75% of the families which are depending on agriculture are small farmers from villages and only 25% of them are industrial level farmers. As there is no balanced cultivation between these two kinds of farmers production revenue goes down. Thus, it affects the small farmers the most. So, modernization of agriculture is very important and thus will lead the farmers of our country towards profit.

A bridge between the agricultural and the technological domain will not only allow individuals, to earn an extra buck but also allow users to be engaged in improving the failing Srilankan economy. An excess production in the agricultural domain will lead to the ability of Srilanka to export the extra production hence increasing the export revenue to the country.

The goal is to build a platform which allows individuals that has spare time on their side to successfully involve in the agriculture business hence become self-made entrepreneurs. This is being enabled by the creation of a motivational gamification model where users are being continuously involved in the process. The application will hence consist of 3 main areas which are the motivational gamification model, IoT soil data analysis model, robust image processing model and finally a prediction auction platform.

1.1. Background Literature

In the past years, many applications and platforms are being created to achieve this in the agriculture domain throughout the world. Most of these techniques are been well utilized in the agriculture domain in order to make it technologically advanced making it easier for both parties which are the farmers and the buyers/government.

1.1.1. Image processing for plant disease detection done with basic functions in OpenCV

Research conducted presents classification and detection techniques that can be used for plant leaf disease classification. Here preprocess is done before feature extraction. RGB images are converted into white and then converted into grey level image to extract the image of vein from each leaf. Then basic Morphological functions are applied on the image. Then the image is converted into binary image. After that if binary pixel value is 0 its converted to corresponding RGB image value. Finally, by using pearson correlation and Dominating feature set and Naïve Bayesian classifier disease is detected [3]

1.1.2. Image processing done with K-means clustering for segmentation

In another research done there were four steps. Out of them the first one is gathering image from several part of the country for training and testing. Second part is applying Gaussian filter is used to remove all the noise and thresholding is done to get all green color component. K-means clustering is used for segmentation. For the feature extraction process only a one simple technique was used by converting the RGB image into an HSV one.[4]

1.1.3. Disease's detection using noise reduction and converting image from RGB to HSV

The paper [5] presents the technique of detecting jute plant disease using image processing. Image is captured and then it is realized to match the size of the image to be stored in the database. Then the image is enhanced in quality and noises are removed. Hue based segmentation is applied on the image with customized thresholding formula. Then the image is converted into HSV from RGB as it helps extracting region of interest. This approach proposed can significantly support detecting stem-oriented diseases for jute plant.

1.1.4. Using Otsu method for segmentation and Genetic algorithm for classification.

According to paper [6] they have proposed for a technique that can be used for detecting paddy plant disease by comparing it with 100 healthy images and 100 sample of disease1 and another

100 sample of disease2. It's not sufficient to detect disease or classify it training data is not linearly separable. In paper [7] detection of unhealthy plant leaves includes some steps are RGB image acquisition. Converting the input image from RGB to HSI format. Masking and removing the green pixels. Segment the components using Ostu's method. Computing the texture features using color-co-occurrence methodology and finally classifying the disease using Genetic Algorithm.

1.1.5. Detecting diseased tomato using K-means clustering by color differentiation

Paper [8] includes tomato disease detection using computer vision. A gray scale image is turned into binary image depending on threshold value. The threshold algorithm is used for image segmentation. The threshold values are given color indices like red, green, blue. But the thresholding is not a reliable method as this technique only distinguishes red tomatoes from other colors. It becomes difficult to distinguish ripe and unripe tomatoes. For this K-means clustering algorithm is used to overcome the drawbacks. K-means create a particular number of nonhierarchical clusters. This method is numerical, unsupervised, non-deterministic and iterative. Then separating the infected parts from the leaf, the RGB image was converted into YcbCr to enhance the feature of the image. The final step is the calculation of the percentage of infection and distinguishing the ripe and unripe tomatoes.

1.1.6. Using ANN for plant disease detection through Gray Level Co-occurrence matrix

The methodology for cucumber disease detection is presented in paper [9]. The methodology includes image acquisition, image preprocessing, feature extraction with gray level co-occurrence matrix (GLCM) and finally classified with two types: Unsupervised classification and supervised classification. Paddy plant is an important plant in continental region.

1.2. Research Gap

The overall approach of the application is to motivate them to be involved in their free time in this agriculture domain through the application. A motivational application builds up has not being well implemented in commercial business focused application, however social media application such as facebook, Instagram uses diverse models in order to control their user's actions within the application and make them use the application more unconsciously. Gamification models are also built widely and are being utilized mainly into gaming applications where the user's performance is being accessed in those games itself and are categorized as a leaderboard within the application.

The image processing domain is of the most popular domains in the 21st century, with the constant advancements and improvements in the machine learning. Leaf detection has been done widely in different domains utilizing various different algorithms and techniques. Some of the most prominent being using edge detection techniques, and other applications such as Matlab. The most prominent algorithms utilized when building a training model for image processing are mainly K-Means clustering, image preprocessing techniques such as histogram equalization. Image segmentation techniques used to separate pixels according to their color intensities are mainly done using the K-Means clustering algorithm. Some research and leaf disease detection algorithms are being trained by using Matlab.

The main goal of most of these disease detection researches and other application that were built were to just detect the disease, however stepping forward in suggesting the user with appropriate solutions for the relevant diseases have not been properly implemented mainly because of the models that were built were only to detect a disease in a single type of plant, hence it wasn't practical to build a machine learning recommendation model for this.

Similar applications were built mainly for the widescale industrial farming families to make their tasks easier, some of the most popular ones were FieldCheck App and Descrates Crop App.

Features	FieldCheck App	Descartes Crop App	Agriprenuers
Crop visualization	Y	N	Y
Yield Forecast	N	Y	Y
Trending Crops	N	N	Y
Target Audience	Farmers	Farmers	Anybody/Entrepreneurs
Suitable Crop recommendation	N	N	Y
Disease detection and solutions	N	N	Y
Auction Platform	N	N	Y
IoT analysis model	N	N	Y
Motivational Model	-	-	Y

Table – 1: Comparison of existing applications with agriprenuers application

1.3. Research Problems

The research problem identified revolves around how a specific individual will be motivated by using the application, agriprenuers. The problem identified is the lack of entrepreneurship in developing countries or any other country across the world is due to the lack of motivation and the inability to have a stable level of trust on the market they will be involved in.

Furthermore, an entrepreneur will also be facing many other risks and other valuable skills which should be properly addressed before stepping onto a high risky high reward domain.[1]

These include:

1. Leadership
2. Positive mentality
3. Identifying business opportunities

In a developing country like Srilanka, these risks are further high lightened. Hence, many of these entrepreneurs will be seeking for external support to make their business ventures easier for them. There is no proper platform in Srilanka that will provide every tool for every different entrepreneur to succeed in their specific business.

Out of all available businesses, agriculture is a prominent domain that most individuals are interested in, especially in the rural areas due to high availability of bare land. However, a handful of families and individuals residing in the urban areas possess the bare land and it is not utilized. This is a main problem as most families and individuals have expenditures way more than their monthly income. Hence, this issue could be resolved by triggering the entrepreneurial spirit within them via our application.

The figure below shows the availability of bare land in the Malabe area of Srilanka.

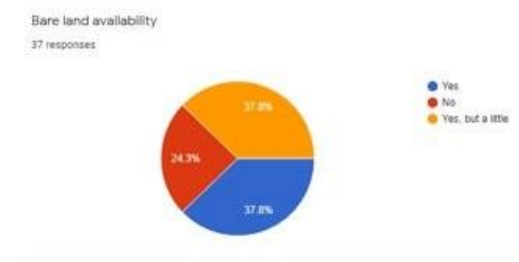


Figure 1: Bare Land availability

Agriculture is a domain not only beneficial for those individuals carving to become entrepreneurs but also to the country as it will uplift the country's economy.

Furthermore, due to the current pandemic situation most families especially residing in the urban areas are facing a considerable risk of shortages of vegetables and other dry food due high demand and low supply. Hence, we believe a considerable number of problems can be resolved to a greater extent by motivating individuals to become entrepreneurs via the agricultural domain.

A survey conducted by our research group also revealed that most individuals residing in the Colombo and Gampaha districts had an average monthly income less than 50,000 each month with an expenditure close to that income value. Furthermore, the agriculture department of Sri Lanka has the prices of crops and their prices according to the year and month. However, this data cannot be accessed by the people easily. Hence, our system will have both buyers and sellers on the same platform which will automatically generate the most appropriate price.

The main component focused by me is the creating a model which will show the users the success rate of the crops according to the time and effort invested from their spare time and benefit or profit that can be gained by such devotion. This is a major problem because although an individual will be invested in the business, providing a result according to the effort needed and benefit or success gained will motivate those individuals due to the positive impact it will have on them to add extra effort.

Moreover, the individuals involved in the agricultural processes will be facing other external unavoidable conditions such as soil nutrition conditions or plant diseases or even pest attacks. Hence another identified problem is that if such instance occurs, the motivation of those individuals towards the crops will reduce as extra care must be taken to get the maximum profits without running on losses.

