CROPCARE: An Intelligent Real-Time Sustainable IoT System for Crop Disease Detection Using Mobile Vision

Garima Garg, Shivam Gupta, Preeti Mishra, *Member, IEEE*, Ankit Vidyarthi, Aman Singh, and Asmaa Ali

***Abstract*—Agriculture is an important sector that plays an essential role in the economic development of a country. Each year farmers face numerous challenges in producing good quality crops. One of the major reasons behind the failure of the harvest is the use of unscientific agricultural practices. Moreover, every year enormous crop loss is encountered either by pests, specific diseases, or natural disasters. It raises a strong concern to employ sustainable advanced technologies to address agriculture-related issues. In this article, a sustainable real-time crop disease detection and prevention system, called CROPCARE, is proposed. The system integrates mobile vision, Internet of Things (IoT), and Google Cloud services for sustainable growth of crops. The primary function of the proposed intelligent system is to detect crop diseases through the CROPCARE—mobile application. It uses the superresolution convolution network (SRCNN) and the pretrained model MobileNet-V2 to generate a decision model trained over various diseases. To maintain sustainability, the mobile app is integrated with IoT sensors and Google Cloud services. The proposed system also provides recommendations that help farmers know about current soil conditions, weather conditions, disease prevention methods, etc. It supports both Hindi and English dictionaries for the convenience of the farmers. The proposed approach is validated by using the PlantVillage data set. The obtained results confirm the performance strength of the proposed system.**

***Index Terms*—Crop analysis, Internet of Agriculture Things, recommendation system, smart farming, sustainable computing.**

I. Introduction

AGRICULTURE forms a vital part of every country's economy. Every year farmers struggle to grow a variety of crops to a considerable population. Plant diseases severely affect the production and quality of food, fiber, and biofuel crops. The losses might be disastrous or incessant. The normal record for 42% of the creation of the six most significant food crops [1] generally degrades every year. Farmers burn huge resources on disease management and often without proper technical support. It results in poor disease control and can cause harmful results. A report published by a UC Agriculture and Natural Resources with other members of the International Society for Plant Pathology states that the crop yield is degrading because of pathogens and pests.[1]

On the contrary, manual work requires more time for processing and agronomists to detect the diseases and their cause [3]. Meanwhile, farmers are also battling with outrageous climate conditions in regions around the globe. Based on the Indian study [5], it was found that yields worth Rs. 50 000 crore were lost significantly due to pest and disease attacks every year as pesticide utilization is low in India.

There were several existing solutions [4]–[6] given for handling such kinds of problems. Most of the existing approaches are merely based on machine learning analysis, which fails to solve the real-time problems faced by the farmers more realistically. Some of the existing work [10] is limited to the use of traditional classification algorithm. Some researchers [16], [19] have adopted the use of advanced machine learning and deep learning algorithms. However, they do not provide any recommendation system for the detection of crop diseases. The existing mobile-based approaches [7] are not integrated with Internet-of-Things (IoT) sensors. Hence, such approaches cannot detect the weather conditions, such as temperature, humidity, soil moisture, pH value, chemical, and biological parameters. The IoT sensors can help in knowing more about soil and appropriate crop type for monitored fields. Merely using IoT sensors [29] may not be sufficient for crop disease detection (CDD) and sustainable growth of crops. The existing systems [19] are not flexible in usage. The farmers face issues interacting with the application as they do not have a user-friendly interface. Moreover, the technological interface of the existing applications increases the complexity of the system. The farmers face difficulty in clicking many buttons to reach the required feature.

This article proposed a sustainable real-time smart CDD system, called *CROPCARE*, which detects various diseases in crops using mobile vision, IoT, and cloud computing technologies. Toward smart farming, the proposed system also facilitates the recommendation functionality to suggest

Manuscript received 1 May 2021; revised 25 June 2021; accepted 26 August 2021. Date of publication 1 September 2021; date of current version 6 February 2023. *(Corresponding authors: Ankit Vidyarthi; Preeti Mishra.)*

Garima Garg and Shivam Gupta are with the Department of CSE, Graphic Era Deemed to be University, Dehradun 248002, India (e-mail: ggarima163@gmail.com; shivamguptasg1808@gmail.com).

