Easy Agriculture: Crops' disease detection, pesticide, fertilizer and crop recommendations

Ravikiran Kolagani[1*], *Jagadeeshwar Reddy* Venkatram[1], *Varshith* Kemmasaram[1], *Gagan Sai* Ravilla[1], *Vishwateja* Vadla[1], and *Sai Ketan Reddy* Julakanti[1]

[1]Department of Information Technology, GRIET, India

Abstract. Around the world due to pests and pathogens almost 50% of the agricultural produce is lost which is so alarming given the fact that many people die everyday due to starvation in poor nations. Crop diseases disturb the normal growth and physiological processes. It is estimated that every year 20-40% of crop loss is reported and, in some cases, whole production gets destroyed. So, to produce higher yield and for sustainable agriculture it is important to identify any diseases from the early stage itself. Technology can do a great help in this cause to detect plant disease by using various AI techniques. It is also important to recommend proper pesticides for the persisting disease. The model proposed is based upon a 9 layer resnet deep learning algorithm that takes in present time images of various crops and detects the disease & also recommends the suitable pesticide. Plant Village Dataset taken from Kaggle comprising 87000 images (38 Classes,13 Crops) is used. A custom dataset is also built consisting of disease-description-measures to be taken-pesticide or fertilizer to be used. The end system developed also has two other models integrated that are used for crop and fertilizer recommendations. They are built using the Random Forest Classifier algorithm and a parameter conditional statements function.

1 Introduction

With modern practices and innovative technology we humans are able to produce food required for more than 7 billion people. Even though food security remains a major issue across the globe as threats exist from climate change, decrease in pollinators and significantly by crop diseases. But many diseases can be cured if they are identified at an early stage which is a great fact that needs to be focussed and worked upon. Current practices for crop disease identification are based upon significant support from agriculture organisations and local plant clinics, but millions around the world do not have access to these resources, especially farmers in developing and underdeveloped countries. Manual identification of crop diseases is a tedious task and at times is inefficient which can lead to crop loss in the long run. The agriculture industry is in dire need of a tool that can efficiently and accurately diagnose crop diseases while being free, easy-to-use, and widely accessible and also which recommends what pesticides to use. Tools which are available right now use deep learning techniques like

* Corresponding author: ravi.10541@gmail.com

eAGROBOT, YOLO and edge detection algorithms. They take images of the crop in its current stage and detect the diseases. Latest models developed are all based upon hybrid algorithms and achieve higher accuracy. But they do not recommend which pesticides to use after detecting the disease of crops which don't solve the problem from end to end.

2 Literature Survey

SamyakShrimali et.al., [1] they have used MobileNetV2 model architecture was chosen for building the model and training image filters used are Gaussian Blurred, Laplacian Sharpening, Canny Edge Detection and it yielded an accuracy of 95.7% and F1 score of 96.1%. Dataset consisted of 87,860 leaf images split into 38 classes. Each class label is a crop-disease pair, and the goal was to predict that label from an image of the crop leaf. Model diagnoses 26 crop diseases of 14 different species. It doesn't require the internet, is easy to use and wildly scalable. All the images in the dataset were resized to 224*224 pixels. For building the model the dataset was split into a ratio of 85:10:5 for training, validation, and testing, respectively.

Leninisha Shanmugam et.al., [2] the proposed methodology is used for automatic detection of plant diseases. For training the model remote sensing images are used. After the model is trained and ready to be used remote sensing images are passed as input. Both for trained and input images using MATLAB function is used to separate red, blue, green layers from the images. This is done so as to obtain threshold values from images. If threshold values are found to be greater or lesser than the threshold they are converted to grayscale and CANNY's edge detection is applied on them to obtain the edges which mostly help to identify the diseases. In addition to that they have even used histogram analysis techniques and matching for accurate and effective disease identification. Alert to farmers is sent using Message Transferring Technology in MATLAB.

Achyut Morbekar et.al., [3] You Only Look Once (YOLO)v3 algorithm is used for detecting a plant disease. Plant Village dataset from Kaggle comprising 9 various plant diseases and 25 classes is used for training and testing the model. It is stated that Deep Learning techniques work well on large datasets so they have used augmentation technique Generative Adversial Networks (GAN) to artificially increase the size of the dataset. YOLOv3 algorithm comprises a darknet network which contains 53 convolutional networks and it is faster than other object detection models.