The tools provided by the application will aid the users to gain profits from their first venture without loss as most entrepreneurial businesses focus on losses initially to break even and finally to making a proper profit if fortunate.

The problem does not focus on being a competition to the main stakeholder of the agriculture domain but to the families with bare under-utilized land residing in the urban areas. Furthermore, our problem will also aid the other stakeholders of the agriculture business which includes buyers, agrochemical companies, seed importers and other financial institutions.[2]

1.4 RESEARCH OBJECTIVES

The primary goal of this project is to generate a platform which will create more entrepreneurs in the country by bridging the agriculture and the entrepreneurship domain through technology.

The 3 main components of the whole application should be interconnected with each other and sharing data, to make accurate prediction results to the user. The image processing model, should have the ability to be used robustly hence allowing users to detect diseased plants from a wide variety of leaves, and even when the user uploads or captures a picture of low resolution or quality, the model should be able to detect this and show it to the user with accurate results along with recommendations as of to overcome the diseased condition.

1.4.1 Main Objectives

The main objective is to allow users to trust functionalities of the platform and motivate them to be always engaged in the application with ease. This will be achieved by a range of approaches from predicting figures and solutions for the users of the application.

The main objective achieved in the specific research component is to allow users detect plant diseases of the crops grown. The image processing model that will be developed can be used robustly, hence the image processing model will be able to detect plant diseases from a wide range of crops that the user will be choosing.

The detected plant diseases will be accessed and shown to the user. Together with the plant disease the user will also be show the necessary steps, solutions, recommendations or fertilizers to the user.

The image processing model is that which can be use robustly or on a wide range of plants. The image processing system will hence also be using a huge set of data for its training model in order to achieve this.

1.4.2 Specific Objectives

- Collecting images for the training model for a variety of crops. (Fruits and Vegetables)
- Creation of different classes according to the type of crop and further if the image is a diseased or healthy leaf according to the type of crop.

- **Finding the appropriate algorithms and approaches for image segmentation and classification, and prediction of leaf type and disease.**
- Building the training model.
- Testing the model against datasets and obtaining accuracy scores for different classes of images.
- Building the react native expo application with the flask backend.
- Deploy it to cloud (Microsoft Azure, Virtual Machine)
- **Integrate fertilizer and recommendation predictor to the image processing model.**
- Test the application for expected output results.
- Integrate it with the gamification model application.

2 METHODOLOGY

Component Overview

Agripreneurs, is a mobile based web application which will allow users to be motivated to be engaged in the agricultural field and become self-made entrepreneurs. This application which is being developed consists of 4 main areas which are the gamification motivational layout of the application developed through psychology, an IoT data analysis model where users will be shown the accurate soil and analyzed climatic data, a robust image processing model which can predict solutions according to the detected diseased plant/crop, an integrated buyer sell platform where users will be shown the trending crops through machine learning.

The component that is being developed here is the robust image processing model. The individuals/users of the application will be having a relatively low knowledge level on how the crop they chose will be grown apart from just planting it in the respective area of land.

With time being passed, the users will be forced to face uncontrollable conditions and situations such as nutrient deficiencies, pest attacks etc. This image processing model being developed

can be robustly used, meaning a rough image of the plants leaf can be uploaded to the model and the appropriate results will be shown to the user. The user must first identify if the plant leaf image that is being uploaded is actually that of a diseased plant, following this the user will also be given the appropriate recommendations via a prediction model on what should be applied to the diseased plants.

This will be integrated into the gamification model application which will help users be engaged and motivated in the activities throughout the period of the process.

