

**CSE3999 - Technical Answers for Real World Problems
(TARP)**

Project Report

**Application for plant disease detection and farmer assistance
guideline**

By

18BCE1024 - P.Maruthi Sai Saketh

18BCE1264 – K.Bharath Sai

B.Tech. Computer Science and Engineering

Submitted to

Dr.Asnath phamila

School of Computer Science and Engineering



VIT[®]

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

June 2021

DECLARATION

I hereby declare that the report titled “Application for plant disease detection and farmer assistance guideline” submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of Dr.Asnath phamila, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai.

Signature of the Candidate

18BCE1024

P.Maruthi Sai Saketh

P.Saketh

18BCE1264

K.Bharath Sai

K.Bharath

CERTIFICATE

Certified that this project report entitled “Application for plant disease detection and farmer assistance guideline” is a bonafide work of P.Maruthi Sai Saketh (18BCE1024), K.Bharath Sai (18BCE1264) and they carried out the Project work under my supervision and guidance for CSE3999 - Technical Answers for Real World Problems (TARP).

Dr.Asnath phamila
SCOPE, VIT Chennai

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, Dr.Asnath phamila, School of Computer Science and Engineering for his/her consistent encouragement and valuable guidance offered to us throughout the course of the project work.

We are extremely grateful to **Dr. R. Jagadeesh Kannan**, Dean and **Dr. S. Geetha**, Associate Dean, School of Computer Science and Engineering (SCOPE), Vellore Institute of Technology, Chennai, for extending the facilities of the School towards our project and for their unstinting support.

We express our thanks to **Dr. Justus S**, Head of the Department, B.Tech. Computer Science and Engineering for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the courses.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

18BCE1024

P.Maruthi Sai Saketh

P. Saketh

18BCE1264

K.Bharath Sai

K. Bharath

ABSTRACT

Crop diseases are a major threat to food security and rapid identification of these diseases remains difficult in many parts of the world due to the lack of the necessary infrastructure. There are some cases where farmers lost their crops due to sudden weather changes and using wrong fertilizers and growing un-seasonal plants. With the increasing of global smart-phone and advancements in computer vision and deep learning made possible for the farmers to overcome the above problems. So we made a farmer assistance web application where the farmer or user can detect the disease of a plant by uploading the infected leaf image and he/she can get the description and suggestions of the disease in English and selected local language, user can predict next day weather conditions in his place based on past 3 months weather data, Question Answering Platform where farmers can post questions/doubts regarding farming and people with agriculture background can answer them. By this application, the above mentioned problems can be minimized. Along with these, we also incorporated market price platform where any user can access it and check the vegetable prices in his place. We used deep learning, machine learning, and web scrapping techniques to develop this application.

CONTENTS

	Declaration	i
	Certificate	ii
	Acknowledgement	iii
	Abstract	iv
1	Introduction	1
1.1	Objective and goal of the project	1
1.2	Problem Statement	1
1.3	Motivation	2
1.4	Challenges	2
2	Literature Survey	2
3	Requirements Specification	5
3.1	Hardware Requirements	5
3.2	Software Requirements	5
4	System Design	6
5	Implementation of System	8
6	Results & Discussion	10
7	Conclusion and Future Work	17
8	References	17

1. Introduction

1.1 Objective and goal of the project

- To build an application which can detect the disease of a plant by just clicking and uploading a photo.
- Suggest description and suitable solutions in local and English languages.
- To collect weather data of the user given place.
- Train the collected weather data and predict next day weather conditions.
- Question answering platform for farmers to get their doubts solve with people in Agriculture background.
- Displaying Market Price of fruits and vegetables for the selected state and district.
- To build a user friendly web application using which can incorporate the above features.

1.2 Problem Statement

Agriculture in India is livelihood for a majority of the population where in 2020, 41.49 percent of the workforce in India were employed in agriculture [1]. Major crops grown in India are rice, wheat, millets, pulses, tea, coffee and jute, etc [2]. Economic growth of Farmers depends on the quality of the products they produce, which relies on the plant's growth and the yield they get. Plants are prone to diseases and late detection of these diseases leads to loss of crop and waste of money. An estimated 15-25 percent of potential crop production is lost due to plant diseases. Manual search and getting solutions of plant diseases is quite difficult for farmers. Even the disease is identified there should know the solutions to overcome the disease. So we need an application which helps the farmers to detect the disease in early stages. Using deep learning we can design a web application where farmers can click the image of the leaf of the crop and identify the disease. They can know the solutions in their local language and they can even assist doubts to overcome the disease in Question answering platform. Another problem is due to sudden weather changes and calamities, there is chance of loss of crop. Weather forecasting helps to predict the future weather conditions and famers can take appropriate measures.

1.3 Motivation

Farming is the main source of raw materials, international trade, nation's revenue, employment, etc. If this great work is incorporated with technology then we may achieve even better results. The main problem for farming is plant diseases due to pests, growing un-seasoned plants and non soil suitable plants, water problem, etc. With the technology we have today we can try to solve some of these problems. The existing method and mostly used for plant disease detection is simply through naked eye observation by experts through which plant disease detection is done. If there are no experts and just by identifying some malicious plants, farmers cannot decide what type of disease that the plant is having and through text search it quite difficult for anyone. So, we thought of developing an application that reduces the crop wastage by easy and early detection of crop diseases. We need to develop an application that is easy and feasible to use by anyone.

1.4 Challenges

The main aim of the project is to assist farmers in farming practices. So, the application needs to be understandable and easily handable. The predictions need to be accurate, so that mistakes won't occur while predicting of diseases. Since, English is not understandable for all; we need to implement local language assistance. Suggestions provided by us may not be enough to address the doubts of farmers, so volunteers with agriculture background can assist them.