Omkar Kulkarni et.al., [4] Plant Village dataset has been used to train the model. The model is based upon Convolutional Neural Networks (CNN). During preprocessing segmentation is performed on all the images so that it extracts only relevant parts of the image. Model first identifies the crop based upon the image given as input and then the second step is to identify the crop disease. Two types of models MobileNet and InceptionV3 are compared but InceptionV3 performs better than MobileNet in terms of accuracy and validation loss. Transfer learning models are used to achieve better performance.

Sanket Solanke et.al., [5] the proposed model IoT smart device is deployed in the farm for automatically spraying the pesticide after getting consent from the farmer via an application used by the latter. Workflow of the model starts after the user opens the application and the smart device in the field notifies the user if there are any diseases identified by capturing and image preprocessing. If the plant is infected, a smart device sprays the right amount of pesticide onto the crop after getting permission from the user. All the data collected is stored in the cloud so as to integrate it with the mobile application.

3 System Architecture

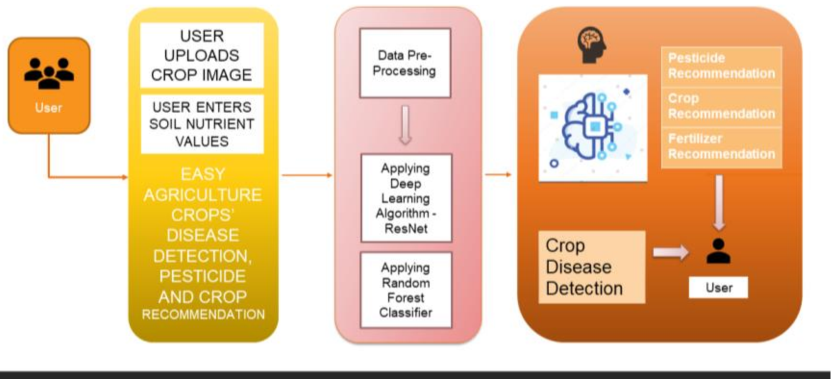

User
USER UPLOADS CROP IMAGE
USER ENTERS SOIL NUTRIENT VALUES
EASY AGRICULTURE CROPS' DISEASE DETECTION, PESTICIDE AND CROP RECOMMENDATION
Data Pre-Processing
Applying Deep Learning Algorithm - ResNet
Applying Random Forest Classifier
Pesticide Recommendation
Crop Recommendation
Fertilizer Recommendation
Crop Disease Detection
User

Fig. 1. System Architecture.

System architecture includes three models: crop disease detection and pesticide recommendation, fertilizer recommendation model, crop recommendation model. Each of the models mentioned solve an agriculture problem. All the three models are integrated into a website and we named it Easy Agriculture.

Crop disease detection model uses resnet algorithm for detecting the disease. User needs to upload a crop leaf image to the model. Model converts it into a 256*256 size and passes onto the algorithm. Algorithm gives the label of the predicted class. Now this label is used for picking out the corresponding pesticide to the disease by looking out in the dataset dictionary which contains crop disease class label as key and pesticide as value. In case if the predicted class is not a disease then a suitable fertilizer is recommended based upon the crop identified. Recommendation has 1 pesticide or 1 fertilizer.

Crop recommendation model uses a random forest classifier algorithm.It takes soil nutrient values: nitrogen, potassium, phosphorus, rainfall(in mm), pH level as direct inputs. It also asks the user to select the geographical location of the farm i.e city and state. It only contains India states and cities. When the user clicks on the predict button of the webpage, a call is made to the openweather api to fetch the current humidity, temperature of that place. All the inputs are set up into an array. This array is passed onto the random forest classifier model. Model gives crop recommendations. This is displayed on the result webpage.

Fertilizer recommendation model uses conditional statements based upon the crop selected by the user. Other inputs taken from the user are nitrogen, phosphorus, potassium. Fertilizer recommended is by comparing the threshold soil nutrient values for the crop selected and values entered. Model gives a key based upon the comparison. Value for this key is fetched from the fertilizer recommendation dataset dictionary. This value is shown as the output to the user.