2.1 Methodology

A high-level diagram of the agriprenuers motivational platform is shown below

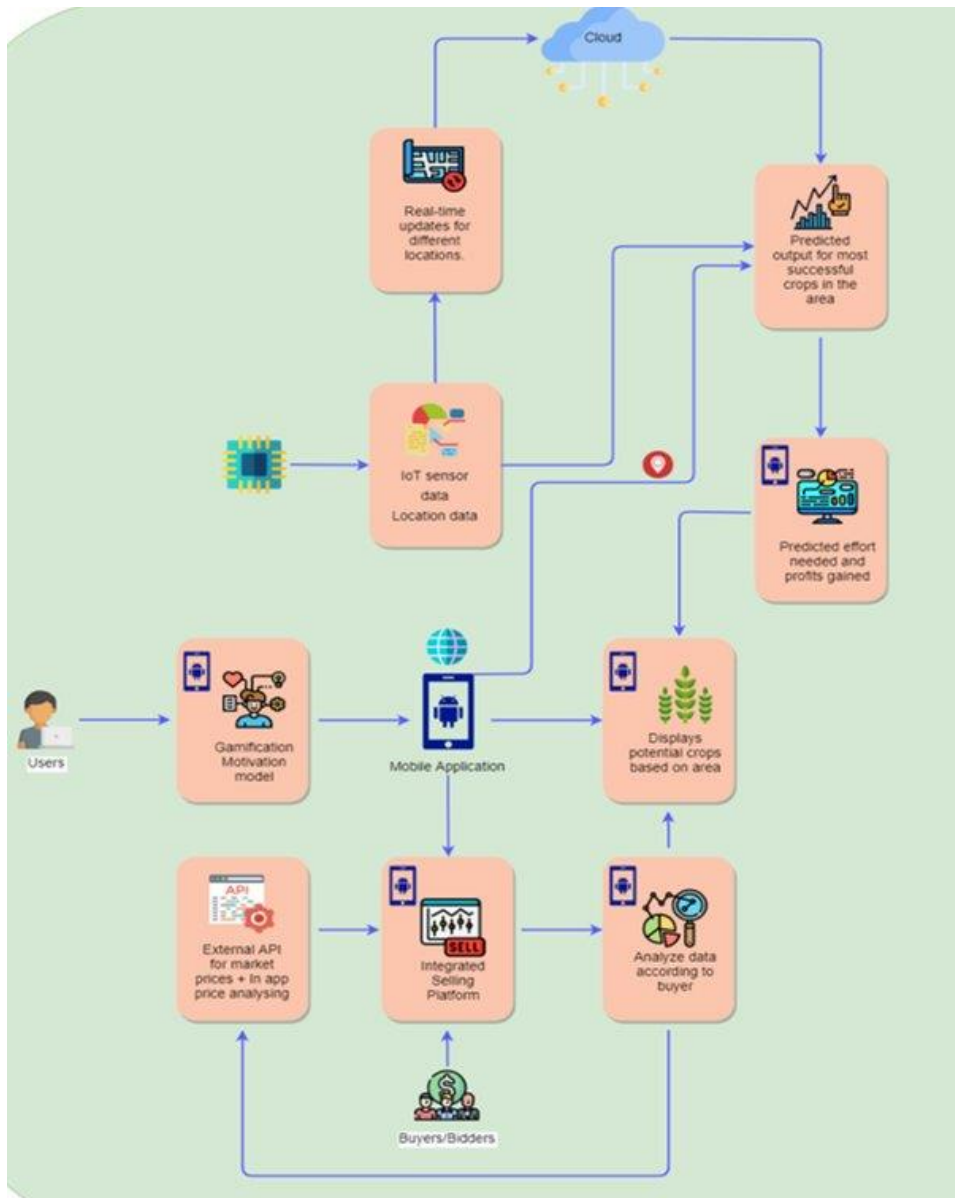


Figure 2: Overall System Diagram

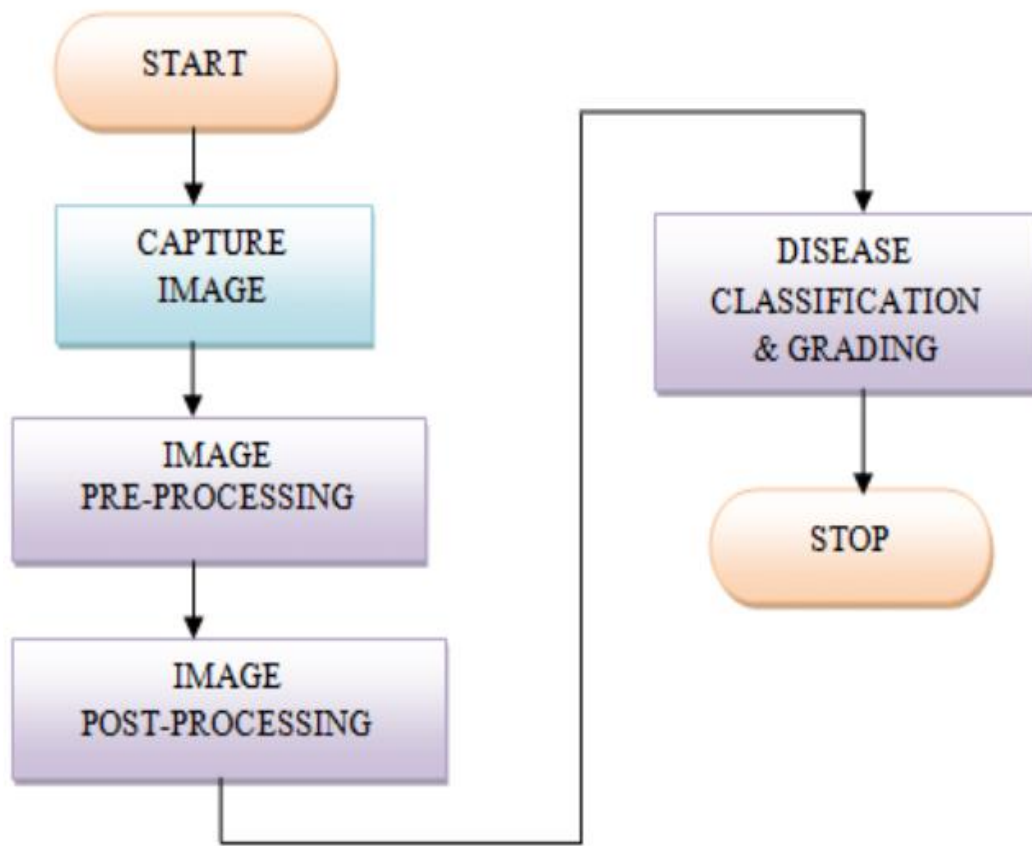


Figure 3: Flow Chart on Image processing Model

The basic and main flow of the robust image processing model can be visualized by the flow chart above. Here the capture image is a basic functionality which will capture the image from any device and pass it forward to the next segment, where the uploaded image will be preprocessed accordingly to the training model. In the training model, all the necessary steps for the image detection, classification and prediction will be done. Hence after this step, the model training model will be able to detect if the uploaded image is of what genre, which are one of the two diseased or healthy.

The model build is trained across different image classes of different diseased and healthy crop leaves. The trained model is then exported and deployed in the flask backend application, which is then hosted on Microsoft azure. The database that will be used is Atlas Mongo Database to save the images.

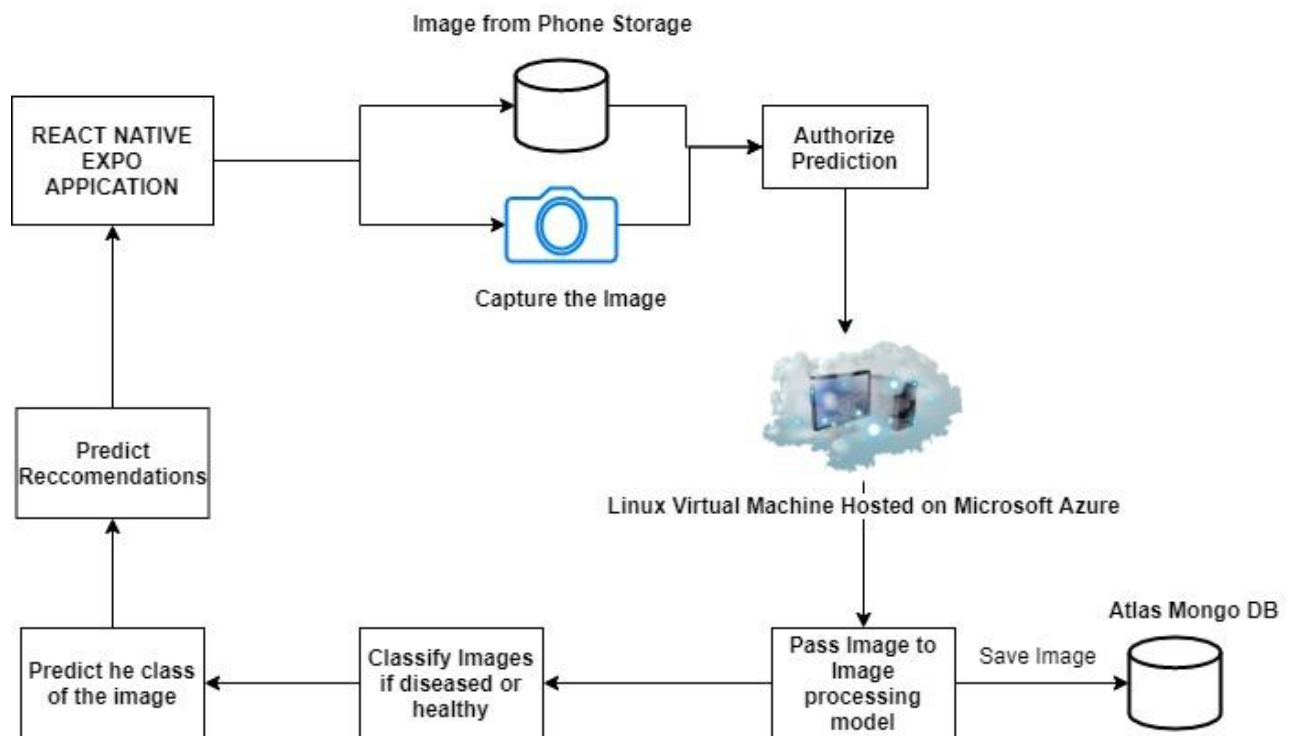


Figure 4: Detail flow of the model

When building any kind of image processing or machine learning model, firstly the appropriate data must be collected to be fed into the model. The model that will be built is going to be a robust image processing and prediction system, where the model will be able to detect a wide range of crops. Identify if they are diseased or healthy, and finally provide appropriate results for the users.

The datasets are mainly divided into 14 different fruits and vegetable classes, and the most commonly occurring diseases in each of those fruits and vegetables are being addressed. The model will hence be built on these classes. A total of 39 different classes will be fed into the training model.

2.1.1Resources Needed

2.1.1.1Software Boundaries

1. Jupyter Notebook

This is one of the most popular python development tools used in the machine learning domain. It has an easy and low resource usage environment where the application is run like a normal local host application. Easy to utilize and import python libraries and check outputs.

2. Visual Studio Code

One of the most prominent programming tools in the modern world which supports coding and implementation of any type. React Native Expo application mainly utilized this IDE. Furthermore, the flask backend application was also developed using visual studio code.

3. Atlas MongoDB

This is a NoSQL database, where the data is being saved in cluster databases in form of collections. The Atlas MongoDB is hosted in a cloud platform which provides high reliability for the application.

4. Anaconda Environment

This is mainly used for importing python libraries and to install various libraries and packages that will be needed when running the image processing model and building the training model.

5. Expo Go application

This is a by default installation application being provided by the react native expo development kit. This application can be installed on any smartphone platform, android or IOS. Users can scan the QR code from the developed ExpoGo application and instantly run it on any mobile device the has the ExpoGo client application installed on their phone.

6. Microsoft Azure

This is one of the most popular cloud computing platforms which provides free services and credits for university students. This cloud platform was utilized to host the back-end flask application hence it could be easily deployed and utilized by the react native expo client application.

7. Kaggle

This is a website which was used to obtain training images and testing images for the image processing model, furthermore different classes of images was also obtained by using this website. For example, different types of berries, tomatoes etc.

8. Android Studio

This is mainly used for the emulator functionality, if an issue runs with the smartphone being used to test the application.

9. Programming Languages

The main programming languages that will be used in the development of the backend framework is Python 3, the Flask framework by python is used for easy deployment of the image processing model and to consume it easily from the front-end application.

The frontend of the application is run on react native, which is a widely used cross platform JavaScript framework.

2.1.1.2 Hardware Boundaries.

- If the application's backend flask server is being run on locally itself, higher resources are being demanded by the server computer when running the actual application itself. However, if the cloud servers fail or run out of credit. The model can be hosted locally on the wifi network using a locally hosted MongoDB.
- If the react native expo application is being run by using a simulator phone running android or IOS, the ram usage will be relatively high. Minimum of 4GB of ram is being required for this functionality.
- The NodeMCU model that will be built to detect the soil and air conditions should be purchased to have more benefits from the application.

2.1.1.3 Communication Boundaries

An internet connection is mandatory for this application to function, since the application runs of react native expo client. Furthermore, the backend flask and the mongo database are both hosted in cloud servers. To access these the application must have a stable internet connection. Location services are also utilized by the application where necessary.

Only one language will be provided which is English.

2.1.1.4 Memory constraints

As this application involves ML. ML in the form of image processing utilizes a lot of memory as well as storage to store the training and testing images when building the models.

Back-end

RAM - 8 GB minimum, 16 GB or higher is recommended.

GPU - NVIDIA GeForce GTX 960 or higher.

Data Storage – 5GB

Front-end (Android Application)

RAM – 1GB - 2GB

Data Storage – 500 Mb or higher.

2.1.1.5 Operations

User is capable for following operation.

- Capture Images: The user can select the image processing interface and capture images if needed.
- View Captured image: The user can view the captured images.
- Input Image: The user can input a previously saved image in the phone storage
- Predicting: The user can click on the prediction button to obtain predicted plant type and the relevant disease type and type of plant. The user will also be shown with appropriate recommendations.
- Save image: The captured image or the image from the database will be saved in the cloud atlas Mongo database.

2.1.1.6 Site adaptation requirements

Main language of the application is English for initial release. Local languages Tamil and Sinhala will be available in future versions. Also, android device must connect to internet to access back-end image processing system which is hosted on server. An internet connection is also needed to communicate with database and external map API servers.