2. Literature Survey

S.Santhana Hari et al [1] proposed a new architecture for effective classification of plant diseases. The dataset they used consists of several varieties of plants of both affected and healthy. They trained for 5 different plant diseases using plant leaf images. The proposed CNN architecture consists of 2 sets of 3 convolution layers followed by a Max pooling and dropout layer, 2 sets of convolution layer followed by max pooling layer, 2 fully connected layers and finally a softmax layer where we get output. Regularization is done here by applying Batch Normalization and dropout. During training phase they used mixed dataset of open source images (Plant Village) and field images which were

captured by them. The results show that their proposed architecture (86%) achieved better accuracy than MobileNet (50%).

S.Yegneshwar Yadhav et al [2] designed an optimized real time detection of diseases that affect the plant and the area affected using CNN and K-mean clustering models. They developed a new optimized activation function “ $f(x) = e^x (ex) + \ln(1/(1 + e^{-x}))$ ” to improve convergence which is better than other activation functions like ReLU, tanh, sigmoid, softmax. The CNN model is trained with the new activation function achieved an accuracy of 95%. They used K-means clustering (K = 3) to segment the affected area of predicted infected leaf image.

Sammy V. Militante et al [3] designed a system detect to recognize several plant varieties specifically apple, corn, grapes, potato, sugarcane, and tomato and also detect several diseases of plants. The trained dataset consists of approximately 35,000 images with 32 different classes’ plant varieties and diseases. The data is trained with CNN model for 75 epochs with 32 batch size and achieved an accuracy of 96.5%.

Manoj kumar et al [4] designed a system that detect the disease in coffee plant using convolution neural network and transfer learning. The dataset consists of 1747 images with 5 categories - Healthy leaves, Infected leaves with Phoma, Leaf Miner, Coffee Leaf Rust and Cercospora Spots. 70% of images are used for training the model, 15% each for validation and testing. To create a varied dataset, data augmentation is implemented. They used inception v3 model for transfer learning, which is a 48-layer deep neural network. Accuracy of the model increased with increase in learning rate at first, reaching a maximum at 0.005 with an accuracy of 97.61%, and started to decrease further. Hence, final value of the learning rate in the model is 0.005. Ms.L.Pavithra et al [5] designed a system detects the banana leaf and fruit diseases using image processing and classify the diseases by ANN algorithm. Dataset which consists of various images of leafs and fruits of the banana plant for testing process. Each image is transferred from RGB to gray image and histogram equalization enhancement process is applied. They used Fuzzy c-means clustering to segment the area of interest (banana). Features are extracted using statistical features. Finally the model is trained using ANN and detects the infected banana.

Achyut Morbekar et al [6] developed a system which makes use of a novel approach of the object detection technique to detect plant disease, YOLO. YOLO processes the leaf images at 45 frames per second in real-time, which is faster than other object detection techniques. The dataset consists of thousands of images classified into 25 classes of healthy and diseased plants of 9 various plant species. The two types of bounding boxes are used, one for the entire leaf and other for just to detect infected areas. A darknet-53 network containing 53 Convolutional layers is used in YOLOv3 which is used for training the model. The model is able to detect multiple diseases (if any) at a time for the given image. Vinod Kumar et al [7] developed a plant leaf disease detector using ResNet34 model. The dataset consists of 15200 images that covers 14 crops and is divided into 38 different classes. Residual network34 (ResNet34) is a CNN architecture with 34 layer Convolutional layers whose core building element is a residual block. A residual block makes the use of skip connections to address the degradation problem. They achieved an accuracy of 99.40%.

Neha Rale et al [8] used machine-learning techniques to develop a prediction model for crop yield production. They compared the performance of various linear and non-linear regressor models and concluded that random forest regressor performed the best. The dataset contains two years' of wheat data, geo-located to specific latitude-longitude and counties during winter. The tuned model achieved an R² Value of ~0.83 with a root mean square error (RMSE) of 5.3 (yield values in the dataset range from 10 to 80). The mean absolute percentage error is ~5%.

Ankita H. Tidake [9] developed a farmer assistance application using Machine learning models. The modules included are leaf disease detection and weather forecasting. The climate data obtained from www.indianwaterportal.org The time series historical data over 100 years is taken into study for this experiment. The record attributes are placed as Year/Month, Average Temperature, Cloud Cover, Diurnal Temperature, Maximum Temperature, Minimum Temperature, Cotton production quantity, soya production quantity, Groundnut production quantity for every month of a specified year. The predicted information is sent to the farmer mail.

Research gap:

The main limitation for plant disease detection is data. The image dataset obtained on the internet is regarding plant disease is the plant leaf dataset. Even in the plant leaf, only the limited crops/species which are largely grown in India are present. Increasing the number of plant species and diseases can make the applications even better.

The current research works are lacking in providing suggestions for farmers once the disease is detected. The diseases are displayed in scientific names and most of the farmers might have no prior information about them. So, a farmer assistance application should be included where people with an agriculture background can assist them. Language is another limitation, most of the application suggestions are in English, which cannot be understandable to farmers. Language selection should be included in the application. For weather forecasting, the data that researchers are using is from different timelines and different places. The same type of climate cannot be present in another place. In some places rainfall may be more or temperature may be more and the same type of climate won't be there for all the seasons. So for the given place, past weather conditions should be used while training the model. The season also should be included while training the model.

3 Requirements Specification

3.1 Hardware Requirements

For development of application

- Personal computer
- It is better to use minimum RAM of 8GB for training the deep learning model over large dataset.

