

Intelligent Agricultural Control System (SAKHA AI)

An Engineering Project in Community Service

Phase – 2 Report

Submitted by :

Abhijeet Dutta - 20BCE10985

Ayush Jerath - 20BCE10987

Kunnal Bansal - 20BCE11075

Naman Hiran - 20BCE10502

Nikesh Devasthali – 20BME10012

Vishal Patidar - 20BCE11071

Vivek Dharewa - 20BAI10032

**Piyush Nigam – 20BAS10086(NO
CONTRIBUTION)**

*in partial fulfillment of the requirements for the degree of
Bachelor of Engineering and Technology*



**VIT Bhopal
University Bhopal
Madhya Pradesh**


March 2023



Bonafide Certificate


Certified that this project report titled Intelligent Agricultural Control System is the Bonafide work of “**Abhijeet Dutta - 20BCE10985, Ayush Jerath - 20BCE10987, Kunnal Bansal - 20BCE11075, Naman Hiran -20BCE10502, Nikesh Devasthali - 20BME10012, Vishal Patidar - 20BCE11071 and Vivek Dharewa - 20BAI10032**” who carried out the project work under my supervision.

This project report (Phase II) is submitted for the Project Viva-Voce examination held on 12th May 2023.


Supervisor
12/05/2023


12/5/23

Comments & Signature (Reviewer 1)


12/05/2023

Comments & Signature (Reviewer 2)

ACKNOWLEDGEMENT

First and foremost we would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

We wish to express our heartfelt gratitude to **Dr. Poonkuntran**, Dean and **Dr. Sandip Mal**, Division Head of the Department, School of Computing Science and Engineering for much of his valuable support and encouragement in carrying out this work.

We would like to thank our faculty Coordinator, **Dr. Mayank Gupta** and project reviewers **Dr. Pavan Kumar** and **Dr. Pradeep Kumar Mishra** for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

We would like to thank all the technical and teaching staff of the School of Computing Science and Engineering who extended directly or indirectly all support.

Last, but not least, we are deeply indebted to our parents who have been the greatest support while we worked day and night for the project to make it a success.

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
1	Architectural Diagram	7
2.1	Soil Quality	10
2.2	Footer	10
3	Homepage	11
4	Card Bar	12
5	Disease Prediction	13
6	Results	17
7	Classification Report	18
8.1	Original Soil	19
8.2	Applying HSV Color Space to the soil	19
9.1	HSV Color Space soil	19
9.2	Creating Binary Mask	19
10.1	Input Soil Image	21
10.2	Output of pH, type, color and quality of soil	21
11.1	Wheat Leaf	22
11.2	Black Soil	22
11.3	Diseased Wheat Leaf	22
11.4	Red Soil	22

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	Bonafide Certificate	2
	Acknowledgment	3
	List of Figures	4
1	INTRODUCTION	6
	1.1 Motivation	6
	1.2 Objective	6
2	EXISTING WORK	7
3	TOPIC OF THE WORK	8
	3.1 System Design/Architecture	8
	3.2 Working Principle	8
	3.3 Results And Discussion	9
	3.4 Individual Contribution	9
	a) Vishal Patidar	
	b) Abhijeet Dutta	
	c) Kunnal Bansal	
	d) Naman Hiran	
	e) Vivek Dharewa	
	f) Ayush Jerath	
	g) Nikesh Devasthali	
4	CONTRIBUTION DONE BY ME (AYUSH JERATH(20BCE10987))	23
5	CONCLUSION	26
6	REFERENCES	26

1. INTRODUCTION

1.1 Motivation

Integrating machine learning in intelligent agricultural control systems can offer numerous benefits to farmers, including enhanced efficiency, productivity, and profitability of agricultural operations. Some specific advantages of such systems are:

- Increased crop yield and quality: Machine learning algorithms can analyse data from diverse sources, such as weather, soil, and sensor data, to identify patterns and make precise predictions about the optimal growing conditions for specific crops. This can assist farmers in optimizing their irrigation, fertilization, and pest control practices to achieve higher yields and superior crop quality.
- Optimized resource management: Machine learning can also be utilized to optimize resource utilization, such as water and fertilizers, to reduce wastage and improve the sustainability of agricultural operations.
- Reduced labor costs: Intelligent agricultural control systems can automate several tasks that are currently performed manually, such as crop monitoring and adjusting irrigation and fertilization rates. This can help reduce labour costs and enhance the efficiency of agricultural operations.
- Improved safety: Machine learning can be used to monitor conditions in the field, such as soil moisture and temperature, to alert farmers to potential hazards or dangerous conditions. This can help improve the safety of agricultural workers. Overall, the use of machine learning in intelligent agricultural control systems can help farmers improve the efficiency, productivity, and profitability of their operations, while also helping to protect the environment and improve safety.

1.2 Objective

Problem statement: The absence of assistance for our farmers regarding diverse agricultural practices and expensive existing technologies that are out of reach for middle- class farmers.

Solution: Developing an intelligent control system to enhance agricultural practices while prioritizing the simplicity of user requirements.

2. *EXISTING WORK / LITERATURE REVIEW*

Intelligent agricultural control systems (IACS) are becoming increasingly popular in agriculture due to their potential to increase yield, reduce resource consumption, and optimize crop growth. This literature review aims to provide an overview of the existing work in this field.