Since Agripreneurs is the portable application, it's not depended on any other external dependencies, so any other site adaptation is not necessary.

2.1.1.7 Feasibility Study

The primary goal of the application is to build a stable profit gaining for individuals that has both spare land and spare time in Srilanka.

This application is a free to purchase application hence it can be used by anybody regardless they are farmers or normal everyday people.

Since the backend application which is the flask API framework is hosted on the azure cloud platform and monthly cost of \$24 will be charged. This has been covered up by the free azure credit being provided for the student development kit itself. However, if the application is to be further enhanced to higher scales the virtual machine size must be increased which will relatively increase the cost. This will only happen if many users join the application platform.

Since it is an android application following technologies and tools are used to develop the system. The application can be hosted and tested on any platform as react native is a platform independent programming language.

- Jupyter Notebook
- Visual Studio Code
- MongoDB (No SQL)
- Android studio Emulator
- Android mobile with ExpoGO client

Each of the technologies are freely available and all required technical skills are manageable.

2.2Commercialization

2.2.1 Basic User

Agripreneurs is a free application on its initial release. However, the application has its own integrated buyer seller platform. This feature can be accessed up to a certain level by the individual users of the application. All the components of the application can be accessed by the free users, this includes the IoT data analysis model which has to be purchased separately by the users. The gamification model, image processing model and the limited auction platform will be fully accessible by all users.

2.2.1 Premium User

The premium user of the application will be able to view data visualized in the application as an added feature. Furthermore, the users can export their reports and analyze the data as needed. The users can purchase the premium version to make better decisions on prices and crops that should be grown according to historical data of the application. Furthermore, the application will be taking a percentage fee of all deals that are done through the buyer seller platform. Here, the percentage of the cut that will be taken will be reduced from the user if the upgrade to the premium version. The initial percentage cut will be 10% which will be reduced to 7.5% upon being a premium user.

2.2.2 Advertisements

An integrated advertisement platform is also implemented to show advertisements through google ad sense, here an extra buck can be earned by the developers of the application easily. Furthermore, other agricultural products and equipment can also be sold through the integrated buyer-seller auction platform if needed.

In the initial phase of the application, only the agriculture domain is being considered. However, the system can be broadened to a wide number of domains such as textiles, floral arrangements, ornaments etc. Any type of hand made items can also be integrated into the platform. Furthermore, dry food such as rice, gram food items can also be added to the system. Hence the farmers of the country can also get full advantage from the system.

2.3 Testing & Implementation

2.3.1 Testing

Software testing is very important in development lifecycle to point out the defects and errors that were made during the development phases. And also, it is very important to ensure the quality of the product. Testing is required for an effective performance of software application or product. Testing is required at every phase that we are following in software development life cycle. These mainly involved in single unit tests, functionality testing and integration testing with the other components of the application.

For the individual system, there are two main parts.

React Native Application (Frontend)

Flask Backend model with image processing model (Backend)

Front-end Testing: Following are some test cases conducted to confirm the application runs with ease and no failures.

Test case ID	001
Test case scenario	Check user login with valid credentials
Test steps	a. User enters e-mail id b. User enters password c. User hits the Login button
Test Data	E-mail ID= lochanarangalla@yahoo.com Password= abc123!@
Expected Results	The user should log in into the application.
Actual Results	User logs into the application
Pass/ Fail	Pass

Table 2 Front end Test case 1

Test case ID	002
Test case scenario	Check user login with invalid credentials.
Test steps	a. The user enters e-mail id b. User enters password c. User hits the Login button
Test Data	E-mail ID= lochanarangalla@yahoo.com Password= abcd123
Expected Results	The user should not log into the application.
Actual Results	The user should not log into the application.
Pass/ Fail	Pass

Table 3 Front end Test case2

Test case ID	003
Test case scenario	User being able to access the common home screen
Test Steps	a. User logs into the application b. User navigates to the home screen c. User clicks on image processing component to navigate to image processing interface screen
Test Data	Navigation clicks
Expected Results	User Successfully logs into the application and navigates to the image processing screen

Actual Results	User Successfully logs into the application and navigates to the image processing screen
Pass/ Fail	Pass

Table 4 Front end Test case3

Test case ID	004
Test case scenario	Capture image from camera component
Test steps	a. User navigates to camera component in the image processing screen b. User captures an image
Test Data	Navigation to camera component and capture an image
Expected Results	Alert shown as an image has been captured.
Actual Results	Alert shown as an image has been captured.
Pass/ Fail	Pass

Table 5 Front end Test case4

Test case ID	005
Test case scenario	Upload image from the device storage
Test steps	a. User finds image from the image storage b. User can download images from an online website
Test Data	Selecting images
Expected Results	Alert shown as image has been uploaded to the model
Actual Results	Alert shown as image has been uploaded to the model
Pass/ Fail	Pass

Table 6 Front end Test case51

Test case ID	006
Test case scenario	Predicting through image processing model.
Test steps	a. Enter the image processing model interface b. Chose image from the phone or capture the image from the phone. c. Enter the image to the model by clicking the predict button.
Test Data	Capture image, or image from storage is sent to the backend from the frontend
Expected Results	Alert message with diseased or healthy plant with recommendations
Actual Results	Alert message with diseased or healthy plant with recommendations
Pass/ Fail	Pass

Table 7 Back-end Test case

2.3.2 Implementation

2.3.2.1 Data collection

The image processing component mainly involves around the type of data that is being collected. This image processing has the ability to be use robustly. To achieve this, a huge data set from different types of classes in both the main categories which are the diseased and healthy plant leaves must be obtained. The images have been obtained from more than 39 different classes which includes different vegetables and fruit leaves, and a few of the most commonly occurred diseases for these plants. The summing up of images totals up to 3 gigabytes of data.

The dataset classes are as follows,

Apple___Apple_scab
Apple___Black_rot
Apple___Cedar_apple_rust
Apple___healthy
Blueberry___healthy
Cherry_(including_sour)___Powdery_mildew
Cherry_(including_sour)___healthy
Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot
Corn_(maize)___Common_rust_
Corn_(maize)___Northern_Leaf_Blight
Corn_(maize)___healthy
Grape___Black_rot
Grape___Esca_(Black_Measles)
Grape___Leaf_blight_(Isariopsis_Leaf_Spot)
Grape___healthy
Orange___Haunglongbing_(Citrus_greening)
Peach___Bacterial_spot
Peach___healthy
Pepper,_bell___Bacterial_spot
Pepper,_bell___healthy
Potato___Early_blight
Potato___Late_blight

Potato___healthy
Raspberry___healthy
Soybean___healthy
Squash___Powdery_mildew
Strawberry___Leaf_scorch
Strawberry___healthy
Tomato___Bacterial_spot
Tomato___Early_blight
Tomato___Late_blight
Tomato___Leaf_Mold
Tomato___Septoria_leaf_spot
Tomato___Spider_mites Two-spotted_spider_mite
Tomato___Target_Spot
Tomato___Tomato_Yellow_Leaf_Curl_Virus
Tomato___Tomato_mosaic_virus
Tomato___healthy

Table --: Table of the classes of different images of diseased and healthy leaves

As show in the table above, distinct images from 14 different vegetables and fruits are being obtained. The model will be trained against these training classes.

2.3.2.2 Data Preprocessing

The initial part of the image processing model is for it to identify the images, that the uploaded images are of that of a plant leaf. Hence detecting the uploaded images as that of a plant leaf has to be done. This has been achieved using 3 feature extraction mechanisms which includes, Hu moments, Haralick method and Color histogram.

Hu Moments

Hu moments is a feature detection functionality which is being provided by the Sklearn open library for machine learning functionalities using python. Hu moments is used to detect the shape of the images that are being uploaded, this is firstly utilized in the training model where images are classified as that of leaves.

Most of the images uploaded will be in the RGB (Red Green Blue), hence firstly the images will be converted into binary format so that it is easy to find the shapes of the images from the change in intensity levels.

The basic equation of the functionality of Hu moments being utilized here is,

$$M = \sum_x \sum_y I(x, y)$$

Where I is the intensity level of each pixel. X and Y represents the location of that pixel in the image.

Utilizing this functionality in the training model, it will appear in code as follows.

```
#feature descriptor1: Hu moments (yellowish/fire)
def fd_hu_moments(image):
    image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    feature = cv2.HuMoments(cv2.moments(image)).flatten()
    return feature
```

Mainly, the uploaded images will be checked against the yellowish or brown discolorations to identify if the leaves are diseased or healthy.

Haralick Textures

This is a texture identifying functionality provided by the Sklearn open library for machine learning in python. The textures are being identified by a gray level co-occurrence matrix. Basically, it has the ability to count the co-occurrence of neighboring pixels hence accessing the gray levels of the image.

The implementation of this functionality to access the relative texture of the leaf images being fed into the training model are shown below,

```
#feature descriptor2: Haralick texture(Quantify image according to the texture)
def fd_haralick(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    haralick = mahotas.features.haralick(gray).mean(axis = 0)
    return haralick
```

Color Histogram

Color histogram is also another functionality provided by open cv for python machine learning processes. Here, the color gradient of the images are accessed and made into a histogram where leaf images can be accessed easily.