Preeti Mishra is with Department of CS, Doon University, Dehradun 248001, India (e-mail: scholar.preeti@gmail.com).

Ankit Vidyarthi is with the Department of CSE&IT, Jaypee Institute of Information Technology, Noida 201309, India (e-mail: dr.ankit.vidyarthi@gmail.com).

Aman Singh is with the Department of CSE, Lovely Professional University, Phagwara 144411, India, and also with the Uttaranchal Institute of Technology, Uttaranchal University, Dehradun 248007, India (e-mail: amansingh.x@gmail.com).

Asmaa Ali is with the School of Computing, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: ali@cs.queensu.ca).

Digital Object Identifier 10.1109/JIOT.2021.3109019

[1]https://ucanr.edu/blogs/blogcore/postdetail.cfm?postnum=29354

appropriate pesticides and prevention methods on the detection of specific crop disease, current weather conditions, appropriate soil parameters, etc. The proposed system components are divided into two parts: 1) front-end component and 2) back-end component. The front-end component uses mobile resources for execution and is part of the *CROPCARE*-mobile application. The mobile application provides an easy to use user interface (supporting both Hindi and English interface) and is operated by farmers. The farmers can easily control IoT-sensors, capture the crop images to detect the type of diseases in real time, and invoke recommendation modules toward smart farming. On the other hand, to maintain sustainability, the proposed system uses IoT sensors, recommendation modules, and decision model [which uses a superresolution convolution network (SRCNN)] as part of back-end components using the Google Cloud platform for their execution. The IoT-sensor nodes are configured to store the real-time sensor values of the monitoring field in the Google Cloud Platform. The back-end components are invoked automatically based on the input received from the user interface. In the traditional approaches [17], [19], [21], the authors limit their work to only a single crop, i.e., a small data set is used for training and validating the model. Some existing prototypes need Internet access to perform operations and cannot generate the proper report regarding the crop. In many existing works, the author does not make the proper use of the latest technologies, such as cloud, real-time database, etc., through which the data can be used in future analysis. All such issues were taken into consideration in the proposed work that differentiates it from existing solutions. The key contributions of the proposed work are as follows.

1) Propose a sustainable real-time smart CDD system by integrating mobile mobile vision with IoT.
2) Develop an Android application supporting an easy to use interface that can be easily operated by farmers. The interface is automatically connected to Google Cloud and IoT sensors nodes for smart farming.
3) Test the leaf diseases using the proposed MobileNet-V2 Deep architecture, and the model was trained on the benchmark PlantVillage data set.
4) Maintain sustainability and smart farming. The proposed system uses the real-time analysis, prediction, and recommendation modules on the mobile app using Google API.
5) The proposed system performance was validated with a publicly available data set, and do the comparative analysis with the existing literature.

The remainder of this article is organized as follows. Section II gives the highlights of the related work toward the problem domain. The design architecture of the proposed *CROPCARE* is explained in detail under Section III. The performance analysis is presented in Section IV followed by the conclusion and future work in Section V.

II. Related Work

In the literature, there exists several research works [2], [20] that are based on machine learning and deep learning models for solving agriculture-domain problems. A deep model was proposed in [12] using the cycle-consistent adversarial network (CycleGAN) to distinguish the plant diseases into four classes. That model has been trained by using a data augmentation (DA) technique with densely connected layers to optimize feature layers. Based on their experimentation, it was found that the model was able to detect anthracnose lesions that were present on apple surfaces in orchards with an accuracy of only 90.40% whereas CROPCARE is able to detect the diseases in 14 crops and attains the accuracy of 96.17% with a proper recommendation system.