3.2 Software Requirements

1. Web application: We used HTML, CSS, and Bootstrap for developing the front end of the website. We used Django as the backend for connecting the pages and running the python scripts. The versions used for the above software are
 - HTML5
 - Bootstrap 4.0.0
 - Django 3.1.7

2. Disease detection: We used Tensorflow Keras library as a deep learning framework to train the mode in python. For translating the English description and suggestions of the effected leaf, we used “googletrans” library in python. The versions used for the above software are
 - tensorflow 2.3.1
 - keras 2.3.1
 - googletrans 3.1.0a0
3. Weather forecasting: To fetch weather dataset for the given input village, we used “wwo_hist” library in python. We used sklearn library to train weather data using Random forest regressor. Versions are
 - sklearn 0.22.1
 - wwo_hist 0.0.7
4. Market Price: We used web scrapping to fetch the vegetable price data, for that we used “requests” and “BeautifulSoup” libraries in python. Versions are
 - requests 2.24.0
 - bs4 4.8.2

4 System Design

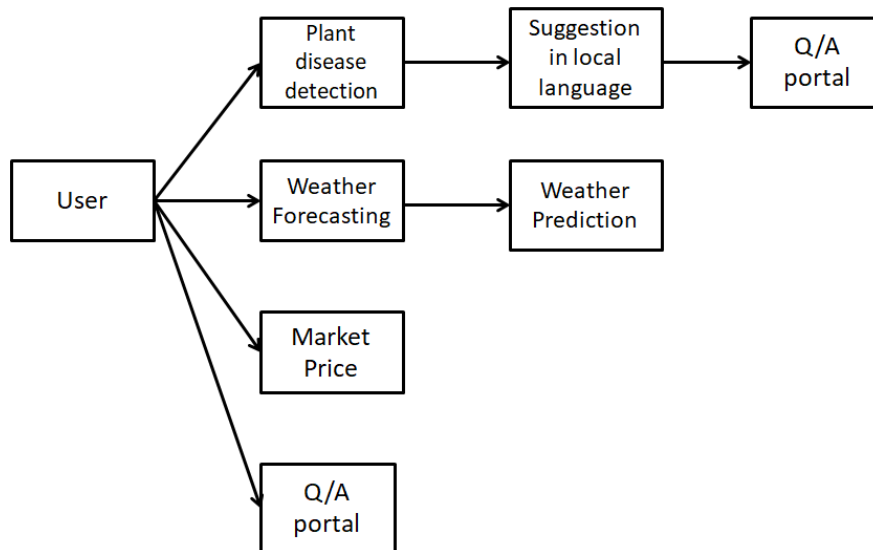


Fig 1

Fig 1 shows the complete flow of the system. User can be a normal user, farmer and volunteer. Unregistered users have access to Market price detection while users who are registered as Farmer or Volunteer have access on Plant disease detection, weather forecasting and question answering portal. User need to upload the infected plat leaf image in a format shown in website. The disease is detected in the backend server and diseases name, disease description and solutions are displayed in the user interface, User can select his/her local language and results are displayed in that language along with English. For weather forecasting, user need to give his place name and using weather API, past 3 months weather data is collected and trained in the backend with random forest regressor. Based on the past one month data, the next day weather conditions are predicted and displayed in the user interface. If there are any other requirement/ doubts, farmer registered user can use question answering portal to raise a question, volunteers who are registered in the same village/ district are displayed with those questions and they can answer and assist them.

We are using HTML, CSS and bootstrap for front end development. We are developing the application with an easily handable interface where anyone can use it. For backend we are using Django since we have Deep learning and machine learning codes which are written in python. To store the details of farmers, volunteers and question answering interface we are using SQL database.

5 Implementation of System

In this section we will describe about how we implemented each module.

5.1 Plant disease detection:

We used CNN deep learning model to train our plant disease detection model. We used the famous “Plant disease” dataset to train our model. The dataset contains 35 categories, which includes diseases of different plants. (Apple, Cherry, Grape, Orange, Peach, Pepper, Potato, Strawberry and Tomato). We used 42,191 images for training and 5266 images for testing. We used Keras library as a framework to implement CNN model. We achieved an accuracy of 94% for testing data. Fig 2 represents the CNN model we used to train the model. The images are resized to 200*200 pixels (height and width). Four sets of Convolution layer and Maxpooling layer are used to train the model. Number of filters in four convolution layers are 32, 64, 64, 64 respectively with kernel size (3,3) for all the layers. All the kernels of four max pooling layers are of size (2,2). After the convolution operations, all the nodes are flattened and finally the output layer contains 35 nodes which is equal to number of classes we have.

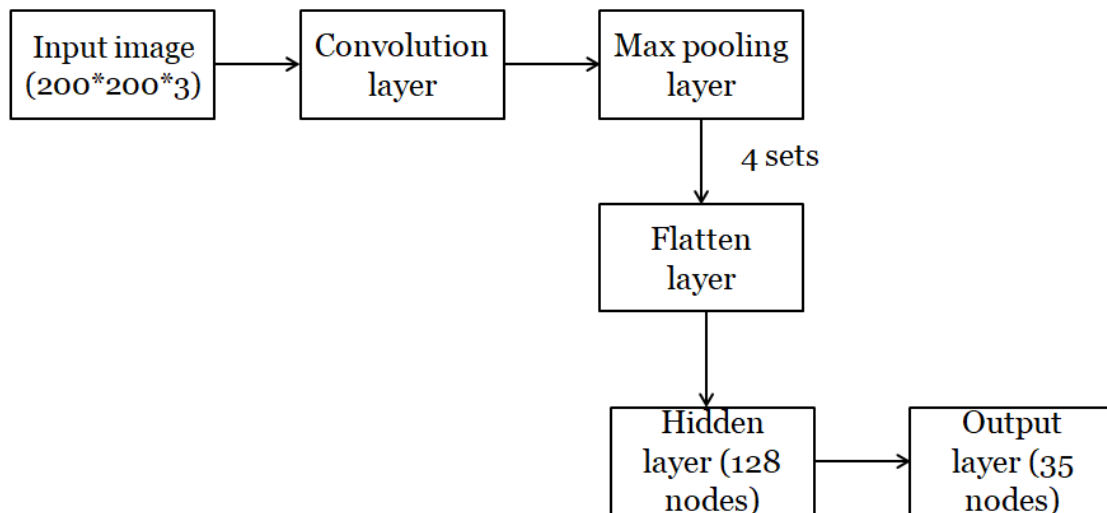


Fig 2: CNN model

Fig 3 describes the process of detection for new images. Farmer uploaded images is taken in the backend. We apply the preprocessing steps and with the help of saved model, disease is detected and displayed to the user.