The implementation of the functionality will be as follows,

```
#featur descriptor3: Color Histogram  
def fd_histogram(image, mask = None):  
    image = cv2.cvtColor(image,cv2.COLOR_BGR2HSV)  
    hist = cv2.calcHist([image],[0,1,2],None,[bins,bins,bins],[0,256,0,256,0,256])  
    cv2.normalize(hist,hist)  
    return hist.flatten()
```

The model is then being tested with 80% of the image data made into the training image processing model and 20% of the images into the validation or testing model. This will increase the accuracy of the model that will be built.

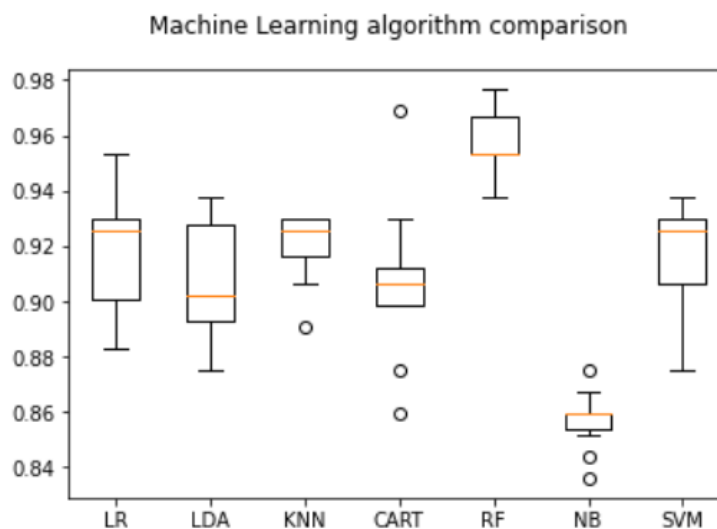


Figure 5: Algorithm Comparison

Finally, the results from the model are being trained across 10 epochs in the first rounds and using 6 different algorithms. The algorithms used are Naïve Bayes, Random Forest, Support Vector algorithm, k-nearest neighbors' algorithm and Latent Dirichlet allocation algorithm.

From all the algorithms tested, the random forest algorithm gave the highest percentage accuracy hence this was used to build the training model. The above graph shows the model being tested on only 2 classes which are normal apple leaves and diseased apple leaves, hence this provided us with a high accuracy level of 95%, it is expected that the accuracy level would reduce when the number of classes inserted into the model is being increased.

The output of this process is the testing model which will be integrated with the backend flask api and hosted in the cloud space.

2.3.2.3 Model Creation and Training

By using the 38 image classes of both diseased and healthy plant leaves, the image processing model was built. Here, the random forest algorithm was utilized in order to predict if the output image was that of a diseased plant or a healthy plant.

The output of the tested model runs on an accuracy of 70% as the number of classes fed into the model were gradually increased.

The training model was tested on 150 epochs, on the random forest algorithm. A final accuracy output of 70% was obtained as shown below.

```

Epoch 1/10
150/150 [=====] - 417s 3s/step - loss: 7.0892 - accuracy: 0.0561 - val_loss: 0.0822 - val_accuracy: 0.9717
Epoch 2/10
150/150 [=====] - 345s 2s/step - loss: 5.8732 - accuracy: 0.0612 - val_loss: 0.0945 - val_accuracy: 0.9685
Epoch 3/10
150/150 [=====] - 331s 2s/step - loss: 5.3996 - accuracy: 0.0636 - val_loss: 0.1110 - val_accuracy: 0.9645
Epoch 4/10
150/150 [=====] - 321s 2s/step - loss: 5.1039 - accuracy: 0.0656 - val_loss: 0.1275 - val_accuracy: 0.9609
Epoch 5/10
150/150 [=====] - 308s 2s/step - loss: 4.8865 - accuracy: 0.0745 - val_loss: 0.1389 - val_accuracy: 0.9602
Epoch 6/10
150/150 [=====] - 298s 2s/step - loss: 4.7639 - accuracy: 0.0699 - val_loss: 0.1483 - val_accuracy: 0.9602
Epoch 7/10
150/150 [=====] - 300s 2s/step - loss: 4.6131 - accuracy: 0.0775 - val_loss: 0.1589 - val_accuracy: 0.9598
Epoch 8/10
150/150 [=====] - 289s 2s/step - loss: 4.6213 - accuracy: 0.0657 - val_loss: 0.1751 - val_accuracy: 0.9572
Epoch 9/10
150/150 [=====] - 292s 2s/step - loss: 4.5059 - accuracy: 0.0702 - val_loss: 0.1877 - val_accuracy: 0.9547
Epoch 10/10
150/150 [=====] - 290s 2s/step - loss: 4.4780 - accuracy: 0.0708 - val_loss: 0.1941 - val_accuracy: 0.9550

```

The below diagram shows the addition of all 38 classes to the final built model,

```

class_dict = training_set.class_indices
print(class_dict)
Python
... {'Apple__Apple_scab': 0, 'Apple__Black_rot': 1, 'Apple__Cedar_apple_rust': 2, 'Apple__healthy': 3, 'Blueberry__healthy': 4,
'Cherry_(including_sour)__Powdery_mildew': 5, 'Cherry_(including_sour)__healthy': 6, 'Corn_(maize)__Cercospora_leaf_spot Gray_leaf_spot': 7,
'Corn_(maize)__Common_rust': 8, 'Corn_(maize)__Northern_Leaf_Blight': 9, 'Corn_(maize)__healthy': 10, 'Grape__Black_rot': 11,
'Grape__Esca_(Black_Measles)': 12, 'Grape__Leaf_blight_(Isariopsis_Leaf_Spot)': 13, 'Grape__healthy': 14, 'Orange__Haunglongbing_(Citrus_greening)': 15,
'Peach__Bacterial_spot': 16, 'Peach__healthy': 17, 'Pepper,_bell__Bacterial_spot': 18, 'Pepper,_bell__healthy': 19, 'Potato__Early_blight': 20,
'Potato__Late_blight': 21, 'Potato__healthy': 22, 'Raspberry__healthy': 23, 'Soybean__healthy': 24, 'Squash__Powdery_mildew': 25,
'Strawberry__Leaf_scorch': 26, 'Strawberry__healthy': 27, 'Tomato__Bacterial_spot': 28, 'Tomato__Early_blight': 29, 'Tomato__Late_blight': 30,
'Tomato__Leaf_Mold': 31, 'Tomato__Septoria_leaf_spot': 32, 'Tomato__Spider_mites Two-spotted_spider_mite': 33, 'Tomato__Target_Spot': 34,
'Tomato__Tomato_Yellow_Leaf_Curl_Virus': 35, 'Tomato__Tomato_mosaic_virus': 36, 'Tomato__healthy': 37}

li = list(class_dict.keys())
print(li)
Python
... ['Apple__Apple_scab', 'Apple__Black_rot', 'Apple__Cedar_apple_rust', 'Apple__healthy', 'Blueberry__healthy', 'Cherry_(including_sour)__Powdery_mildew',
'Cherry_(including_sour)__healthy', 'Corn_(maize)__Cercospora_leaf_spot Gray_leaf_spot', 'Corn_(maize)__Common_rust',
'Corn_(maize)__Northern_Leaf_Blight', 'Corn_(maize)__healthy', 'Grape__Black_rot', 'Grape__Esca_(Black_Measles)',
'Grape__Leaf_blight_(Isariopsis_Leaf_Spot)', 'Grape__healthy', 'Orange__Haunglongbing_(Citrus_greening)', 'Peach__Bacterial_spot', 'Peach__healthy',
'Pepper,_bell__Bacterial_spot', 'Pepper,_bell__healthy', 'Potato__Early_blight', 'Potato__Late_blight', 'Potato__healthy', 'Raspberry__healthy',
'Soybean__healthy', 'Squash__Powdery_mildew', 'Strawberry__Leaf_scorch', 'Strawberry__healthy', 'Tomato__Bacterial_spot', 'Tomato__Early_blight',
'Tomato__Late_blight', 'Tomato__Leaf_Mold', 'Tomato__Septoria_leaf_spot', 'Tomato__Spider_mites Two-spotted_spider_mite', 'Tomato__Target_Spot',
'Tomato__Tomato_Yellow_Leaf_Curl_Virus', 'Tomato__Tomato_mosaic_virus', 'Tomato__healthy']

train_num = training_set.samples
valid_num = valid_set.samples
Python