Another similar type of work related to disease detection on tomato crops is presented in [8] and [10]. The adopted methodology was more toward machine learning using a small data set of 138 images. The authors presented their results into multiple phases, i.e., initially, various different machine learning algorithms were used, such as SVM with linear kernel and quadratic kernel (QK), radial basis function (RBF), multilayer perceptron (MLP), and polynomial kernel on RGB images. The motivation of that work was to detect leaf curl disease (TYLCD) on tomato leaves only. They got the accuracy of 90%. Then, the author used the SVM algorithm with thermal and stereo visible light to detect the tomato powdery mildew fungus, namely, Oidium neolycopersici.

The similar work [9], [11] is based on SVM and other models, such as decision tree (DT), random forest (RF), and Naive Bayes (NB), for the detection of potato and corn diseases, respectively. In [15] developed an automated system to detect the diseases in leaf using sensors for temperature, humidity, color variations, and compared their accuracy. They are able to attain 88% (highest) by comparing the leaves based on their color variations. They performed their work on 100 sample leafs only in which 50 are having diseases and rest are healthy.

Based on machine learning algorithms, there exist several works that were used either for segmentation or classification of crop leaves [27], [28]. Guan *et al.* [13] proposed a method to discriminate between pairs of diseases in wheat and grapevines for which the images are segmented by a K-means algorithm, and features are extracted using the shape and texture. While Hu *et al.* [14] have used the hyperspectral imaging technique for detecting diseases in potatoes and attain the accuracy of 95%. Another work [16] provided a smart application on the agriculture sector, which is evaluated only on grape diseases with accuracy of 90%.

However, there were several literatures that have used deep learning in agriculture sector for the classification of leaf diseases [17], [18]. Their proposed approaches use a convolution neural network and mainly focused on coffee leaves and attains the accuracy of 97.07% with ResNet-50 using images of symptom spot. The authors also provide a framework to detect the diseases but that framework needs Internet access to perform the operations. Another work based on deep modeling was presented in [17] that explored the use of CNN for detecting the diseases in rice and used the data set of 500 images only for training and validating the model. The adopted methodology was based on transfer learning using AlexNet to build a rice diseases classier. The overall accuracy of the deep model was 95.48% using a tenfold cross validation, which was found better as compared with SVM—91% and standard back propagation—92%, and particle swarm optimization (PSO) achieved 88%.

Singh and Misra [19] proposed an automatic technique that was used for detecting little leaf disease found in pine tree using image segmentation and soft computing techniques. From these techniques, they are able to detect the symptoms of little leaf diseases. Several other works for the identification of the leaf diseases were reported in [20], [24], and [26] using deep learning and only able to identify spots and lesions on the leafs.

Thakur and Mittal [29] proposed a real-time classification of crop diseases using machine learning and IoT applications in a cloud environment in which they used unsupervised machine learning. In this system, a cloud environment is used to store large amounts of data, but the chances of increasing the accurate result degrade due to the use of clustering algorithms. On the other hand, Nguyen *et al.* [30] tried to include the important factors, which are reasons for the production of crops. It comprises all major sensors and implements IoT technology for seed recognition with image processing.

Some author [32] proposed their work by identifying 26 diseases in crops using plant village in which they used deep learning metaarchitectures, such as Single Shot MultiBox Detector, Faster Region-based Convolutional Neural Network, and Region-based Fully Convolutional Networks, and applied them using the TensorFlow object detection framework.

Some current work [34] detected the disease in Agrobot using *K*-means and neural network to identify the diseases in plant leaf. Yoganand *et al.* [35] used sensors, such as DHT-11, Soil moisture sensor, GSM, webcam, and controllers to receive the data from groundnut farms, and analyzed them using the XG-boost machine learning model. After analyzing, the result is sent to farmers' mail, which is one drawback as most farmers are not aware of mails.