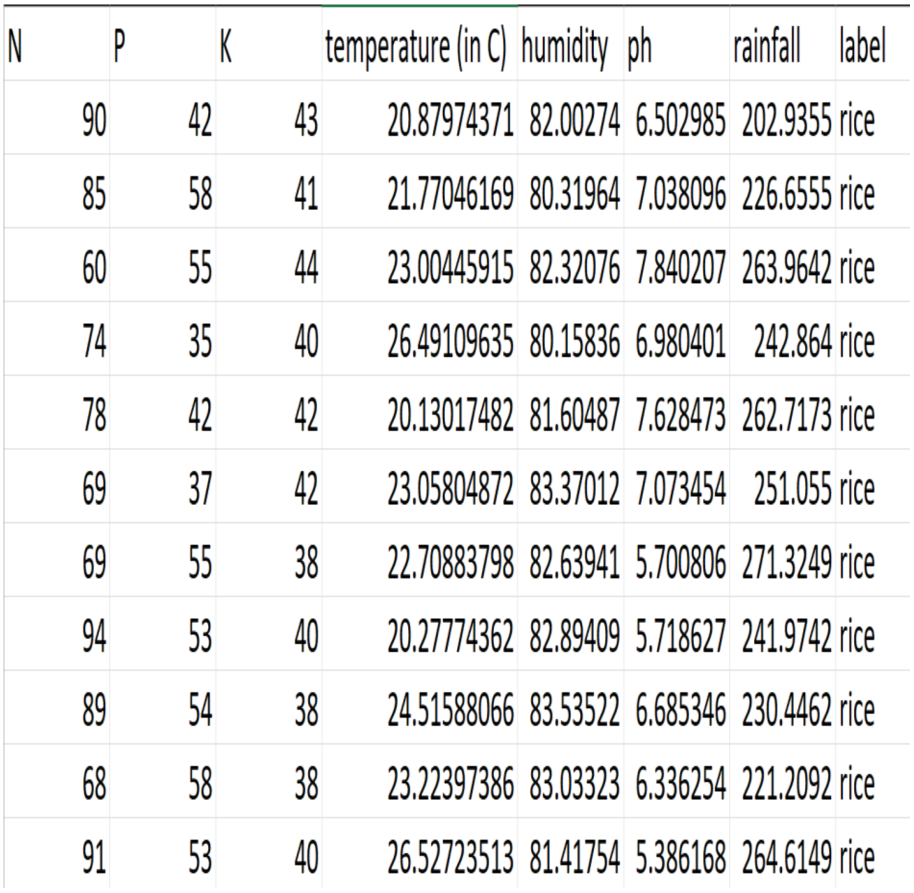
Website is built using the Flask framework. It comprises a homepage and three other dynamic web pages each for the three models.

https://doi.org/10.1051/e3sconf/202339101055

4 Methodology

4.1 Dataset

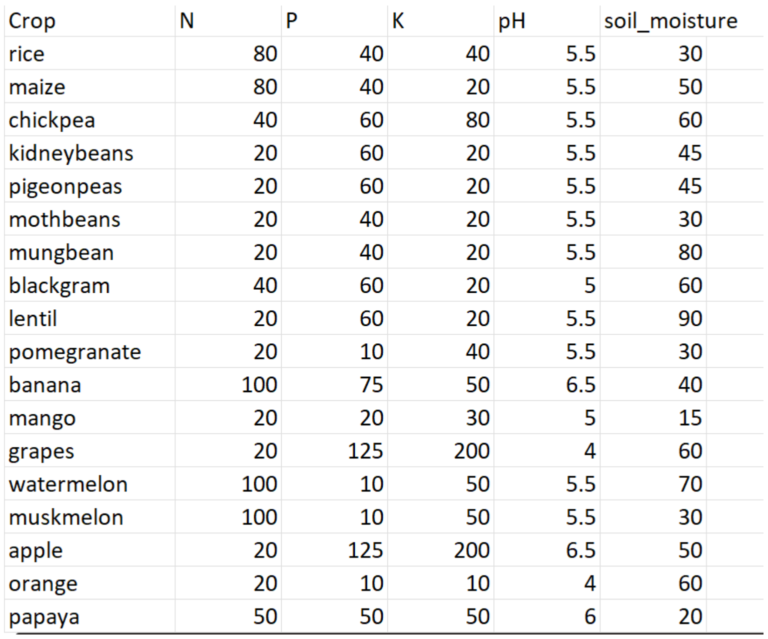
The proposed system comprises 3 models: residual neural network model, random forest classifier model and conditional statements function model. Total 4 datasets are used, of which 3 are collected from Kaggle. They are: plant leaf image dataset comprising 87000 images, crop recommendation dataset, fertilizer dataset, pesticide dataset.