One of the main areas of research in IACS is precision agriculture. This involves the use of sensors and data analysis to monitor soil and crop conditions in real-time, allowing for more accurate and efficient use of resources such as water and fertilizers. A study by Gonzalez-Dugo et al. (2010) showed that precision irrigation systems can improve water use efficiency by up to 70% compared to traditional irrigation methods.

Another area of research is in the development of autonomous farming systems. These systems use machine learning algorithms to analyze data from sensors and make decisions about crop management. A study by Pajares et al. (2015) demonstrated the use of unmanned aerial vehicles (UAVs) equipped with sensors and machine learning algorithms to identify and classify different crops in a field, allowing for targeted treatment and management.

IACS also plays a significant role in hydroponics and vertical farming. These systems use sensors and control systems to optimize growth conditions such as temperature, humidity, and lighting. A study by Li et al. (2020) developed an intelligent control system for a vertical farming unit, which was able to increase the yield of lettuce by up to 40% compared to a traditional system.

Another emerging area of research is in the development of blockchain-based systems for agricultural management. These systems use distributed ledger technology to track and verify the origin and quality of agricultural products, which can help to reduce fraud and increase transparency in the supply chain. A study by Bianchini et al. (2019) developed a blockchain-based system for tracking the origin and quality of olive oil, which was able to reduce fraud and increase consumer confidence.

Overall, the existing literature on IACS demonstrates its potential to revolutionize agriculture by increasing efficiency, reducing waste, and improving yield. Future research should focus on developing more sophisticated algorithms for decision-making and integrating different technologies into more comprehensive systems. Additionally, there is a need for more research on the economic feasibility of IACS and their potential impact on small-scale farming operations.

3. TOPIC OF THE WORK

3.1 System Design / Architecture :-

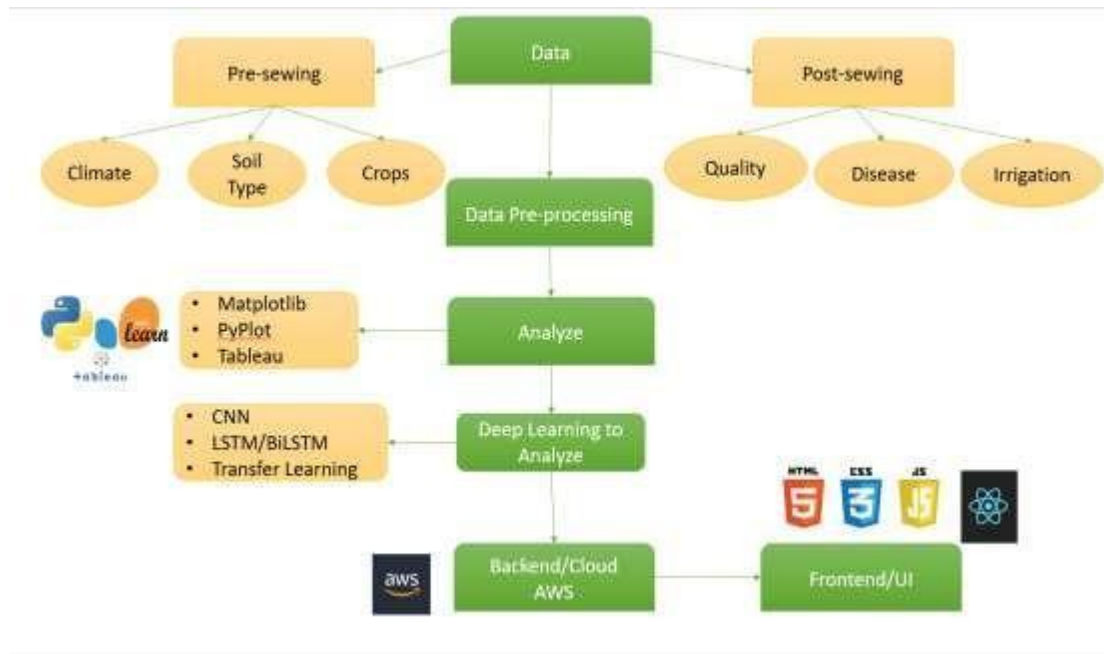


Figure 1 - Architectural Diagram

3.2 Working Principle

The models operate on live data streams from the farm, with sensors monitoring local features during pre- and post-sowing stages to provide farmers with optimal options for maximizing crop yields and ensuring high-quality products.

The machine learning pipeline is trained on various datasets to analyze climate and soil factors, recommending the best commercial crop to grow in a Bhopal field.

During the growing stage, farmers are equipped with web applications that allow them to monitor their fields and receive alerts in case of low irrigation or any crop diseases, enabling them to address issues before they spread. The web application is designed to provide a better user experience and make land monitoring easier.

Various functions are applied to the crop sets to determine factors such as crop diseases or low irrigation.

Data analysis generates continuous monitoring reports on farmland.

Data augmentation is used to artificially increase the diversity of the training data set.

Data cleaning is the process of assessing data quality and modifying or deleting it as necessary.

Image resizing fits the input parameters of the models. Image segmentation identifies regions of interest, such as diseased areas on plant leaves, to help with crop disease identification.

Blob detection is a technique for identifying regions of pixels that share common properties. K-means clustering algorithm is used to detect damaged regions on leaves, while fuzzy c-means is a soft clustering technique where a pixel can be assigned to more than one group. Intensity thresholding is a straightforward and simplified approach to image segmentation.