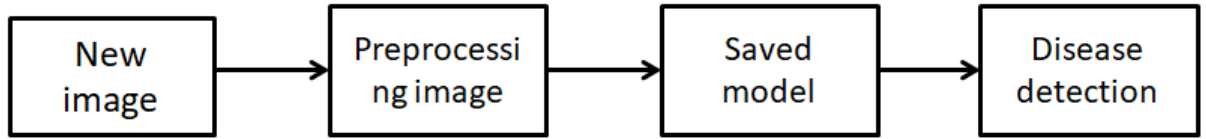


Fig 3: Detecting new images

5.2 Weather forecasting:

Fig 4 and 5 describes the fetching and training of weather dataset for predicting next day weather conditions. Once the user provides the city name as input, a dataset will be downloaded for the past 3 months weather data using API. Out of 24 features, we are selecting 6 features ('maxtempC','mintempC','humidity','precipMM','pressure','tempC'). Two months data is used for training and for each day, we got 8 entries in the dataset which tells weather condition at 8 different times. (with 3 hr difference). A total of 706 entries are used in which 564 for training and 142 for testing. The model is trained in such a way that for every 30 days of data, we are predicting the 31st day temperature, precipitation and humidity. Since weather details are different for different places we can't use a fixed dataset for all type of places and all for all seasons. This can help to more focus weather details on particular region.

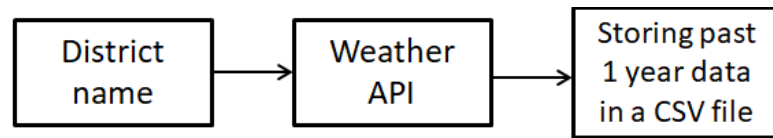


Fig 4: Getting weather dataset using API

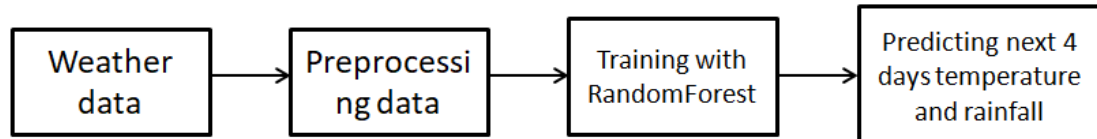


Fig 5: Training of weather dataset

5.3 Market price:

We used Market today price website (<https://market.todaypricerates.com/vegetables-daily-price>) to fetch Market price of different vegetables. User can select state and nearby city to check the prices. With the selected city name and state name a URL will be made which is used to scrap the website and fetch the data.

URL= “[https://market.todaypricerates.com/'+cname+'-vegetables-price-in-'+sname](https://market.todaypricerates.com/'+cname+'-vegetables-price-in-'+sname')”

The fetched data is then displayed in application user interface.

5.4 Suggestions and Translation:

Once the disease is detected, user will be displayed with disease name, disease description and suggestion to overcome the disease. User can select his/her local language and can get the results in the selected one along with English language results. We used “googletrans” API with given input as selected language and English results to convert results into selected language.

6. Results and Discussion

6.1 Plant disease detection

We used 5 layers of CNN layer and achieved an accuracy of 94.68% for testing data. If we use transferring learning approaches like VGG16, resnet we may achieve even better results but due to unavailability of local GPU and huge dataset of 46,000 images we were unable to perform that step. We are comparing the results of using 3 CNN layers and 2 CNN layers below.

Model	Accuracy
5 CNN layers	94.68%
3 CNN layers	90.47%
2 CNN layers	83.21%

6.2 Weather forecasting:

Mean absolute error for

Singapore dataset:

Feature	Baseline model	Training	Testing
Temperature	1.40	0.14	0.34
Precipitation	0.72	0.13	0.41
Humidity	6.34	0.65	1.57

Nellore dataset:

Feature	Baseline model	Training	Testing
Temperature	2.26	0.19	0.56
Precipitation	0.12	0.03	0.08
Humidity	11.03	1.03	1.95

We experimented with linear regression and Random forest regressor models. Below are the results obtained for the Nellore dataset.

Mean absolute error for predicting Temperature with baseline value 2.26,

Data	Linear regression	Random forest regressor
Training	0.37	0.19
Testing	0.57	0.56

Mean absolute error for predicting Precipitation with baseline value 0.12,

Data	Linear regression	Random forest regressor
Training	0.11	0.03
Testing	0.23	0.08

Mean absolute error for predicting humidity with baseline value 11.03,

Data	Linear regression	Random forest regressor
Training	1.45	1.03
Testing	2.00	1.95

From the above results, it is clear that random forest regression model is working better than linear regression model.

6.3 Other results of the website:

Fig 6

Fig 7

Fig 6 and Fig 7 are snippets of the developed website where farmers and volunteers need to provide their details and signup for the website. The details are stored in the postgres sql database. Username is the primary key which should be unique and if same username is given then a popup window will be displayed to change username. Village and district details are helpful in Q/A portal.