N	P	K	temperature (in C)	humidity	ph	rainfall	label
90	42	43	20.87974371	82.00274	6.502985	202.9355	rice
85	58	41	21.77046169	80.31964	7.038096	226.6555	rice
60	55	44	23.00445915	82.32076	7.840207	263.9642	rice
74	35	40	26.49109635	80.15836	6.980401	242.864	rice
78	42	42	20.13017482	81.60487	7.628473	262.7173	rice
69	37	42	23.05804872	83.37012	7.073454	251.055	rice
69	55	38	22.70883798	82.63941	5.700806	271.3249	rice
94	53	40	20.27774362	82.89409	5.718627	241.9742	rice
89	54	38	24.51588066	83.53522	6.685346	230.4462	rice
68	58	38	23.22397386	83.03323	6.336254	221.2092	rice
91	53	40	26.52723513	81.41754	5.386168	264.6149	rice

Fig. 2. Crop Recommendation Dataset

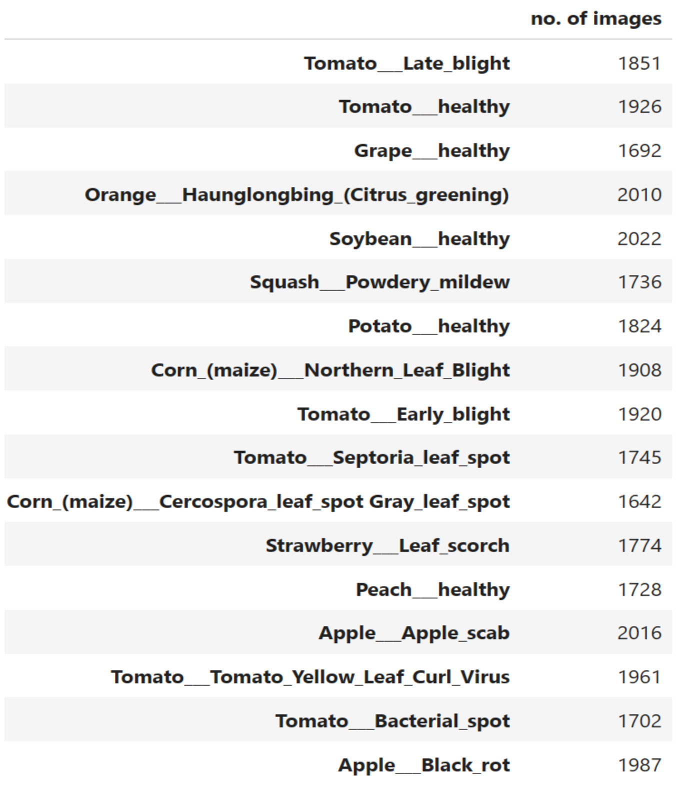
It contains data of 21 crops and parameters included are nitrogen, phosphorus, potassium, temperature, humidity, ph,rainfall and label.



Crop	N	P	K	pH	soil_moisture
rice	80	40	40	5.5	30
maize	80	40	20	5.5	50
chickpea	40	60	80	5.5	60
kidneybeans	20	60	20	5.5	45
pigeonpeas	20	60	20	5.5	45
mothbeans	20	40	20	5.5	30
mungbean	20	40	20	5.5	80
blackgram	40	60	20	5	60
lentil	20	60	20	5.5	90
pomegranate	20	10	40	5.5	30
banana	100	75	50	6.5	40
mango	20	20	30	5	15
grapes	20	125	200	4	60
watermelon	100	10	50	5.5	70
muskmelon	100	10	50	5.5	30
apple	20	125	200	6.5	50
orange	20	10	10	4	60
papaya	50	50	50	6	20

Fig. 3. Fertilizer Recommendation Dataset

It contains recommendations for 22 crops.



	no. of images
Tomato___Late_blight	1851
Tomato___healthy	1926
Grape___healthy	1692
Orange___Haunglongbing_(Citrus_greening)	2010
Soybean___healthy	2022
Squash___Powdery_mildew	1736
Potato___healthy	1824
Corn_(maize)___Northern_Leaf_Blight	1908
Tomato___Early_blight	1920
Tomato___Septoria_leaf_spot	1745
Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot	1642
Strawberry___Leaf_scorch	1774
Peach___healthy	1728
Apple___Apple_scab	2016
Tomato___Tomato_Yellow_Leaf_Curl_Virus	1961
Tomato___Bacterial_spot	1702
Apple___Black_rot	1987

Fig. 4. Crop Leaf Images Dataset

4.2 Data loading and Preprocessing:

4.2.1 Function: splitfolders.ratio() and train_test_split():

To create training, validation datasets from image dataset, splitfolders.ratio() is used. Splitting is done as 80% for training, 20% for validation. A new directory comprising 25 leaf images is created for testing.