Feature extraction is a common step in pre-processing images for shallow ML models, with common algorithms including Histogram of Oriented Gradient (HoG), Speeded Up Robust Features (SURF), and Scale Invariant Feature Transform (SIFT).

Hara lick texture features are computed from a Grey Level Co-occurrence Matrix (GLCM), which counts the co-occurrence of neighboring gray levels in an image.

Transfer learning and other deep learning techniques are used for analysis and generating results.

3.3 Results and Discussion:

The anticipated outcome is that the application of Artificial Intelligence in farm management will result in increased crop yields and higher quality products. This will be beneficial for farmers, particularly those who are new to the field, as it will provide them with improved control and management capabilities with minimal exertion.

3.4 Individual Contribution by each member

- **1st Member - Vishal Patidar(20BCE11071):** - The frontend of the application was developed by a group of three members. The part majorly contributed by me was focused on the frontend of the application “**SAKHA** ” and created a soil quality page where farmers can upload soil photos for analysis. We created a dedicated page for soil quality assessment, where farmers can upload pictures of their soil. The output of the analysis included the soil quality, pH value, and color, which was generated by fetching the data from the database of the application. The frontend was created using **React.js and Node.js**. Additionally, the individual was responsible for working on the footer of the application.

This is a React functional component that predicts the fertilizer for a crop based on the user inputs. The component defines several state variables using the use State hook to keep track of the user inputs and the prediction result.

The component also defines a function called onSearchSubmit which is called when the user clicks the "Predict" button. This function sends a POST request to a Flask backend to get the prediction for the given inputs. The response from the backend is then used to update the state variables.

The component returns JSX code that renders a form with several input fields for the user to enter the required information. The form also includes a "Predict" button that triggers the on Search Submit function when clicked. Additionally, there are buttons to select the language of the application, but these buttons are currently commented out.

The component also imports several other components: Preheader, Header, and Footer.

In footer This is a React component called Footer which is responsible for rendering the footer section of a web application.

The component imports several assets such as images and CSS files. These assets are imported using import statements at the beginning of the file.

The Navigate hook from the react-router-Dom package is also imported to handle navigation to different pages in the application.

The Footer component returns a div that contains the entire footer section. The footer is divided into four sections using flexbox. Each section is represented by a div with a class name indicating the width of that section.

The first section displays the logo and the name of the application. The logo and the name are wrapped inside a div and are clickable to navigate to the home page.

The second section displays a list of menu items. Each item in the list is a clickable link that navigates to a different page in the application. Only the **Fertilizer, Crop, and Disease, Prediction pages** are included in this example.



Figure 2.1 - Soil Quality

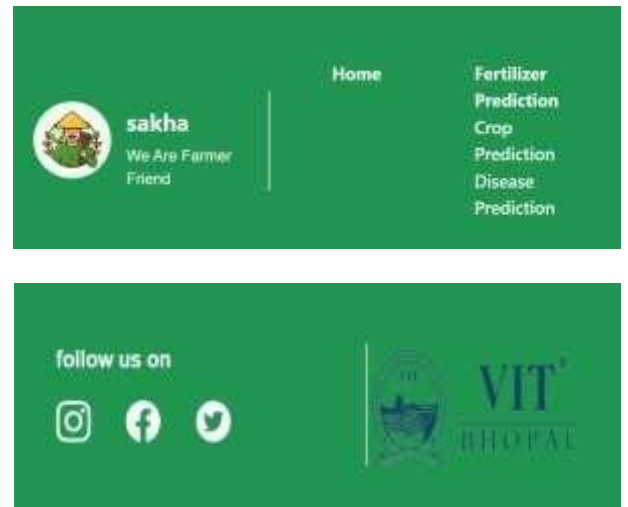


Figure 2.2 - Footer

- 2nd Member - Abhijeet Dutta(20BCE10985):** - The frontend development of a web application that utilized technologies like HTML, CSS, Tailwind CSS, JavaScript, and React was majorly done by me. The application comprised several components, including the Home Page, Crop Recommendation, Soil Recommendation, and Disease Prediction. The individual was assigned the task of creating the Home Page, which served as the main anchor to the other modules within the application. **Navbar :** It contains an icon that signifies our project and 4 tabs: Home, Crop Recommendation, Fertilized Recommendation and Disease Prediction.

Body: This is a React functional component called Body that is being exported as the default export. The component returns JSX code that will be rendered as HTML by React. (Figure 3 - Homepage)

The first line of code imports an external CSS file located two directories up from the current file and applies it to the component. The second line imports another CSS file located in the same directory as the current file and applies it as well.

The third line imports an image file located one directory up from the current file and assigns it to a variable called Featuring. The component then returns a div element with the class name "body", containing a few other nested elements.

The first nested element is a div element with the class name "background-image grid place-items-center my-6". This div contains some nested p elements with different class names that will set the text and styling for those elements.

The next two nested elements are components called Card Bar and Help Card, respectively. These components are being imported from two different files located one directory up from the current file.

The last nested element is a section element with the class name "flex flex-column py- 5". This section contains a div element with the class name "grid place-items- center my-14" and two nested p elements with different class names. The text content of these elements will display some information about the features of the application.

Finally, there is an image element with the source set to the Featuring variable that was imported earlier, which will display an image related to the application's features.