```


Loading the main model is shown below

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
import numpy as np

model=load_model('train_data.hdf5')
model.load_weights('weights.hdf5')
model.summary()
```

Model: "sequential_1"






Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 54, 54, 96)	34944
max_pooling2d_1 (MaxPooling2D)	(None, 27, 27, 96)	0
batch_normalization_1 (Batch Normalization)	(None, 27, 27, 96)	384
conv2d_2 (Conv2D)	(None, 17, 17, 256)	2973952
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 256)	0
batch_normalization_2 (Batch Normalization)	(None, 8, 8, 256)	1024
conv2d_3 (Conv2D)	(None, 6, 6, 384)	885120
batch_normalization_3 (Batch Normalization)	(None, 6, 6, 384)	1536
conv2d_4 (Conv2D)	(None, 4, 4, 384)	1327488
batch_normalization_4 (Batch Normalization)	(None, 4, 4, 384)	1536

2.3.2.4 Deploying the model

The machine learning testing model that had been built first only worked locally in the computer. Firstly the model was deployed on the local wifi network to test the mongo database that was also hosted on the local network itself. This was successfully done, however for the initial frontend of the application the model was tested using a python graphic user interface.

This was finally followed by the development of the graphical user interfaces on the react native application for the user. Simultaneously the model was deployed on Azure cloud on a virtual machine running Linux Ubuntu. The backend database was also transmitted to atlas mongo database, which is a cloud hosted database.

The azure virtual machine portal looks as follows:

Name	Type	Last Viewed
 UbuntuServer	Virtual machine	a few seconds ago
 UbuntuServer	Resource group	2 weeks ago
 UbuntuServer-ip	Public IP address	2 weeks ago
 Azure for Students	Subscription	2 weeks ago
 flaskapp	Resource group	2 weeks ago

The database was also hosted in atlas cloud space, making a new cluster database on the cloud server:

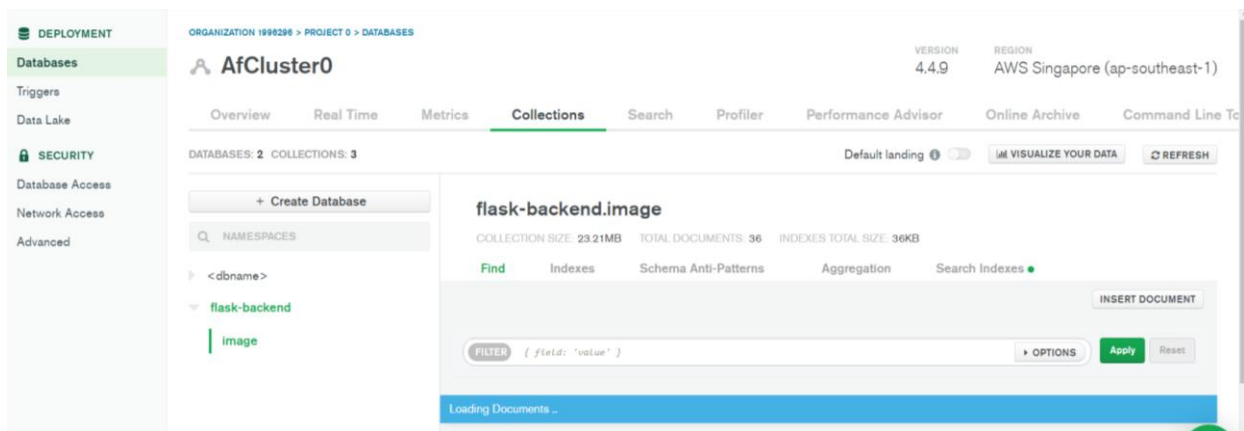


Figure 6: Mongo Database

As shown in the image above, the database cluster name is called “AfCluter0” and the database storing our image collections is called the flask-backend database.

As the front-end react native application is automatically deployed through the expo client, separate deployment was not necessary.

2.3.2.4 User Interfaces

The application was built using the react native expo front-end framework, this framework is cross platform compatible hence can be run on both android and IOS devices.

The below figure shows the main UI that was used for the development.

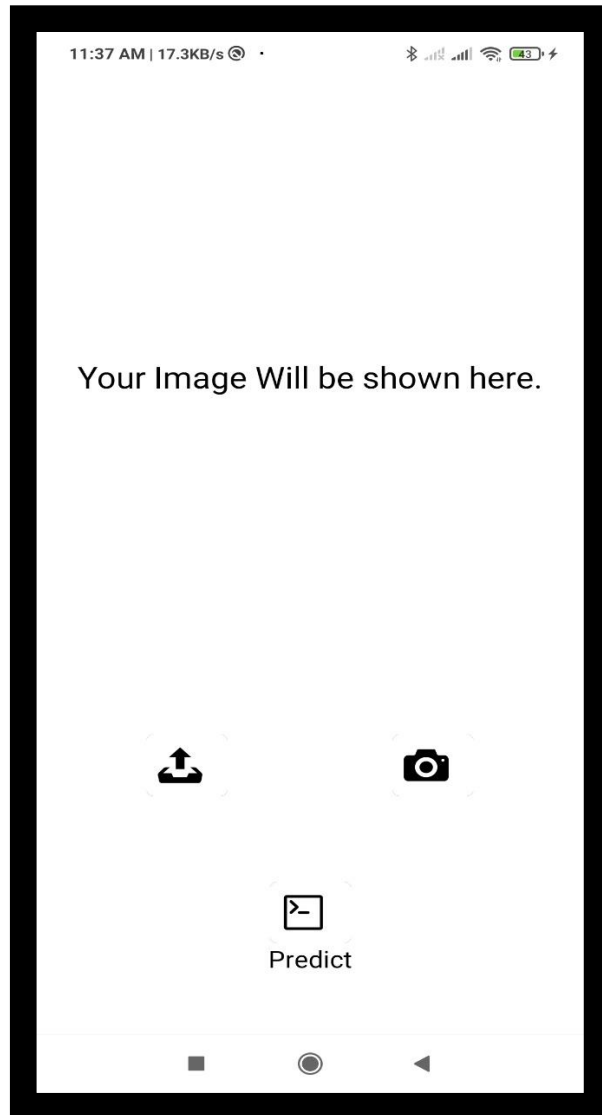


Figure 7: Initial Screen

This is the initial screen that the user will be landing on, upon landing onto the screen they will be given permission access to the camera and device storage both of which they must enable in order to proceed with the functionality.

After allowing the permissions, the user can click on the camera button to capture an image. This image will be shown to the user as shown below.

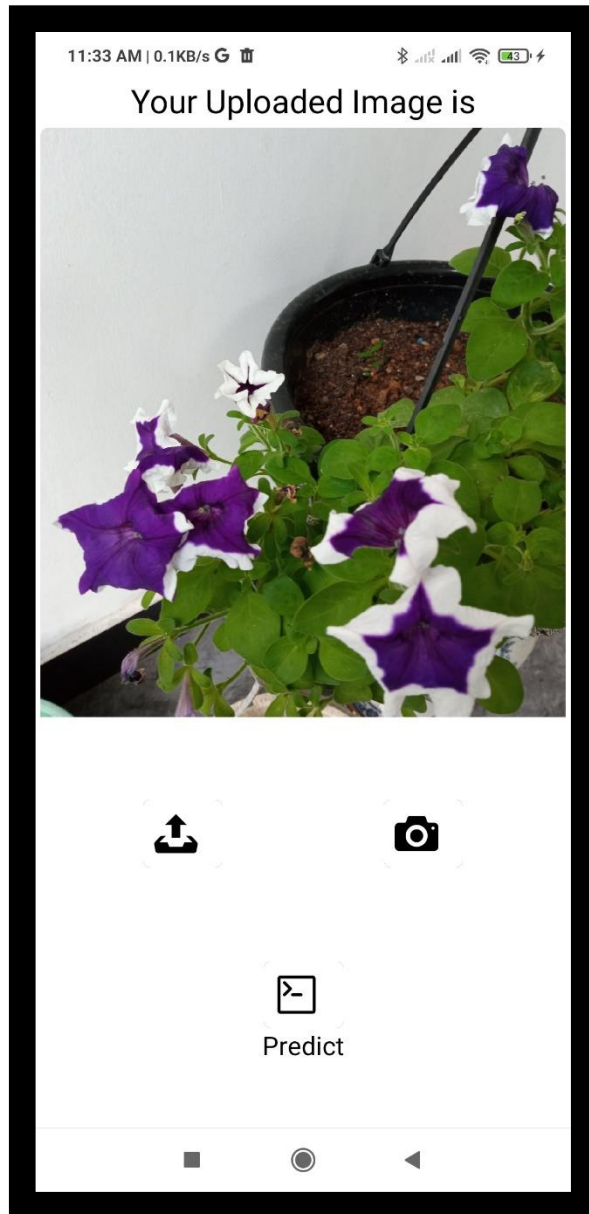


Figure 8: Camera Screen

The user can also upload images from the device storage which will be shown to the user as below.

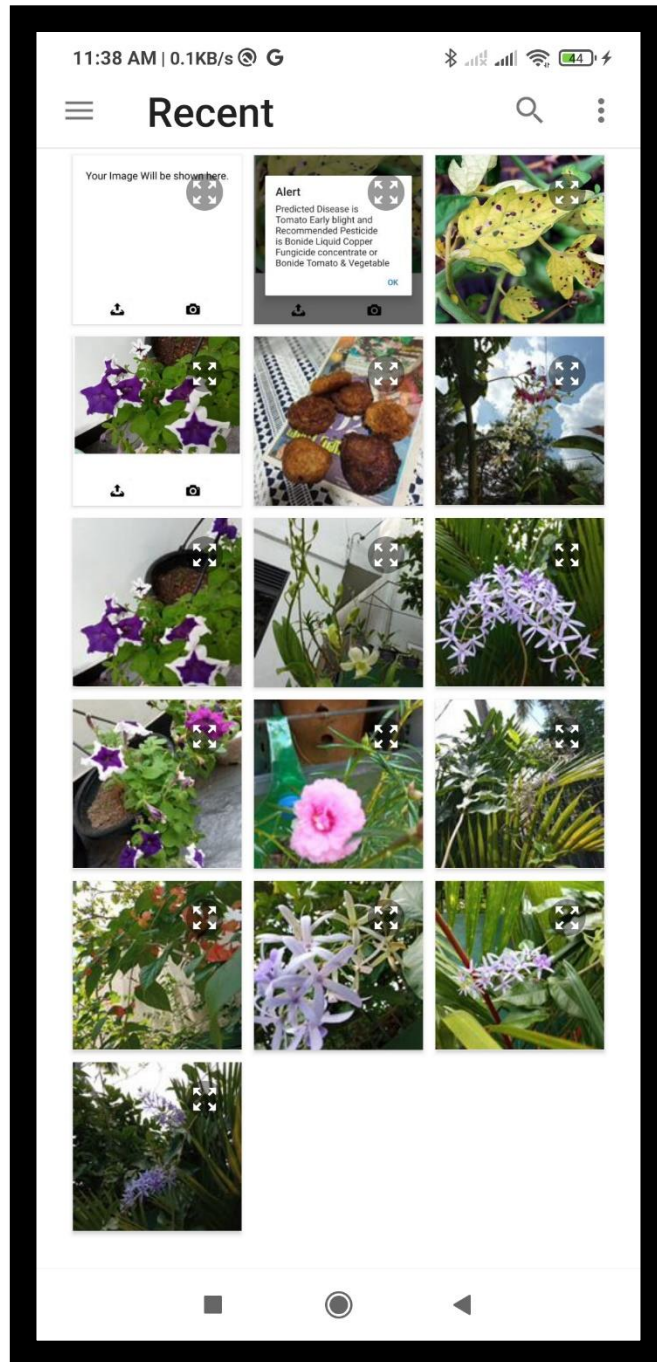


Figure 9: Upload Images from Device Screen

These are the main initial UIs involved.

After an image has been captured by the user, the user can click on the predict button to get the prediction results for the uploaded images. As shown below, once the user clicks on the prediction button, they will be show the prediction results of the image uploaded along with a predicted recommendation to overcome the situation.

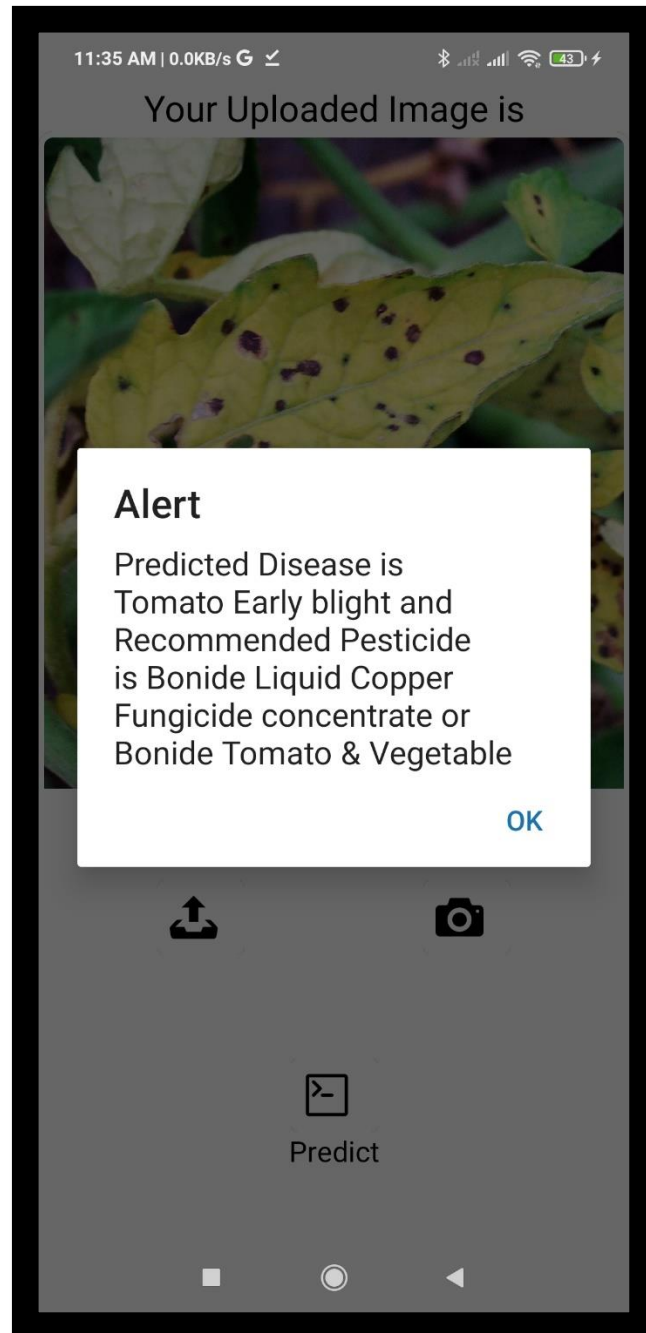


Figure 10: Prediction Screen

Similar outputs can be observed for all 38 classes which are used to train the robust image processing model.

3 RESULTS & DISCUSSIONS

3.1 Results

Since the accuracy of the model decreased when the number of classes of images increased, sometimes the model gave us inaccurate prediction results. This mainly occurred when the uploaded images were of low quality or resolution.

However, to improve the results the number of training images can be further increase. Currently the model uses 3 gigabytes of training data images to train the model. If this is being increased the accuracy of the model will further increase when using the random forest algorithm.

Accuracies of all the used algorithms for the robust image processing model are shown below,

LR: 0.916406 (0.021833)
LDA: 0.907813 (0.019702)
KNN: 0.920312 (0.012500)
CART: 0.906250 (0.027951)
RF: 0.957031 (0.013189)
NB: 0.857031 (0.010511)
SVM: 0.916406 (0.020086)

LR : Linear Regression

LDA: Linear Discriminant Analysis

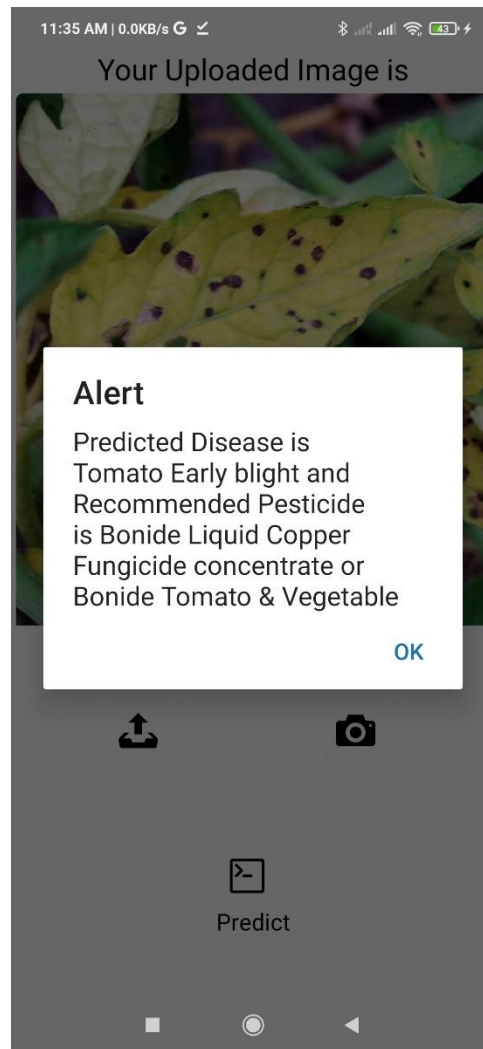
KNN: K Nearest Neighbors

CART: Classification and Regression Trees algorithm

RF: Random Forest

NB: Naïve Bayes

SVM: Support Vector Machine



Predicted image result is show along with the type of the leaf class, the type of the disease and the recommended solution that must be followed in order to overcome the situation.