For crop recommendation dataset train_test_split() from sklearn is used for creating training,validation and test datasets as it is a .csv data file.

4.2.2 Function: ImageFolder():

ImageFolder() is imported from torchvision.datasets().It helps in loading custom datasets.After loading the data it converts the pixels of each image to 0-1 as neural networks work pretty well with normalized data.

4.2.3 Function: DataLoader():

DataLoader is a subclass from torch.utils.data. Loading large memory consuming datasets can be done using dataloader. It takes in batch_size, shuffle, num_workers as arguments.

4.2.4 Residual Neural Networks

The ResNet algorithm is a big breakthrough algorithm in the deep learning field. It was introduced in 2015. It overcomes the vanishing gradient problem that was witnessed in many prior advanced deep learning algorithms like ImageNet, CNN, VGGNet etc. It uses skip connections, convolutional blocks and identity blocks. It doesn't use pooling in the middle layers of the architecture. Padding, strides, filter sizes are present like in previous groundbreaking algorithms. [6]

4.2.5 Random Forest Classifier

Random Forest is an ensemble technique in machine learning. It reduces the given data into samples and applies many decision trees upon it. Classification result is the majority voting of the decision trees' results. Random Forest Classifier function takes two arguments as input: n_estimators and random_state.First argument is for how many decision trees to be used and second argument signifies the randomness of splitting the data into samples. [7]

4.3 Workflow and Algorithm

4.3.1 Resnet

- First, the necessary libraries are imported, including TensorFlow, NumPy, Pandas, Matplotlib, Torchsummary.
- The plant leaf image dataset is loaded from a local directory which is divided into train,valid,test sub directories.
- Data is visualized to get a sense of it.

- Using ImageFolder function data is prepared for training.
- Then data loader is used for loading the dataset by specifying the required arguments.
- ResNet model is created using a sequential model from keras. It comprises convolutional block and residual functions. Relu activation function is used.
- The model is trained using the fit_one_cycle() function and the training and validation generators. Learning rate scheduler is used rather than specifying a particular learning rate to the whole training.
- Finally, the model is evaluated on the test set using the evaluate() function.

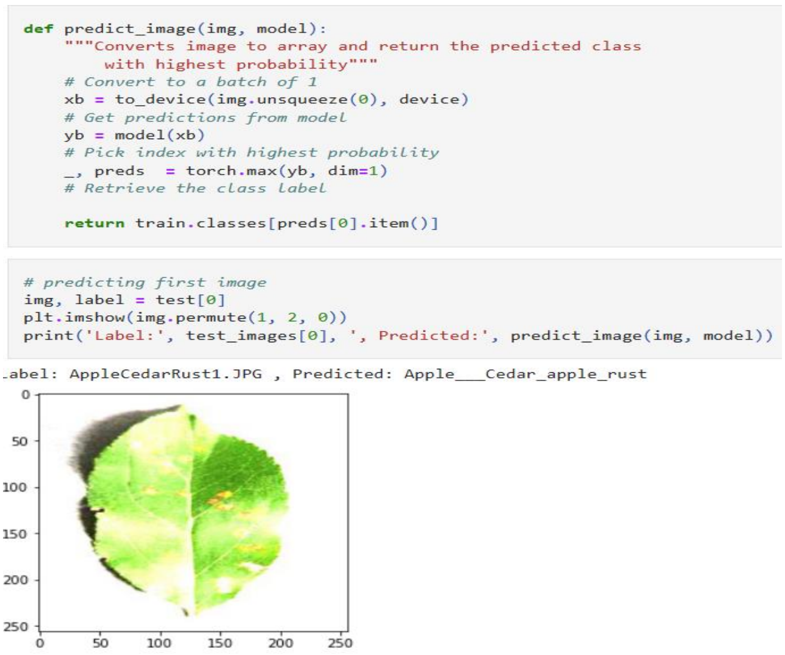
4.3.2 Random Forest Classifier

- Randomforest classifier model is imported from sklearn.ensemble library.
- Using 20 estimators and randomstate=0 as arguments a new random forest classifier model is initiated.
- Using train_test_split function in sklearn crop recommendation dataset is split into train,test data.
- Model is fitted on train data.
- Using predict() function model is tested on test data.
- classification_report is used for calculating model's performance metrics.