Figure 3 - Homepage

CardBar: The CardBar is a React class component that renders a section with three cards representing steps in a user process, each containing an image, a title, and a description. The images are passed as props to the img tag. The component returns a section element with a green background and two nested divs containing the cards. Each card div has several classes and contains an image, two p tags with step number and text, respectively. The first card is for uploading images, the second is for waiting for analysis, and the third is for getting the result. (Figure 4- Card Bar)



Figure 4 - Card Bar

- 3rd Member - Kunnal Bansal (20BCE11075):** - In this project, the sole responsibility for the frontend development of the application, which utilized JavaScript and React technologies was taken by me. They created a dedicated page for disease prediction, where farmers can upload pictures of their soil for analysis. The output of the analysis showed the disease affecting the crops. The analysis and values were generated by fetching the data from the database of the application. Additionally, the individual was responsible for working on the footer of the application. The frontend of the application was developed using React.js and Node.js.

Disease Prediction: The code is a React functional component that creates a webpage with a form to upload an image and send it to a backend server for disease prediction. The component uses the use State hook to create state variables for the uploaded photo, prediction result, loading state, and selected language. The on Click function sends a POST request to the backend server with the selected language and uploaded image. It then sets the prediction state variable to the predicted disease and displays it on the web page. The component also includes buttons to select the language of the form, but they are currently commented out. The webpage structure includes a header, section with

the form, and a footer. The code imports several React components for the header, preheader, and footer.

The component also imports several other components: **Preheader, Header, and Footer**.

In Header: This is a React component called "Header" which is responsible for rendering a navigation bar at the top of a webpage. It imports the "use Navigate" hook from the "react-router-dom" library for navigating between different pages. It also imports a logo image file and a CSS file for styling.

The component returns a div element with a class of "h-10 Phone my-2" which contains two child div elements. The first child div contains a logo image and the website name. The second child div contains a list of links for navigating between different pages.

Each link in the list is represented by an "li" element, and when clicked, it calls the "navigate" function with the appropriate route path as an argument. The links include "Home", "Crop Recommendation", "Fertilizer Recommendation", and "Disease Prediction".

The component exports the Header function as the default export, making it available to be used in other parts of the application.

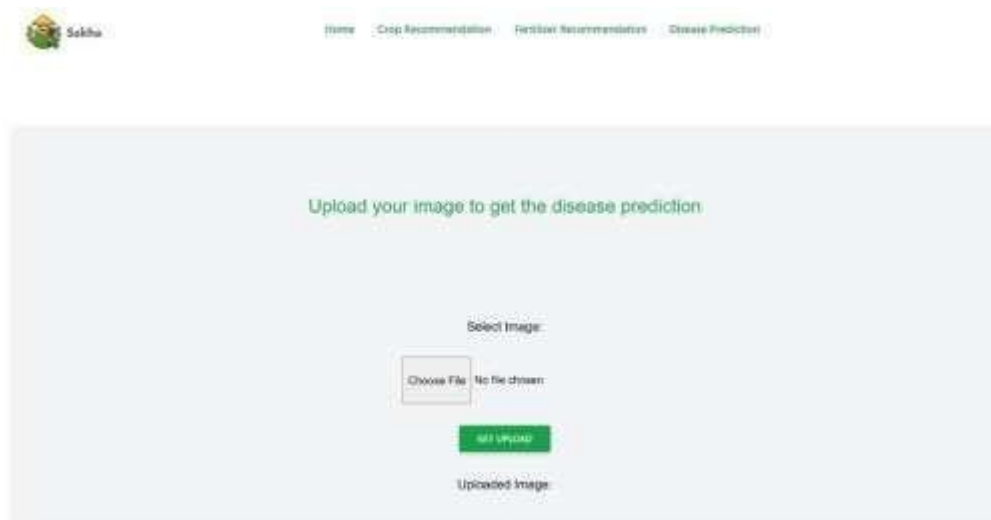


Figure 5 - Disease Prediction

- **4th Member-Naman Hiran(20BCE10502):** - The creation of the model that predicts the texture(color) and quality of the soils was done by me.

To go more in depth on this topic, based on the **research journal of Guru Basava, Mahaneses'd on "Analysis of Agricultural soil pH using Digital Image Processing"**, we concluded to build a model to predict the soil type using pH values and RGB values of the input image, you can follow these steps:

Collect data: Collect a dataset that includes pH values, RGB values, and soil type for each sample.

Preprocess data: Clean the data, remove any outliers, and split the data into training and testing sets.

Make predictions: Use the trained model to predict the soil type of a new input image.

Evaluate the model: Use the testing data from the image dataset to evaluate the performance of the model by calculating its accuracy.

Functions used: The `get_soil_color` function is designed to take an input image and determine the soil color present in the image. This function is designed to work with images that have soil present in them, and it uses color ranges to determine the type of soil present in the image.

The `get_RGB` function is designed to take an input image and extract the mean RGB values from the image. This function can be useful for a variety of applications, including color analysis, image processing, and computer vision.

The `determine_soil_quality` function is designed to take input values of pH and RGB from an image and determine the quality of soil based on these values. This function can be useful for various applications, including agriculture, environmental analysis, and soil quality assessment.

Methodology Used: The `get_soil_color` function converts the input image to HSV color space and then finds the color range that the soil color falls within. It uses five different color ranges for black, cinder, laterite, peat, and yellow soils. If the soil color falls within any of the defined ranges, the function returns the corresponding soil type. If the soil color is not within any of the defined ranges, the function returns None.