The traditional approaches discussed above mainly focus on one crop, i.e., small data set is used to train and validate the model. The prototype discussed above needs Internet access to perform operations and is not able to generate the proper report regarding the crop. Even the information is not stored in cloud for future use.

In the literature, there were several works that focus on smart farming in the agriculture sector. These works use the latest technologies, such as IoT, sensors, and machine learning for increasing the productivity. Based on this, Roy *et al.* [36] provided solutions for better crop productivity using dynamic irrigation scheduling system for efficient water management. Similarly, Bhattacharya *et al.* [37] provided the solution for onboard and autonomous path planning using UAV-based aerial IoT platform. Besides these works, there exists several works that use smart farming using IoT environment [38], [39], which focuses on energy consumption and resource constraints. These works gave rise to the new era of farming in the agriculture domain.

III. Proposed CROPCARE Architecture

The section provides a detailed description of CROPCARE architecture along with the description of detection

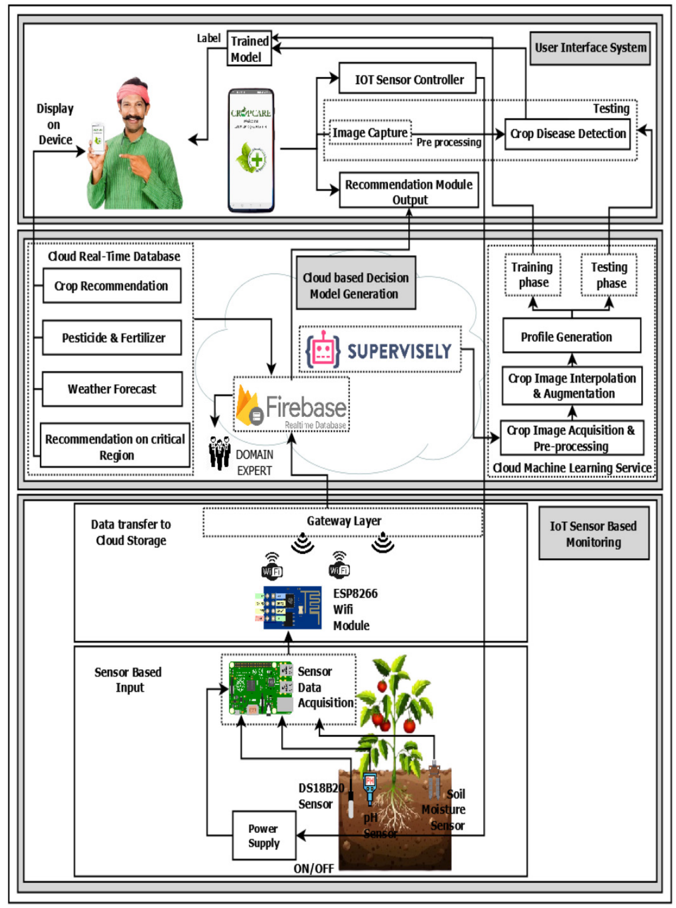

Fig. 1. CROPCARE design architecture.

components. The CROPCARE architecture is divided into three key components, as shown in Fig. 1: 1) IoT sensor-based monitoring (ISM); 2) cloud-based decision model generation (CDMG); and 3) user interaction system–mobile vision (UIS). UIS helps in detecting real-time crop diseases. It implements both mobile vision and deep learning technology. Algorithm for backend and frontend has been discussed in Algorithm 1 and Algorithm 2, respectively. The timely detection of crop disease helps the farmers to take preventive measures and increase the productivity of the crops. Each component is further classified into subcomponents, which are discussed as follows.

A. IoT Sensor-Based Monitoring System

An IoT system is built for the purpose of receiving baseline parameters of the soil. This system consists of sensors that communicate to the cloud through some kind of connectivity. Once the data get to the cloud, software processes it or stores it on the cloud database. These data can be further retrieved when requested by the user. This component is divided into two subcomponents, i.e., sensor-based input (SBI) and data transfer to cloud storage (DCS).