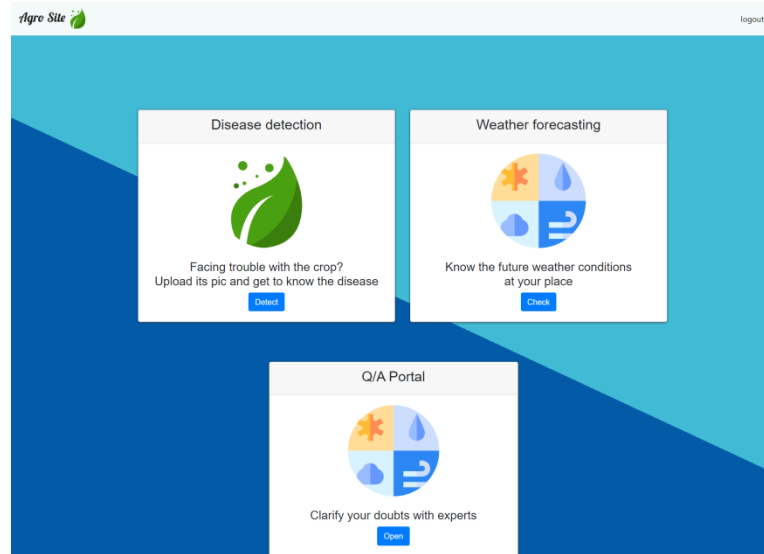


Fig 8

Once farmer logs in to his website, he can choose any of above three (disease detection, weather forecasting, Q/A portal) as shown in Fig 8.

Plant disease detection:

 The image shows a form for plant disease detection. It has a light pink background. At the top, it says 'Upload Image File Here:' followed by a 'Choose File' button and the text 'image (586).JPG'. Below this is a 'Select language' label and a dropdown menu currently showing 'Bengali'. At the bottom of the form is a blue 'Detect' button.

Fig 9

Fig 9 is the snippet from developed website where user needs to upload the image file and can select a language. As shown in Fig 10, the output consists of detected disease name, description of disease and suggestions. The output will be in English language as well as selected local language.

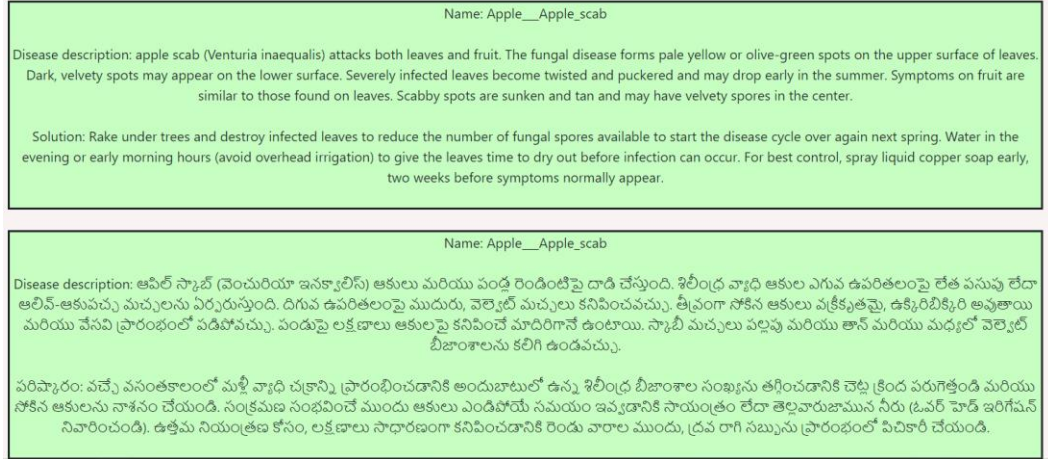


Fig 10

Weather forecasting:

Once the user provides the city name as input, a dataset will be downloaded for the past 3 months weather data using API. The model is trained in such a way that for every 30 days of data, we are predicting the 31st day temperature, precipitation and humidity. The results of training using Random forest regressor for two datasets/cities are provided below.

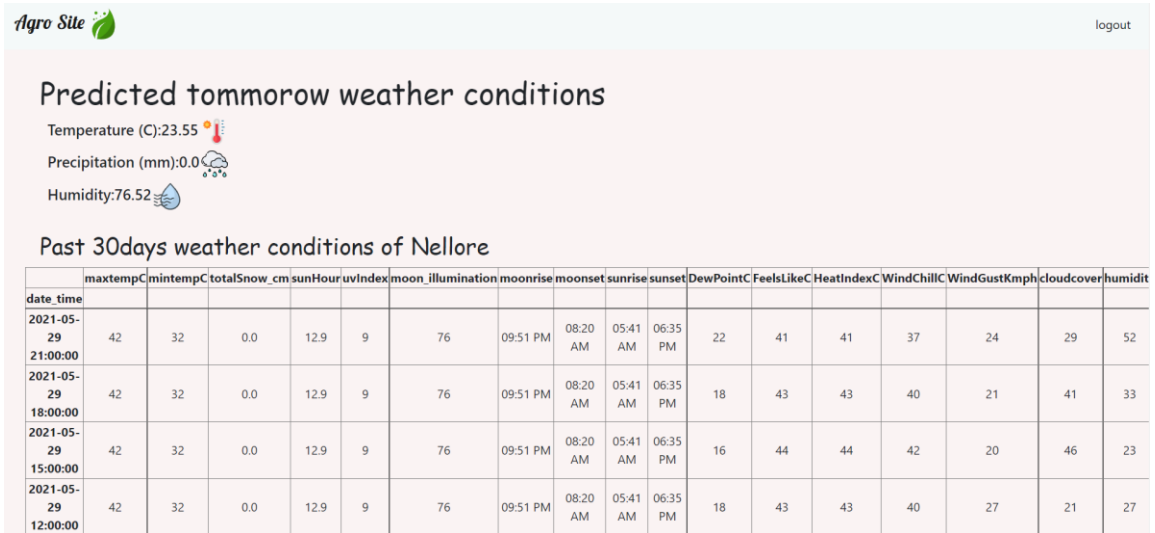
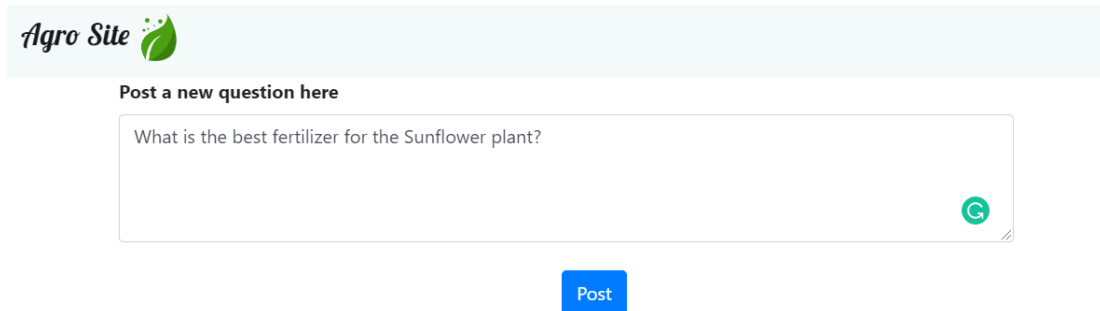