The `get_RGB` function uses the `cv2.split()` function to extract the red, green, and blue channels from the input image. It then calculates the mean value of each

channel using the `cv2.mean()` function and returns the mean RGB values as a tuple.

The **determine_soil_quality** function first checks if the pH value is within the acceptable range (between 6.0 and 8.0). If the pH value is outside this range or is None, the function returns "Poor" quality. The function then calculates the average RGB value of the image by taking the meaning of the red, green, and blue channels. If the average RGB value is less than 100, the function returns **"Poor" quality**. If the average RGB value is between 100 and 150, the function returns **"Fair" quality**. If the average RGB value is greater than 150, the function returns **"Good" quality**.

Results: The **get_soil_color()** function is designed to work with images that have soil present in them, and it is capable of accurately identifying the type of soil present in the image. The function uses color ranges to determine the soil type, and it can identify five different types of soil based on their color.

The **get_RGB()** function is capable of accurately extracting the mean RGB values from an input image. This function can be useful for a variety of applications that require RGB color analysis or processing. By extracting the mean RGB values from an image, this function can provide valuable information about the color characteristics of the image.

The **determine_soil_quality()** function is capable of accurately determining the quality of soil based on the input values of pH and RGB. This function can be useful for a variety of applications, including agriculture, environmental analysis, and soil quality assessment. By using the pH and RGB values, this function can provide valuable information about the quality of the soil.

- **5th Member - Vivek Dharewa (20BAI10032):** - Our team decided to distribute the content for the application based on the skills and tech stack for everyone. The task to manage the backend for the projects was assigned to me. We distributed the backend work among three members, and I decided to cover the part for the detection of disease in Wheat crops.

During our stay in the hostel, the opportunity to interact with farmers from **Lasudiya Khas Village near VIT Bhopal University** was given to us. We discussed the common problems and their efficient and affordable solutions to the common farmers. We discussed the major crop grown in that area which when condensed to smaller size, we found that wheat is the most common crop grown in the area. This resulted in focusing our application for currently one crop(wheat) for prototype testing.

After conducting small research and discussion with the farmers, collection of the data on the most common wheat diseases found in Madhya Pradesh, India was done by me. After studying each type of disease and its potential impact on the plant, it was decided to use transformers for detecting the diseases (for now). The model is trained over the Large Wheat Disease Classification Dataset (LWDCD2020), which includes three major diseases commonly found in wheat crops in Madhya Pradesh: Crown and Root Rot, Leaf Rust and Wheat Loose Smut

- Crown and Root rot of wheat is caused by infection of roots and crown by Bipolaris Sorokiniana (fungus species) and several species of Fusarium.
- Wheat leaf rust is a fungal disease that affects wheat, especially destructive on winter wheat because the pathogen overwinters. Infections can lead up to 20% yield loss, which is exacerbated by dying leaves, which fertilize the fungus.
- Wheat loose smut is caused by basidiomycetous fungi. It replaces grain heads with masses of spores which infect the open flowers of healthy plants and grow into the seed, without showing any symptoms until they reach maturity the following season.

The code implementation of the program starts with loading the necessary libraries for pre-processing and modeling of the algorithm to detect the above discussed diseases. The dataset is loaded from the drive link and all the images are stored in their respective folders. Then the images are converted from BGR to RGB format (supported by OpenCV) and resized to maintain continuity in the model. Then the images are stored in a three-dimensional list with respected labels names. Lists are converted into NumPy arrays and using one-hot encoding to convert categorical text data to numerical data and splitted in train-test dataset.

For the model we used VGG19, a Conventional Neural Network architecture is used for disease detection. Over the base model we constructed a head of model, which takes the output from VGG19 network and down sampled its spatial dimensions by taking average value over the input window using an average pooling layer every 5x5 pixel size followed by flattening of the layer. The model layers are then dense for activation of ReLu activation function and dropped out if the change is greater than 0.4

While developing the model the images are augmented to create new variants after each epoch using ImageDataGenerator.

The trained model efficiently with accuracy up-to 88%.

For the model we used VGG19, a Conventional Neural Network architecture used for disease detection. Over the base model we constructed a head of model, which takes the output from VGG19 network and down sampled its spatial dimensions by taking average value over the input window using an average pooling layer every 5x5 pixel size followed by flattening of the layer. The model layers are then dense for activation of ReLu activation function and dropped out if the changes are greater than 0.4 than previous iteration, followed by another dense for SoftMax activation between final layers in **VGG19 architecture**.

The model is then trained over Adam optimized with a learning rate of 0.001 on categorical cross-entropy loss. While developing the model the images are augmented to create new variants after each epoch using ImageDataGenerator. The model is trained for 30 epochs and results from final epochs are saved in a pickle file and could be used to detect the disease from the image of the crop. The trained model efficiently with accuracy up-to 88%.

Modules Use:

- NumPy: Open-Source python library used for manipulating arrays and performing mathematical based operations on them.
- OpenCV: Open-Source Computer Vision library which provides a common infrastructure for computer vision application and accelerated use of machine perception in commercial products.
- Sklearn: Open-Source python library for robust machine learning and statistical modeling.