1) Sensor-Based Input: The circuit diagram of the control board of an IoT sensor node is illustrated in Fig. 2. Here, Raspberry Pi model 3 B+ is used as a microcontroller unit. It has a 64-bit quad-core processor running at 1.4 GHz with

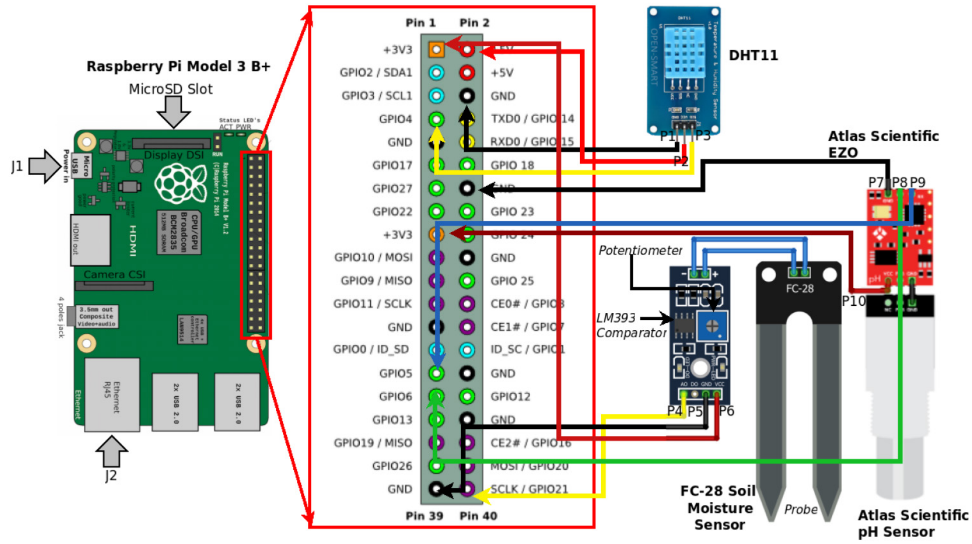

Fig. 2. IoT sensor node configurations.

built-in metal heatsink, dual-band 2.4 GHz, and 5-GHz wireless LAN, faster (300 Mb/s) Ethernet, and PoE capability via a separate PoE HAT. MicroSD card is used in Raspberry Pi to provide storage for computation. The supply voltage of Raspberry Pi is provided by a Li-Ion y 1000-mAh Battery through a micro USB cable at port J1. In addition to this, three sensors (DS18B20 Sensor, FC-Soil Moisture Sensor, and Atlas Scientific pH sensor) are connected to the 40 Pins Connector of the Raspberry Pi. The description of some key sensors is given as follows.

1) The DS18B20 digital thermometer provides 9–12-bit Celsius temperature measurements. Pin P1 of DS18B20 is connected to the GND Pin 6 of Raspberry Pi, pin P2 is connected to +5V Pin 2, and pin P3 is connected to GPIO4 for data transfer. The sensor communicates over a one-wire bus and requires little in the way of additional components.
2) The FC-28 Soil Moisture sensor is used to gauge the volumetric content of water within the soil. This module also includes a Potentiometer that will fix the threshold value. This value can be evaluated by the LM393-Comparator. This module contains 4 Pins: Pin P4 is the analog or digital output (depending upon requirement) of the FC-28 Sensor, which is connected to GPIO21 of Raspberry Pi, Pin P5 is connected to the GND Pin 39, and pin P6 is connected to +3.3V Pin 2.
3) The analog pH sensor is specifically designed to measure the pH of the solution and reflect the acidity or alkalinity. Pin P7 of the pH sensor is connected to the GND Pin 14 of Raspberry Pi, Pin P8, and P9 of the EZO circuit is connected to Raspberry Pi's I2C pins SDA (GPIO 6) and SCL (GPIO 7), respectively. Pin P10 is connected to the +3.3V Pin 17.