Fig 11

Fig 11 is the output snippet of predicted weather conditions for Nellore city.

Q/A portal

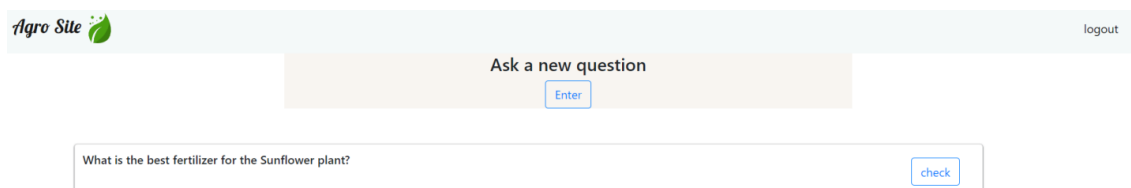
Once farmer login to his account, he can post a question as shown in Fig 12 in Q/A portal.



The screenshot shows the 'Agro Site' logo at the top left. Below it, the text 'Post a new question here' is displayed. A text input field contains the question 'What is the best fertilizer for the Sunflower plant?'. To the right of the input field is a green circular icon with a white 'G' and a small edit icon. Below the input field is a blue 'Post' button.

Fig 12

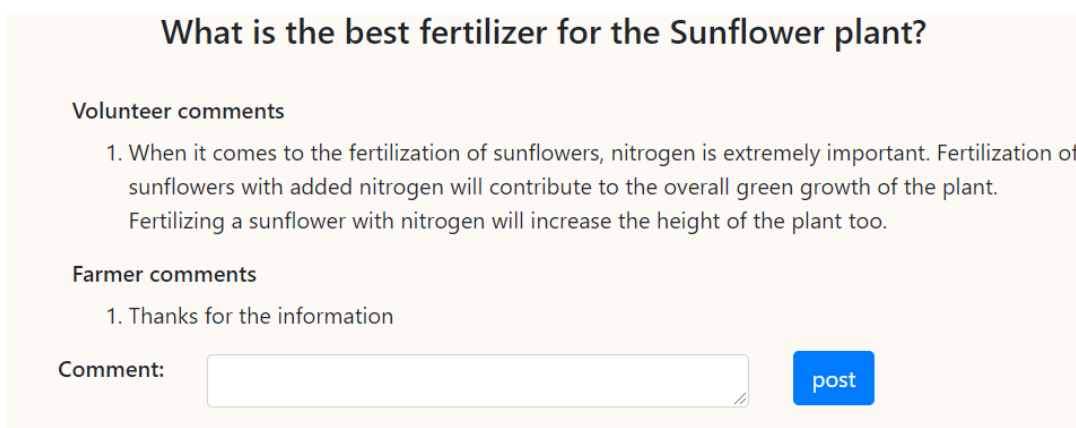
Farmer will displayed with the questions he raised and volunteers are displayed with the farmer questions in his Village/District. This helps to reduce the count of questions volunteers are displayed with. The questions are displayed as shown Fig 13.



The screenshot shows the 'Agro Site' logo at the top left and a 'logout' link at the top right. Below the logo, the text 'Ask a new question' is displayed. A text input field contains the question 'What is the best fertilizer for the Sunflower plant?'. To the right of the input field is a blue 'check' button. Below the input field is a blue 'Enter' button.

Fig 13

Volunteer can comment his suggestions which are displayed to user and the threa can continue. Fig 14 is website snippet showing the conversation for the question raised.