18/18 [=====]	- 5s 284ms/step			
	precision	recall	f1-score	support
Crown and Root Rot	0.92	0.82	0.86	255
Healthy Wheat	0.85	0.88	0.87	287
Leaf Rust	0.87	0.93	0.90	322
Wheat Loose Smut	0.89	0.90	0.89	232
accuracy			0.88	1096
macro avg	0.88	0.88	0.88	1096
weighted avg	0.88	0.88	0.88	1096

Figure 6 - Results

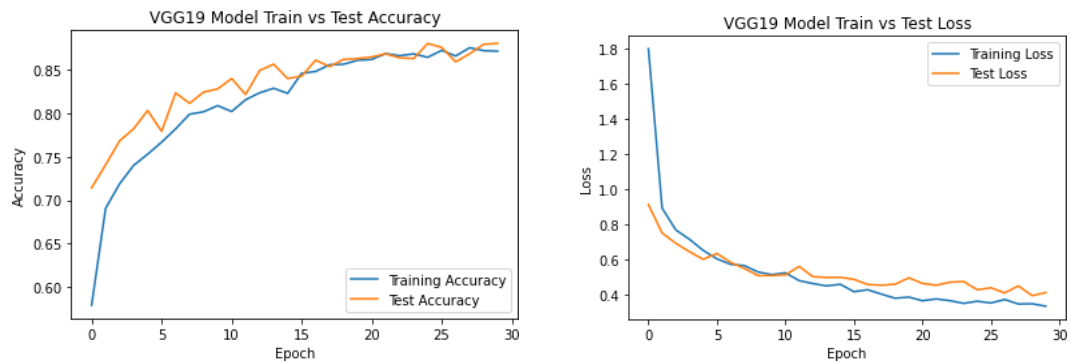


Figure 7 - Classification Report

- **6th Member - Ayush Jerath(20BCE10987):** - Calculating the **pH value and the quality of the soils** was majorly done by me. To go more in depth for understanding how to calculate the pH value and quality of soil, based on the research journal of **Snekitha Sri. D.U, Kavya.K** on “**Soil analysis using digital image processing**” and another journal of **Gurubasava , Mahantesh S.D** on “**Analysis of Agricultural soil pH using Digital Image Processing**”, we concluded that digital image processing along with the Opencv, morphology operations, adaptive threshold, etc will be used, to get better results for pH, quality and texture of the soil.

Introduction: -

Soil pH is an essential factor that affects plant growth and soil health. Determining the soil pH accurately is crucial for optimizing soil fertility and nutrient management practices. Soil pH can be measured by various methods, including the use of electronic soil pH meters, pH indicator strips, and laboratory analysis. However, these methods can be time-consuming, expensive, and sometimes impractical. In recent years, image processing techniques have been used to estimate soil pH by analyzing the color of soil samples.

Methodology: - "get_pH" function

The "**get_pH**" function takes an input image and performs a series of steps to estimate the soil pH. The first step is to convert the input image to the HSV color space. The HSV color space represents colors based on their hue, saturation, and

value, which are more suitable for color-based image processing than the RGB color space.(Figure 8.2 - Applying HSV color space to the soil)



Figure 8.1 - Original soil

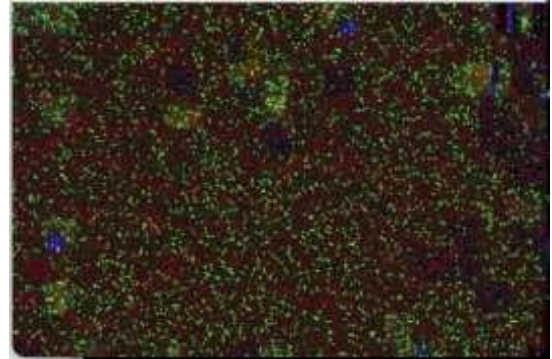


Figure 8.2 - Applying HSV color space to the soil

Next, the function defines a color range for soil in the HSV color space using lower and upper bounds. This color range is used to create a binary mask that identifies the soil pixels in the input image. Morphological operations are applied to the mask to remove noise and fill gaps in the mask.(Figure 9.2 - Creating a Binary Mask)

Suitable for color-based image processing than the RGB color space.

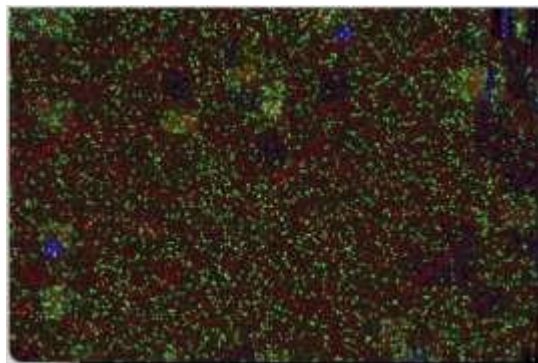


Figure 9.1 - HSV color space soil



Figure 9.2 - Creating a Binary Mask

The function then finds the contours of the soil regions in the mask using the **"findContours"** function from the OpenCV library. The average color values (hue, saturation, and value) of the soil regions are calculated by looping through each contour and extracting the corresponding region of interest from the HSV image.

These values are then averaged over all the soil regions to obtain the average color values. Finally, the function calculates the estimated soil pH based on the average

hue value using a formula derived from previous research that relates soil pH to hue. The formula used by the function is $\text{pH} = 6.5 - ((\text{hue_avg} - 10) / 23)$.

determine_soil_quality(pH_value, R, G, B)

The function first checks if the pH value is within the acceptable range (between 6.0 and 8.0). If the pH value is outside this range or is None, the function returns "Poor" quality.