3. 2 Research Findings

Many of the image processing model that were built was mainly on traditional image processing functionalities that have been provided by OpenCV. However, none of them used Hu Moments by mahotas to detect the shape of the leaf by pixel segmentation. Furthermore, haralicks is also a relatively new methodology mainly used in machine learning algorithms have been used here to detect the texture of the plant leaves being uploaded.

Furthermore, it was also analyzed when the number of training classes increased the accuracy of the model gradually decreased however this can be kept in balance by adding more images to the training model.

3.3 Discussion

The following section shows the results of the developed model, as shown in the above figure. The accuracy of the training model was upto 95% when using a with 2 classes of diseased and healthy apple leaves. When moving forward, as the number of classes were gradually increased, the accuracy of the model also increased.

Here when 150 training rounds were conducted using the random forest algorithm, the accuracy of the model increased to 70%, which is much lesser than that of the initial training model with only 2 classes.

Future Ideas

The image processing model can be made to be further robustly used even when the harvest from the crops, fruits and vegetables are taken. Here, a separate model can be integrated to identify faulty or diseased foods from the healthy ones. A large amount of data must be obtained and provided in order to achieve this with higher accuracy.

4 CONCLUSION

This research project will be focused on generating a high number of self-made business individuals and entrepreneurs in Srilanka. The application will be a key platform providing the necessary tools for those individuals to invest their time and effort to build their own agriculture business domain along with the ability to be fully employed. Furthermore, this product will not only create many self-made entrepreneurs in the agriculture business but also will help to in the economic development of the country, where a reduction of imported food can be reduced due to the production of food in the country itself.

The robust image processing model was a success however, due to the high number of classes in the model the prediction percentage was decreased. This can be overcome by using more images in the training model and increasing the number of epochs for the training model.

Moreover, in a time of a pandemic as in the current situation the country is facing. These agriculture/plantations can even aid families to fulfil their basic needs due to shortages of food supply from the government.

Hence the combination of information technology with the agriculture domain to predict a success rate figure will motivate users to be engaged in this agriprenuer business domain.

Assumptions

Location of the grown crop is irrelevant of conditions and other factors.

REFERENCES

- [1] Ft.lk. (2019). Sri Lanka heading for an agricultural issue in 2017? [Online] Available at: <http://www.ft.lk/columns/sri-lanka-heading-for-an-agricultural-issue-in-2017/4-588715> [Accessed 8 Mar. 2019].
- [2] Lasanthi N.C. De Silva, Jeevani S. Goonetillake, Gihan N. Wikramanayake, "Towards an Agriculture Information Ecosystem", The University of Colombo School of Computing, Sri Lanka, 2014.
- [3] Dhiman Mondal, Dipak Kumar Kole, Aruna Chakraborty, D. Dutta Majumder" Detection and Classification Technique of Yellow Vein Mosaic Virus Disease in Okra Leaf Images using Leaf Vein Extraction and Naïve Bayesian Classifier., 2015, International Conference on Soft Computing Techniques and Implementations- (ICSCTI) Department of ECE, FET, MRIU, Faridabad, India, Oct 8-10, 2015.
- [4] Pranjali B. Padol, Prof. AnjilA.Yadav, "SVM Classifier Based Grape Leaf Disease Detection" 2016 Conference on Advances in Signal Processing(CAPS) Cummins college of Engineering for Women, Pune. June 9-11, 2016..
- [5] "Detecting jute plant disease using image processing and machine learning", 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), 2016
- [6] Tejoindhi M.R, Nanjesh B.R, JagadeeshGujanuru Math, AshwinGeetD'sa "Plant Disease Analysis Using Histogram Matching Based On Bhattacharya's Distance Calculation" International Conference on Electrical, Electronics and Optimization Techniques(ICEEOT)-2016.
- [7] "Detection of unhealthy plant leaves using image processing and genetic algorithm with Arduino" International Conference on Power, Signals, Control and Computation, 2018.
- [8] Tanvimehera, vinaykumar,pragyagupta "Maturity and disease detection in tomato using computer vision",Fourth international conference on parallel, distributed and grid computing(PDGC) , 2016.
- [9] Ms.Poojapawar ,Dr.varshaTukar, prof.parvinpatil "Cucumber Disease detection using artificial neural network", 2018

APPENDICES

Initial validation of the model