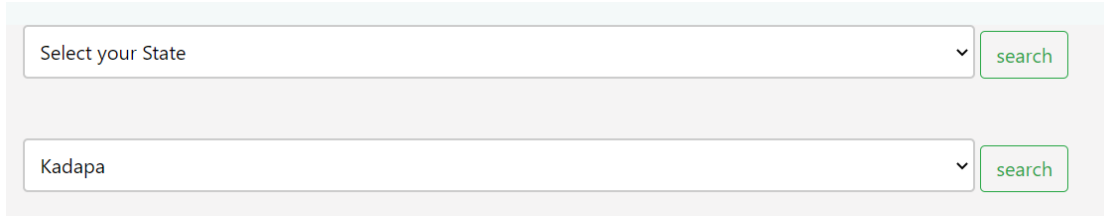
The screenshot shows a conversation thread for the question 'What is the best fertilizer for the Sunflower plant?'. The thread is titled 'What is the best fertilizer for the Sunflower plant?'. Below the title, the text 'Volunteer comments' is displayed. A list of comments follows: '1. When it comes to the fertilization of sunflowers, nitrogen is extremely important. Fertilization of sunflowers with added nitrogen will contribute to the overall green growth of the plant. Fertilizing a sunflower with nitrogen will increase the height of the plant too.' Below the volunteer comments, the text 'Farmer comments' is displayed. A list of comments follows: '1. Thanks for the information'. At the bottom, there is a 'Comment:' label, a text input field, and a blue 'post' button.

Fig 14

Market Price:

User can select state and district as shown in Fig 15 and vegetable prices will be displayed to the user as shown in Fig 16 using web scrapping from the website

<https://market.todaypricerates.com/vegetables-daily-price>.



The screenshot shows a web interface with two dropdown menus. The first dropdown is labeled 'Select your State' and has a green 'search' button next to it. The second dropdown is labeled 'Kadapa' and also has a green 'search' button next to it.

Fig 15

Market Price in				
Vegetable Name	Unit	Market Price	Retail Price	Shopping Mall
Amaranth leaves	Kg / Pcs	₹ 8	₹ 9 - 10	₹ 10 - 13
Amla	Kg / Pcs	₹ 95	₹ 109 - 121	₹ 114 - 157
Ash gourd	Kg / Pcs	₹ 18	₹ 21 - 23	₹ 22 - 30
Baby corn	Kg / Pcs	₹ 58	₹ 67 - 74	₹ 70 - 96
Banana flower	Kg / Pcs	₹ 12	₹ 14 - 15	₹ 14 - 20
Beetroot	Kg / Pcs	₹ 31	₹ 36 - 39	₹ 37 - 51
Bell Pepper (Capsicum)	Kg / Pcs	₹ 38	₹ 44 - 48	₹ 46 - 63
Bitter gourd	Kg / Pcs	₹ 28	₹ 32 - 36	₹ 34 - 46
Bottlegourd	Kg / Pcs	₹ 28	₹ 32 - 36	₹ 34 - 46
Butter beans	Kg / Pcs	₹ 43	₹ 49 - 55	₹ 52 - 71

Fig 16

7. Conclusion and Future Work

In this project we made a farmer assistance application in which farmers and any user can detect plant diseases by uploading the affected part image and get the description in local language and English language, can check next weather conditions of his/her place, can communicate with agriculture experts and clear their doubts and also can check the current vegetable prices in his place. We used CNN for plant disease detection and achieved an accuracy of 94%, we compared Random forest regressor and linear regression training values in terms of mean absolute error and choose Random forest algorithm as it achieved better results. Future work includes, including more diseases in the dataset and using Transfer learning approach to improve the accuracy. For weather forecasting we can predict and display the results for next week weather conditions instead of a single day. In question answering portal, we are displaying the questions of farmers to volunteers who are in same village/ district to reduce the number of questions that volunteers need to check. The other solutions which can improve the performance are using filters to filter out the villages/districts/states the volunteers want to check for, displaying farmer/volunteer details which can help them to contact with phone and also can understand the experience of volunteers. For market price prediction, currently we are displaying only vegetable prices of selected district and this only works for Indian states and districts. We can extend this to prices of fruits and other farming products.

8. REFERENCES

- [1] “Detection of Plant Disease by Leaf Image Using Convolutional Neural Network”, 2019 by S.Santhana Hari, M.Sivakumar, P.Renuga
- [2] “Plant Disease Detection and Classification using CNN Model with Optimized Activation Function”, 2020 by S.Yegneshwar Yadhav, T.Senthilkumar, S.Jayanthi
- [3] “Plant Leaf Detection and Disease Recognition using Deep Learning”, 2019 by Sammy V. Militante1, Bobby D. Gerardoij, Nanette V. Dionisio
- [4] “Disease Detection in Coffee Plants Using Convolutional Neural Network”, 2020 by Manoj Kumar, Pranav Gupta, Puneet Madhav
- [5] “Detection of Banana Leaf and Fruit Diseases Using Neural Networks”, 2020 by N.Saranya, L.Pavitra, N.Kanthimathi
- [6] “Crop Disease Detection Using YOLO”, 2020 by Achyut Morbekar, Ashi Parihar, Rashmi

- [7] “ResNet-based approach for Detection and Classification of Plant Leaf Diseases”, 2020 by Vinod Kumar, Hritik Arora, Harsh
- [8] "Prediction of Crop Cultivation," 2019 by N. Rale, R. Solanki, D. Bein, J. Andro-Vasko
- [9] "Design Efficient Model To Increase Crop Yield Using Deep Learning," 2019 by A. H. Tidake, Y. k. Sharma and V. S. Deshpande,

APPENDIX

Sample code for plant disease detection:

```
batch_size = 128

from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os

path1=r"E:\6th sem\tarp deep\dataset1\test"
classes_train=os.listdir(path1)


train_datagen = ImageDataGenerator(rescale=1/255)
train_generator = train_datagen.flow_from_directory(
    'dataset1/train',
    target_size=(200, 200),
    batch_size=batch_size,
    classes = classes_train,
    class_mode='categorical')


val_datagen = ImageDataGenerator(rescale=1/255)
val_generator = val_datagen.flow_from_directory(
    'dataset1/val',
    target_size=(200, 200),
    batch_size=batch_size,
    classes = classes_train,
    class_mode='categorical')


import tensorflow as tf
```



```

model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(200, 200, 3)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2, 2),

    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Flatten(),

    tf.keras.layers.Dense(128, activation='relu'),

    tf.keras.layers.Dense(35, activation='softmax')
])
model.summary()
from tensorflow.keras.utils import plot_model
plot_model(model, to_file='model_plot.png', show_shapes=True,
show_layer_names=True)