Various IoT sensor nodes with above-described configurations are deployed in the farming fields whose conditions are to be monitored.

2) Data Transfer to Cloud Storage: The IoT sensor nodes that are mentioned in SBI modules are configured with API key of Google Firebase Real-Time Database to collect the real-time values of the sensors from the fields when required. In this module, the data are transferred to the firebase real-time database by using the Wi-Fi module (ESP8266). The modules are configured as a master–slave so that the master always initiates a communication session. Each module will have its own ESP8266. The IoT CPU, i.e., Raspberry Pi, will interact with the ESP8266 via a serial port. The Sensors value will be stored in the firebase real-time database to retrieve the data.

The gateway layer in our proposed system is a network node used in telecommunication that connects two networks (i.e., Firebase Cloud Server and Raspberry Pi LAMP Server) with different transmission protocols together. This layer provides security controls over the data which is transferred through it. All data must travel through or communicate with the gateway before being routed. Hence, gateways serve as an entry and departure point for a network.

B. Cloud-Based Decision Model Generation and Recommendation

CDMG modules thoroughly discuss the usage of cloud services to store several data values and model computation processes. To store data values, CROPCARE used Google Firebase Real-Time Database and for model building, a Web platform named *Supervisely* [29] is used. CDMG module consists of two subcomponents, i.e., cloud real-time database (CRTD) and cloud machine learning service (CMLS).

1) Google Cloud Machine Learning Service: To incorporate machine learning service on the cloud, *Supervisely* provides detailed resources about how to build the Deep model from scratch. It provides a complete environment for performing preprocessing and building our deep learning model. The hardware description to install a new instance of Supervisely on the server can be found on its official website [29]. In the proposed methodology, it is used to link machine learning over a cloud platform.

a) Crop image acquisition and preprocessing: Initially, the Plant Village Data set [22] has been downloaded from the

Algorithm 1: Algorithm of Our Proposed Sustainable CROPCARE Smart System: Backend

```
Part A: Backend Modules of CROPCARE
Function IoT_Sensor_Based_Monitoring() is
    Function Sensor_Based_Input() is
        realTimeSensorValue ← {}
        realTimeSensorValue = IoTSenorNode(SoilMoisture,
          pH, DS18B20 Sensor);
        realTimeSensorValue stored in Rasberry Pi;
        goto Cloud_Storage(realTimeSensorValue);
    end
    Function Cloud_Storage(realTimeSensorValue) is
        Using WifiModule ESP8266;
        Transfer realTimeSensorValue to CloudStorage;
        goto
          Cloud_Real_Time_Database(realTimeSensorValue);
    end
end
Function Cloud_Based_Decision_Model_Generation() is
    Function Cloud_Machine_Learning_Service() is
        PlantVillageDataset = _DownloadFromOfficialSite;
        foreach (image in PlantVillageDataset) do
            Pre-processing(image) {Resize, Normalization,
              RGB to GrayScale};
            do DataAugmentation(image);
        end
        CropImageInterpolation(PlantVillageDataset);
        trainedModel = profileGeneration();
    end
    Function Cloud_Real_Time_Database
      (realTimeSensorValue, DiseaseName) is
        realTimeSensorValue ← {From
          SensorBasedInput};
        centralDatabase ← {};
        if realTimeSensorValue equals best crop condition then
            do CropRecommendation();
        end
        if DiseaseName equals PlantDisease then
            do Pesticide_Fertilizer();
            centralDatabase ← {DiseaseName,
              Coordinates};
        end
        do Weather Forecast();
        report ← {Pesticide_Fertilizer};
        NoofDiseaseinaRegion ← {centralDatabase};
        on analysing CountofDiseaseinRegion do
          Recommendation_on_CriticalRegion();
        return Report;
    end
end
```