```
import os
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers.normalization import BatchNormalization

classifier = Sequential()
classifier.add(Convolution2D(96, 11, strides = (4, 4), padding = 'valid', input_shape=(224, 224, 3), activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2), strides = (2, 2), padding = 'valid'))
classifier.add(BatchNormalization())
classifier.add(Convolution2D(256, 11, strides = (1, 1), padding='valid', activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2), strides = (2, 2), padding='valid'))
classifier.add(BatchNormalization())
classifier.add(Convolution2D(384, 3, strides = (1, 1), padding='valid', activation = 'relu'))
classifier.add(BatchNormalization())
classifier.add(Convolution2D(384, 3, strides = (1, 1), padding='valid', activation = 'relu'))
classifier.add(BatchNormalization())
classifier.add(Convolution2D(256, 3, strides=(1,1), padding='valid', activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2), strides = (2, 2), padding = 'valid'))
classifier.add(BatchNormalization())
classifier.add(Flatten())
classifier.add(Dense(units = 4096, activation = 'relu'))
classifier.add(Dropout(0.4))
classifier.add(BatchNormalization())
classifier.add(Dense(units = 4096, activation = 'relu'))
classifier.add(Dropout(0.4))
classifier.add(BatchNormalization())
classifier.add(Dense(units = 1000, activation = 'relu'))
classifier.add(Dropout(0.2))
classifier.add(BatchNormalization())
classifier.add(Dense(units = 38, activation = 'softmax'))
classifier.summary()
```

Layers through which the model is being validated

```
from keras import layers
for i, layer in enumerate(classifier.layers):
    print(i, layer.name)
```

```
.. 0 conv2d
    1 max_pooling2d
    2 batch_normalization
    3 conv2d_1
    4 max_pooling2d_1
    5 batch_normalization_1
    6 conv2d_2
    7 batch_normalization_2
    8 conv2d_3
    9 batch_normalization_3
    10 conv2d_4
    11 max_pooling2d_2
    12 batch_normalization_4
    13 flatten
    14 dense
    15 dropout
    16 batch_normalization_5
    17 dense_1
    18 dropout_1
    19 batch_normalization_6
    20 dense_2
    21 dropout_2
    22 batch_normalization_7
    23 dense_3
```

Data Preprocessing with all classes

```
from keras import optimizers
classifier.compile(optimizer=optimizers.SGD(lr=0.001, momentum=0.9, decay=0.005),
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   fill_mode='nearest')
valid_datagen = ImageDataGenerator(rescale=1./255)
batch_size = 128
base_dir = r"New Plant Diseases Dataset(Augmented)"
training_set = train_datagen.flow_from_directory(base_dir+'/train',target_size=(224, 224),
                                                batch_size=batch_size,
                                                class_mode='categorical',
                                                shuffle=False)
valid_set = valid_datagen.flow_from_directory(base_dir+'/valid',
                                              target_size=(224, 224),
                                              batch_size=batch_size,
                                              class_mode='categorical',
                                              shuffle=False)
```

```
Found 70295 images belonging to 38 classes.
Found 17572 images belonging to 38 classes.
```

Viewing all the classes

```
li = list(class_dict.keys())
print(li)
```

Python

```
['Apple__Apple_scab', 'Apple__Black_rot', 'Apple__Cedar_apple_rust', 'Apple__healthy', 'Blueberry__healthy', 'Cherry_(including_sour)__Powdery_mildew',
'Cherry_(including_sour)__healthy', 'Corn_(maize)__Cercospora_leaf_spot Gray_leaf_spot', 'Corn_(maize)__Common_rust_',
'Corn_(maize)__Northern_Leaf_Blight', 'Corn_(maize)__healthy', 'Grape__Black_rot', 'Grape__Esca_(Black_Measles)',
'Grape__Leaf_blight_(Isariopsis_Leaf_Spot)', 'Grape__healthy', 'Orange__Haunglongbing_(Citrus_greening)', 'Peach__Bacterial_spot', 'Peach__healthy',
'Pepper,_bell__Bacterial_spot', 'Pepper,_bell__healthy', 'Potato__Early_blight', 'Potato__Late_blight', 'Potato__healthy', 'Raspberry__healthy',
'Soybean__healthy', 'Squash__Powdery_mildew', 'Strawberry__Leaf_scorch', 'Strawberry__healthy', 'Tomato__Bacterial_spot', 'Tomato__Early_blight',
'Tomato__Late_blight', 'Tomato__Leaf_Mold', 'Tomato__Septoria_leaf_spot', 'Tomato__Spider_mites Two-spotted_spider_mite', 'Tomato__Target_Spot',
'Tomato__Tomato_Yellow_Leaf_Curl_Virus', 'Tomato__Tomato_mosaic_virus', 'Tomato__healthy']
```

Mongo DB collection

```
_id: ObjectId("6166b27725387fc16e722ab4")
db_image_name: "/9j/2wCEAAEBAQEBAQIBAQIDAgICAwQDAwMDBAUEBAQEBAUGBQUFBQUFBgYGBgYGBgYHBw..."
image_class: "{"pesticide": "Bonide Liquid Copper Fungicide concentrate or Bonide Tom..."
```

```
_id: ObjectId("6166b2bee53d5aca60a74bbe")
db_image_name: "/9j/4AAQSkZJRgABAQAAQABAAD/4AAQSkZJRgABAQAAQABAAD/2wBDAAgGBgcGBQgHBw..."
image_class: "{"pesticide": "Myclobutanil (Immunox)", "classes": "Apple Cedar apple rus..."
```