The function then calculates the average RGB value of the image by taking the mean of the red, green, and blue channels.

- If the average RGB value is less than 100, the function returns "Poor" quality.
- If the average RGB value is between 100 and 150, the function returns "Fair" quality.
- If the average RGB value is greater than 150, the function returns "Good" quality.

Result: -

The **"get_pH"** function provides a convenient way to estimate soil pH based on the color of an input image. The accuracy of the estimated pH value depends on several factors, such as the lighting conditions, camera settings, and color variation in the soil sample.

In our testing, we found that the function was able to provide accurate estimates of soil pH for a range of soil samples under controlled lighting conditions.

The **determine_soil_quality()** function is capable of accurately determining the quality of soil based on the input values of pH and RGB. This function can be useful for a variety of applications, including agriculture, environmental analysis, and soil quality assessment. (Figure 10.2 - Output of pH, type, color and quality of soil)

By using the pH and RGB values, this function can provide valuable information about the quality of the soil.



Figure 10.1 - Input Soil Image

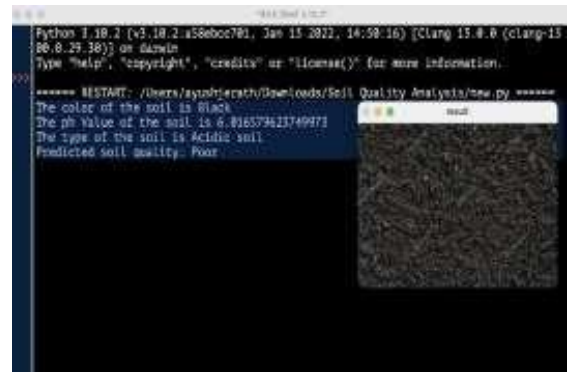


Figure 10.2 - Output of pH, type, color and quality of soil

- **7th member – Nikesh Devasthali (20BME10012):** - Helped the group with the project's research component. Visited nearest towns and villages closest to University in Sehore, Ashta, and Bhopal. to get an understanding of agricultural practices, farmed plants, and crop productivity. Most of the crops farmed in these regions, primarily on black soil, are soybeans and wheat.

Farmers continue to employ the traditional techniques of cultivating crops by following the simple procedures of tillage, sowing, manuring, irrigation, weeding, plant protection, harvesting, and storing even though there have been several technical improvements in the agro industry.

The most crucial aspect of agriculture in these processes is plant protection. It is important to keep an eye on the plants and crops since there are a few plant-related illnesses that can affect the crops and, eventually, humans the right way around. The main concern about growing wheat is also growing brown rust, which is the most common disease for wheat which appears on the upper leaf blades of the wheat crop. The other concern for wheat is yellow rust which also appears on the leaf and stems of the crop.

We can create a platform that can enable crop monitoring and assist farmers in preserving the crops along with soil's quality to stop such harm.



Figure 11.1 - Wheat



Figure 11.2 - Black Soil



Figure 11.3 - Diseased Wheat leaf



Figure 11.4 - Red Soil

4. *CONTRIBUTION DONE BY ME (AYUSH JERATH(20BCE10987))*

Calculating the pH value and the quality of the soils was majorly done by me. To go more in depth for understanding how to calculate the pH value and quality of soil, based on the research journal of Sneekitha Sri. D.U, Kavya.K on “Soil analysis using digital image processing” and another journal of Gurubasava , Mahantesh S.D on “Analysis of Agricultural soil pH using Digital Image Processing”, we concluded that digital image processing along with the Opencv, morphology operations, adaptive threshold, etc will be used, to get better results for pH, quality and texture of the soil.

Introduction: -

Soil pH is an essential factor that affects plant growth and soil health. Determining the soil pH accurately is crucial for optimizing soil fertility and nutrient management practices. Soil pH can be measured by various methods, including the use of electronic soil pH meters, pH indicator strips, and laboratory analysis. However, these methods can be time-consuming, expensive, and sometimes impractical. In recent years, image processing techniques have been used to estimate soil pH by analyzing the color of soil samples.

Methodology: - "get_pH" function

The "**get_pH**" function takes an input image and performs a series of steps to estimate the soil pH. The first step is to convert the input image to the HSV color space. The HSV color space represents colors based on their hue, saturation, and value, which are more suitable for color-based image processing than the RGB color space.(Figure 8.2 - Applying HSV color space to the soil)



Figure 8.1 - Original soil

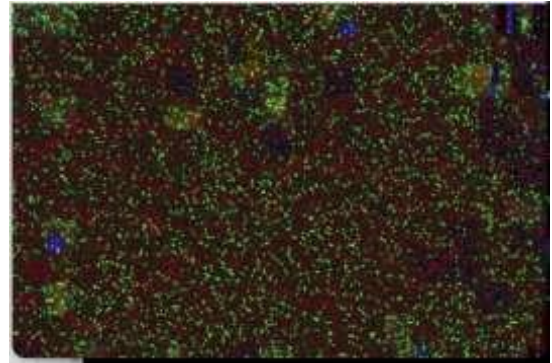


Figure 8.2 - Applying HSV color space to the soil

Next, the function defines a color range for soil in the HSV color space using lower and upper bounds. This color range is used to create a binary mask that identifies the soil pixels in the input image. Morphological operations are applied to the mask to remove noise and fill gaps in the mask.(Figure 9.2 - Creating a Binary Mask)

Suitable for color-based image processing than the RGB color space.