Algorithm 2: Algorithm of Our Proposed Sustainable CROPCARE Smart System: Frontend

```
Part B: Frontend Modules of CROPCARE
Function User_Interface_System() is
    Function IoT_Sensor_Controller() is
        Control SensorNode;
        toggle ← {false};
        if toggle is true then
            Get values from sensor & send to
              CloudRealTimeDatabase;
            Fetch from CloudRealTimeDatabase &
              Display;
        end
    end
    Function Crop_Disease_Detection() is
        captureImage ← {Device Camera};
        do Pre-processing();
        trainedModel ← {CloudMLService()};
        diseaseName ←
          {trainedModel(captureImage)};
        Display (CloudRealTimeDatabase(diseaseName));
        goto RecommendationModuleOutput();
    end
    Function Recommendation_Module_Output() is
        provide access to CloudRealTimeDatabase
          Features {Crop Recommendation, Pesticide &
          Fertilizer, Weather Forecast, Recommenadation
          on critical regions};
    end
end
```

official site and uploaded the entire data set on Supervisely for further processes. Supervisely supports data transformation language (DTL), which allows manipulating our data set. Then, preprocessing is done on the raw data obtained from online sources to make it suitable for building and training deep learning models. All the images are resized into the same dimensions so that processing can be easily done. Normalization is applied to every image to change the pixel intensity values so that they can be fitted on a common scale. Furthermore, each image is converted into the gray scale format from RGB format to reduce computation. Background removal is done to remove all the unnecessary features using the edge detection process. This process makes each image to look more clean and sharp with better identification capability of shadows in the frame.

b) Crop image interpolation and augmentation: Image interpolation techniques help us to detect the low features present in the images, increasing the accuracy of the model. In this model, the Bicubic interpolation technique is used, shown in Fig. 3.[2] After the Bicubic, the SRCNN model will be used for converting low-resolution images to high-resolution images. SRCNN is a convolutional neural network that consists of patch extraction and representation, nonlinear mapping, and reconstruction. DA helps in increasing the model's performance by introducing rescaling, rotation, horizontal flip, and padding to increase the data set size. Rescaling helps to increase the visual appearance, Rotation, and Horizontal Flip helps to collect the data at every angle so that model can predict the image in every condition. Padding will provide the boundary to every image. After preprocessing and DA, the data set has been passed to the PG module.

c) Superresolution convolutional neural network: SRCNN is a deep convolutional neural network that helps to improve the quality of images as it learns the end-to-end mapping and then converts them into high-resolution images for which Keras library is used. In end-to-end mapping, SRCNN

[2]Rafael C. GONZALES, and Richard E. Woods, "Digital Image Processing," 3rd edition, chapter 3, Person publication, 2007.

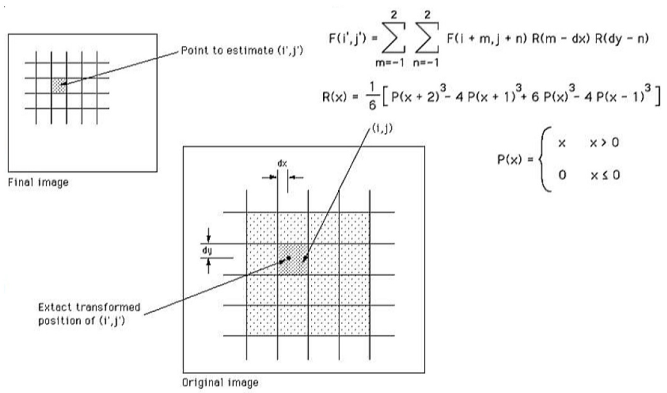

Fig. 3. Mathematical elaboration of Bicubic interpolation.