5 Results and Analysis

5.1 Input:

The input for the ResNet model is a crop leaf image of rgb type. Image is converted to tensor and then passed to the model using ImageFolder function.



```
def predict_image(img, model):
    """Converts image to array and return the predicted class
       with highest probability"""
    # Convert to a batch of 1
    xb = to_device(img.unsqueeze(0), device)
    # Get predictions from model
    yb = model(xb)
    # Pick index with highest probability
    _, preds  = torch.max(yb, dim=1)
    # Retrieve the class label

    return train.classes[preds[0].item()]
```

```
# predicting first image
img, label = test[0]
plt.imshow(img.permute(1, 2, 0))
print('Label:', test_images[0], ', Predicted:', predict_image(img, model))
```

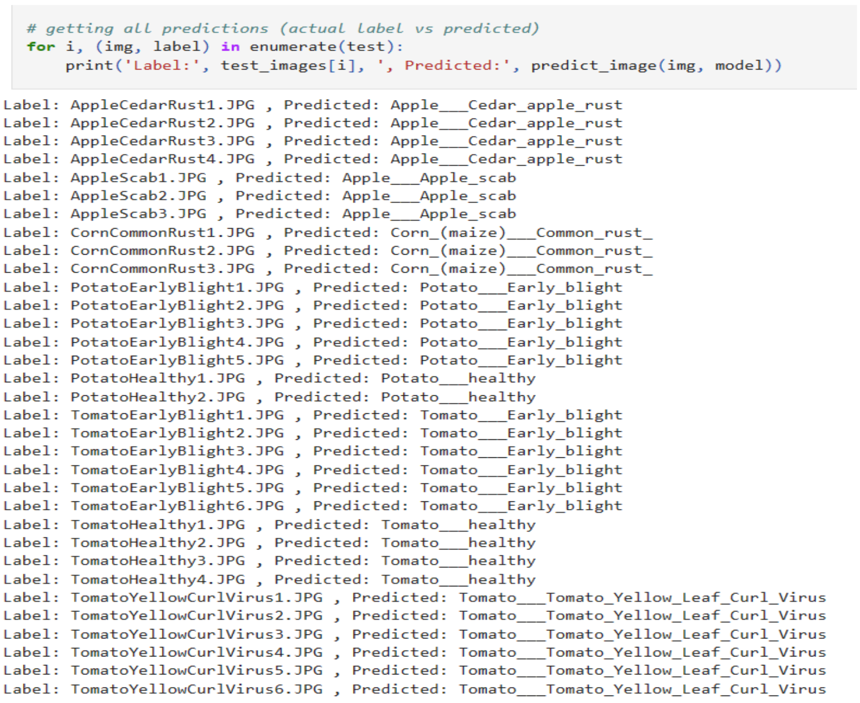
```
.abel: AppleCedarRust1.JPG , Predicted: Apple___Cedar_apple_rust
```

Fig. 5. Input Image

Input given to a random forest classifier model is an array of values that depict nitrogen, potassium, phosphorus, temperature, humidity, rainfall values.

5.2 Output:

The output for the ResNet model is label of the predicted class.