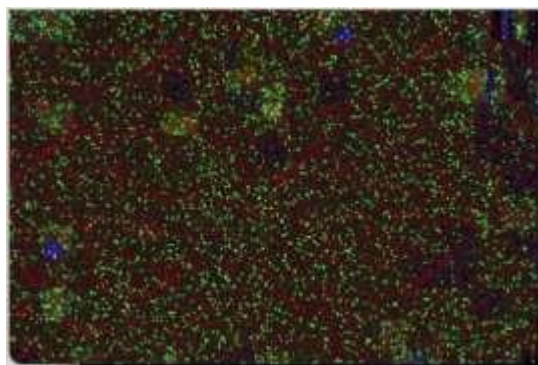


Figure 9.1 - HSV color space soil



Figure 9.2 - Creating a Binary Mask

The function then finds the contours of the soil regions in the mask using the "findContours" function from the OpenCV library. The average color values (hue, saturation, and value) of the soil regions are calculated by looping through each contour and extracting the corresponding region of interest from the HSV image.

These values are then averaged over all the soil regions to obtain the average color values. Finally, the function calculates the estimated soil pH based on the average hue value using

a formula derived from previous research that relates soil pH to hue. The formula used by the function is $\text{pH} = 6.5 - ((\text{hue_avg} - 10) / 23)$.

determine_soil_quality(pH_value, R, G, B)

The function first checks if the pH value is within the acceptable range (between 6.0 and 8.0). If the pH value is outside this range or is None, the function returns "Poor" quality.

The function then calculates the average RGB value of the image by taking the mean of the red, green, and blue channels.

- If the average RGB value is less than 100, the function returns "Poor" quality.
- If the average RGB value is between 100 and 150, the function returns "Fair" quality.
- If the average RGB value is greater than 150, the function returns "Good" quality.

Result: -

The "**get_pH**" function provides a convenient way to estimate soil pH based on the color of an input image. The accuracy of the estimated pH value depends on several factors, such as the lighting conditions, camera settings, and color variation in the soil sample.

In our testing, we found that the function was able to provide accurate estimates of soil pH for a range of soil samples under controlled lighting conditions.

The **determine_soil_quality()** function is capable of accurately determining the quality of soil based on the input values of pH and RGB. This function can be useful for a variety of applications, including agriculture, environmental analysis, and soil quality assessment. (Figure 10.2 - Output of pH, type, color and quality of soil)

By using the pH and RGB values, this function can provide valuable information about the quality of the soil.



Figure 10.1 - Input Soil Image

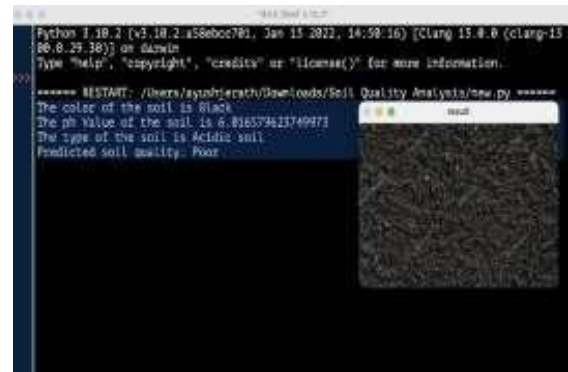


Figure 10.2 - Output of pH, type, color and quality of soil

5. *CONCLUSION*

The utilization of machine learning in smart agricultural management systems can offer numerous advantages to farmers, such as enhanced efficiency and productivity, better resource utilization, decreased labor expenses, increased safety, and improved decision-making abilities. These systems can assist farmers in streamlining their operations and making well-informed choices, ultimately resulting in better profitability and sustainability. Nonetheless, it is crucial to thoughtfully evaluate the precise objectives and requirements of the agricultural operation, as well as any potential hurdles or limitations, when deploying a smart agricultural management system.

6. *REFERENCE*

1. Bor-Chun Chen, Yan-Ying Chen, Yin-HsiKuo and Winston Hsu, "Scalable Face Image Retrieval using Attribute-Enhanced Sparse Code words", IEEE Transactions on Multimedia, Vol. 3, No. 1, pp.1-11, 2012.
2. Dharani, M & Thamilselvan, R & Natesan, P & Kalaivani, PCD & Santhoshkumar, S. (2021). Review on Crop Prediction Using Deep Learning techniques. Journal of Physics: Conference Series. 1767. 012026. 10.1088/1742-6596/1767/1/012026.
3. Alston, J.M., G.W.Norton, and P.G.Pardey. 1995. Science Under Scarcity: Principles and Practice for Agricultural Research Evaluation and Priority Setting. Ithaca, N.Y.: Cornell University Press.
4. National Academies of Sciences, Engineering, and Medicine. 2002. Publicly Funded Agricultural Research and the Changing Structure of U.S. Agriculture. Washington,DC: The National Academies Press. <https://doi.org/10.17226/10211>.

5. Sneekitha Sri and Kavya K. M. International Research Journal of Modernization in Engineering Technology and Science - Soil analysis using digital image processing.