```

```

from tensorflow.keras.optimizers import RMSprop
from tensorflow import keras

model.compile(loss='categorical_crossentropy',
              optimizer=RMSprop(lr=0.0005),
              metrics=['acc'])

total_sample=train_generator.n
n_epochs = 30

checkpoint = keras.callbacks.ModelCheckpoint('abc.h5', monitor='val_loss', verbose=0,
save_best_only=True, save_weights_only=False, mode='auto', period=1)

history = model.fit(
    train_generator,
    steps_per_epoch=int(total_sample/batch_size),
    epochs=n_epochs,
    validation_data = val_generator,
    callbacks = [checkpoint],
    verbose=1)

from tensorflow.keras.models import load_model
model = load_model('abc.h5')
# summarize model.
model.summary()

import numpy as np
from keras.preprocessing import image

```

```

test_image = image.load_img(r'E:\6th sem\tarp
deep\dataset1\test\Corn___Common_rust\image (201).JPG', target_size = (200, 200))
test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis = 0)
result = model.predict(test_image)
model.evaluate_generator(val_generator)

```

Sample code for weather prediction:

```

import pandas as pd
import numpy as np
df = pd.read_csv('nellore.csv')
features=['maxtempC','mintempC','humidity','precipMM','pressure','tempC']
df_f=df[features].values
df_f1=df['tempC'].values
df_f2=df['precipMM'].values
df_f3=df['humidity'].values

X=[]
y=[]
y1=[]
y2=[]
for i in range(30,len(df_f),1):
    X.append(df_f[i-30:i])
    y.append(df_f1[i])
    y1.append(df_f2[i])
    y2.append(df_f3[i])

X=np.array(X)
y=np.array(y)
y1=np.array(y1)
y2=np.array(y2)

```

```

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
X_train1, X_test1, y_train1, y_test1 = train_test_split(X, y1, test_size = 0.2)
X_train2, X_test2, y_train2, y_test2 = train_test_split(X, y2, test_size = 0.2)

X_train=X_train.reshape(len(X_train),-1)
X_test=X_test.reshape(len(X_test),-1)
X_train1=X_train1.reshape(len(X_train1),-1)
X_test1=X_test1.reshape(len(X_test1),-1)
X_train2=X_train2.reshape(len(X_train2),-1)
X_test2=X_test2.reshape(len(X_test2),-1)

print("Training Features Shape:", X_train.shape)
print("Training Labels Shape:", y_train.shape)
print("Testing Features Shape:", X_test.shape)
print("Testing Labels Shape:", y_test.shape)

from sklearn.metrics import mean_absolute_error
y_pred=[y_train.mean()]*len(y_train)
print("baseline mae for temperature: ",round(mean_absolute_error(y_train,y_pred),5))

y_pred1=[y_train1.mean()]*len(y_train1)
print("baseline mae for precipitation : ",round(mean_absolute_error(y_train1,y_pred1),5))

y_pred2=[y_train2.mean()]*len(y_train2)
print("baseline mae for humidity: ",round(mean_absolute_error(y_train2,y_pred2),5))

from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

```

```

from sklearn.ensemble import RandomForestRegressor
rf = make_pipeline(StandardScaler(),RandomForestRegressor())
rf1 = make_pipeline(StandardScaler(),RandomForestRegressor())
rf2 = make_pipeline(StandardScaler(),RandomForestRegressor())

rf.fit(X_train, y_train)
rf1.fit(X_train1, y_train1)
rf2.fit(X_train2, y_train2)

print('Random forest regressor model training mae for temperature: ',
mean_absolute_error(y_train, rf.predict(X_train)))
print('Random forest regressor model testing mae for temperature: ',
mean_absolute_error(y_test, rf.predict(X_test)))

print('Random forest regressor model training mae for precipitation: ',
mean_absolute_error(y_train1, rf1.predict(X_train1)))
print('Random forest regressor model testing mae for precipitation: ',
mean_absolute_error(y_test1, rf1.predict(X_test1)))

print('Random forest regressor model training mae for humidity: ',
mean_absolute_error(y_train2, rf2.predict(X_train2)))
print('Random forest regressor model testing mae for humidity: ',
mean_absolute_error(y_test2, rf2.predict(X_test2)))

y_pred=rf.predict(X_test)
y_pred=y_pred.reshape(len(X_test),1)
errors=abs(y_pred-y_test)
mape=100*(errors/y_test)
accuracy=100-np.mean(mape)
print("Random forest model accuracy for temperature ", round(accuracy,2),"%")

y_pred=rf1.predict(X_test1)

```

```
y_pred=y_pred.reshape(len(X_test1),1)
errors=abs(y_pred-y_test1)
mape=100*(errors/(y_test1+0.0001))
accuracy=100-np.mean(mape)
print("Random forest model accuracy for precipitation ", round(accuracy,2),"%")
```

```
y_pred=rf2.predict(X_test2)
y_pred=y_pred.reshape(len(X_test2),1)
errors=abs(y_pred-y_test2)
mape=100*(errors/y_test2)
accuracy=100-np.mean(mape)
print("Random forest model accuracy for humidity ", round(accuracy,2),"%")
```

```
d1=rf.predict([df_f[1:31].reshape(-1)])
d2=rf1.predict([df_f[1:31].reshape(-1)])
d3=rf2.predict([df_f[1:31].reshape(-1)])
```