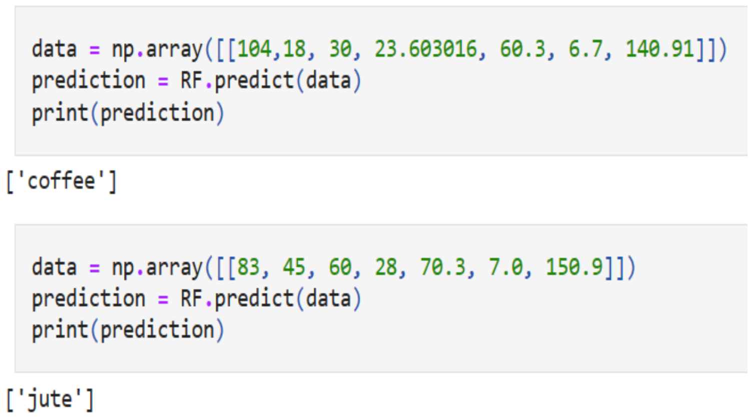


```
# getting all predictions (actual label vs predicted)
for i, (img, label) in enumerate(test):
    print('Label:', test_images[i], ', Predicted:', predict_image(img, model))
```

```
Label: AppleCedarRust1.JPG , Predicted: Apple___Cedar_apple_rust
Label: AppleCedarRust2.JPG , Predicted: Apple___Cedar_apple_rust
Label: AppleCedarRust3.JPG , Predicted: Apple___Cedar_apple_rust
Label: AppleCedarRust4.JPG , Predicted: Apple___Cedar_apple_rust
Label: AppleScab1.JPG , Predicted: Apple___Apple_scab
Label: AppleScab2.JPG , Predicted: Apple___Apple_scab
Label: AppleScab3.JPG , Predicted: Apple___Apple_scab
Label: CornCommonRust1.JPG , Predicted: Corn_(maize)___Common_rust_
Label: CornCommonRust2.JPG , Predicted: Corn_(maize)___Common_rust_
Label: CornCommonRust3.JPG , Predicted: Corn_(maize)___Common_rust_
Label: PotatoEarlyBlight1.JPG , Predicted: Potato___Early_blight
Label: PotatoEarlyBlight2.JPG , Predicted: Potato___Early_blight
Label: PotatoEarlyBlight3.JPG , Predicted: Potato___Early_blight
Label: PotatoEarlyBlight4.JPG , Predicted: Potato___Early_blight
Label: PotatoEarlyBlight5.JPG , Predicted: Potato___Early_blight
Label: PotatoHealthy1.JPG , Predicted: Potato___healthy
Label: PotatoHealthy2.JPG , Predicted: Potato___healthy
Label: TomatoEarlyBlight1.JPG , Predicted: Tomato___Early_blight
Label: TomatoEarlyBlight2.JPG , Predicted: Tomato___Early_blight
Label: TomatoEarlyBlight3.JPG , Predicted: Tomato___Early_blight
Label: TomatoEarlyBlight4.JPG , Predicted: Tomato___Early_blight
Label: TomatoEarlyBlight5.JPG , Predicted: Tomato___Early_blight
Label: TomatoEarlyBlight6.JPG , Predicted: Tomato___Early_blight
Label: TomatoHealthy1.JPG , Predicted: Tomato___healthy
Label: TomatoHealthy2.JPG , Predicted: Tomato___healthy
Label: TomatoHealthy3.JPG , Predicted: Tomato___healthy
Label: TomatoHealthy4.JPG , Predicted: Tomato___healthy
Label: TomatoYellowCurlVirus1.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Label: TomatoYellowCurlVirus2.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Label: TomatoYellowCurlVirus3.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Label: TomatoYellowCurlVirus4.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Label: TomatoYellowCurlVirus5.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
Label: TomatoYellowCurlVirus6.JPG , Predicted: Tomato___Tomato_Yellow_Leaf_Curl_Virus
```

Fig. 6. Resnet Model Output

The output for random forest classifier is a crop recommendation.



```
data = np.array([[104,18, 30, 23.603016, 60.3, 6.7, 140.91]])
prediction = RF.predict(data)
print(prediction)
```

```
['coffee']
```

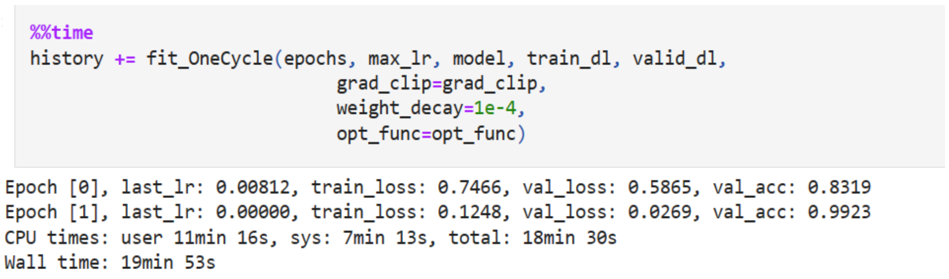
```
data = np.array([[83, 45, 60, 28, 70.3, 7.0, 150.9]])
prediction = RF.predict(data)
print(prediction)
```

```
['jute']
```

Fig. 7. Random Forest Classifier Model Output

5.3 Analysis:

The Resnet algorithm used for the crop disease detection and pesticide recommendation model gave 99.2% validation accuracy. The result is significant as the dataset used comprised 87000 images. The Renet algorithm also overcomes the prominent gradient descent problem that was observed in other popular deep learning algorithms. The Random Forest classifier algorithm gave 100% test accuracy given the fact that the dataset used for training was less and usually it works well for classification and regression problems. It performed pretty well when compared with other algorithms on the same dataset as shown in the figure



```
%%time
history += fit_OneCycle(epochs, max_lr, model, train_dl, valid_dl,
                             grad_clip=grad_clip,
                             weight_decay=1e-4,
                             opt_func=opt_func)

Epoch [0], last_lr: 0.00812, train_loss: 0.7466, val_loss: 0.5865, val_acc: 0.8319
Epoch [1], last_lr: 0.00000, train_loss: 0.1248, val_loss: 0.0269, val_acc: 0.9923
CPU times: user 11min 16s, sys: 7min 13s, total: 18min 30s
Wall time: 19min 53s
```

Fig. 8. Resnet algorithm performance metric

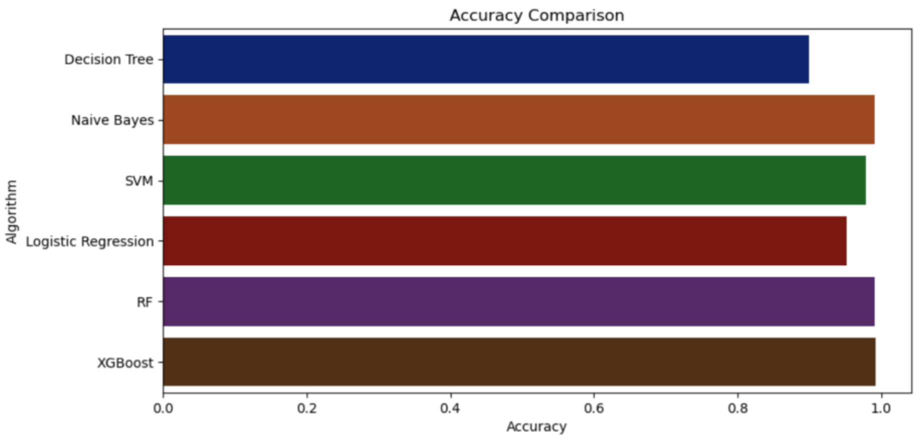


Fig. 9. Random Forest(RF) algorithm vs other algorithms

6 Conclusion

Easy Agriculture: Crops' disease detection, pesticide/fertilizer and crop recommendations is an agri-tech website that solves few of the most important problem statements in agriculture. Crop disease feature requires a crop leaf image as an input and the residual neural network model in the backend diagnoses the condition of the crop and gives results. Crop recommendation feature based upon the soil parameters; environment conditions given as input recommends the suitable crop to be cultivated. Fertilizer recommendation too takes in necessary soil parameters and crop being cultivated and gives the fertilizer recommendation. By going through a crop disease description farmers can decide the severity of the disease and what all can be done. No. of crops taken into consideration are less when compared to no. of crops that are cultivated across the world. This software increases the ease of farming and also the probability of finding the disease of a crop in the initial stage itself. Also, it provides a right medium for farming and the agricultural community to know about agricultural products like pesticides and fertilizers and share the awareness among themselves about the latest agricultural innovations. This is a deployed project, so it increases the ease of use in any device that may be a computer or a smartphone.

Easy Agriculture website can be extended into a mobile app so that in real time image capturing facility without storage is achieved. Also, image modules need to be developed so as to take images of any resolution and type. This project can be integrated with IoT projects that can take soil samples- calculate parameters and capture real time crop leaf images for achieving agriculture process automation. Dataset used here comprises 38 classes,14 plants and 26 diseases which can be extended into even bigger datasets. More authentic and reliable datasets can be used for training the models so that efficiency in real time usage is high. . Even more features and modules can be integrated into the project so that many other agricultural problems statements can be solved via just one integrated solution.

References

1. S. Shrimali, *PlantifyAI: A Novel Convolutional Neural Network Based Mobile Application for Efficient Crop Disease Detection and Treatment*, 2021 2nd Asia Conference on Computers and Communications (ACCC), Singapore, pp. 6-9 (2021)
2. L. Shanmugam, A. L. A. Adline, N. Aishwarya and G. Krithika, *Disease detection in crops using remote sensing images*, 2017 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai, India, pp. 112-115 (2017)
3. A. Morbekar, A. Parihar and R. Jadhav, *Crop Disease Detection Using YOLO*,2020 International Conference for Emerging Technology (INCET), Belgaum, India, pp. 1-5 (2020)
4. O. Kulkarni, *Crop Disease Detection Using Deep Learning*, 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, pp. 1-4 (2018)
5. S. Solanke, P. Mehare, S. Shinde, V. Ingle and S. Zope, *IoT Based Crop Disease Detection and Pesting for Greenhouse - A Review*, 2018 3rd International Conference for Convergence in Technology (I2CT), Pune, India, pp. 1-4 (2018)
6. K. He, X. Zhang, S. Ren and J. Sun, *Deep Residual Learning for Image Recognition*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778 (2016)
7. S Sedkaoui, M Khelfaoui, *Classification Algorithms*, in Sharing Economy and Big Data Analytics ,Wiley, pp.171-194 (2020)