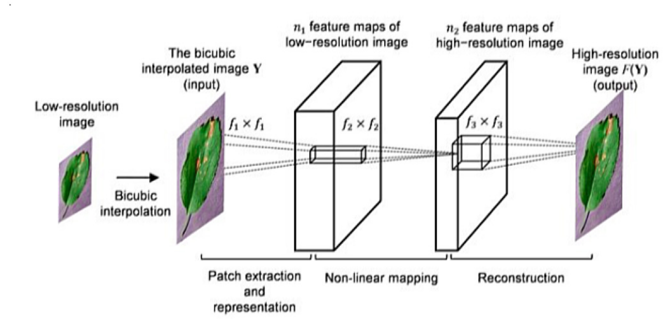

Fig. 4. Learning with deep convolution network for image superresolution.

TABLE I
MobileNet V2 Model Description

Parameters	**Values**
No. of Layers	53
No. of Parameters	2.9 M
No. of Epochs	30
Learning Rate	0.001
Dropout Rate	0.2
Input Shape	(224, 224, 3)
Optimizer	Adam
Batch Size	64

mainly uses three image quality matrices, namely, peak signal-to-noise ratio, mean-squared error, and the structural similarity index [34].

d) Profile generation: PG module describes the process of training a model and testing that model on real-time images. PG module receives the input from the CIA module and trains the proposed modified MobileNet-V2 model on that data set input. The details of the model's parameters are given in Table I. The detailed architecture of SRCNN and MobileNet V2 are shown in Fig. 4 and Fig. 5, respectively. The entire process is divided into two phases before deploying, i.e., Training/Learning and Testing phase. The following is the detailed description of these two phases.

1) *Training Phase:* The MobileNet-v2 model architecture training was done using the state-of-art Plant Village data set. In the MobileNet-V2 model, Depthwise Separable Convolution is introduced, which dramatically reduces the complexity cost and model size of the network that is suitable to mobile devices, or any devices with low computational power. The input size of the model by default is 224 $\times$ 224. The concept of transfer learning is used to train our data set over some layers of the MobileNet-V2 Model for better accuracy. After training, the model gives the validation accuracy of 96.72%. Finally, the model was converted to *tflite* using TensorFlow Lite API to reduce model's size. Later, the *tflite* model was embedded in the mobile app and deployed on the Android device.
2) *Testing Phase:* To test the model image is being captured from the android device, and then this image will be passed to the SRCNN model so that the captured should be converted to high resolution from low resolution. Later, the high-resolution image was passed to the model embedded in the corresponding app, which provides a result in the form of a label.

2) Google Cloud Real-Time Database: It explains the use of firebase real-time cloud-hosted database. The proposed methodology uses firebase security rules to secure our data in firebase real-time database. Firebase security rules check a pattern against database paths and then enforce custom conditions to grant access to the data at those paths. Firebase real-time database rules supports JSON in rule definitions and it is synchronized in real time to every connected client. It stores different forms of data, such as numeric data values, images, text values, etc. Installation and setup instructions can be found on the official site of Google Firebase. The data values, which are store in cloud database, are as follows.

a) Soil-based crop recommendation system: SCRS stores detailed information of soil type, soil moisture value, temperature, pH, etc., required for the best yield of particular crops on the firebase database. After receiving all the information on the soil from the ISM module, the SCRS module will recommend the best suitable crop for that field. We require to store these values in the firebase database because if the user requested again in a short span, it will not again trigger the ISM module for the same, instead, it will fetch the value from the database. This will reduce computation.

b) Pesticide and fertilizer recommendation: PFR facilitates the farmer by providing information on the pesticides, which could be required if any pest attack is monitored. Depending upon the nutrient values present in the soil, PFR suggests the quantity of fertilizer required for the field. PFR extracts information from the cloud database and displays it on the android device. PFR also sends the location of the farmer (field) to the database (cloud) so that analyses on pest control can be performed for future prevention from pests. PFR also has a feature of voice recognition and provides a reply in the form of a voice message. This voice message includes a description of the pesticide and fertilizer.

c) Weather forecast: WF subcomponent truly facilitates farmers on many fronts, including supporting them to make knowledgeable choices to get the best from their investments and hard work. WF involves all weather factors, which right away affect farm planning or operations. The losses in crop production can be decreased by the way of adopting proper

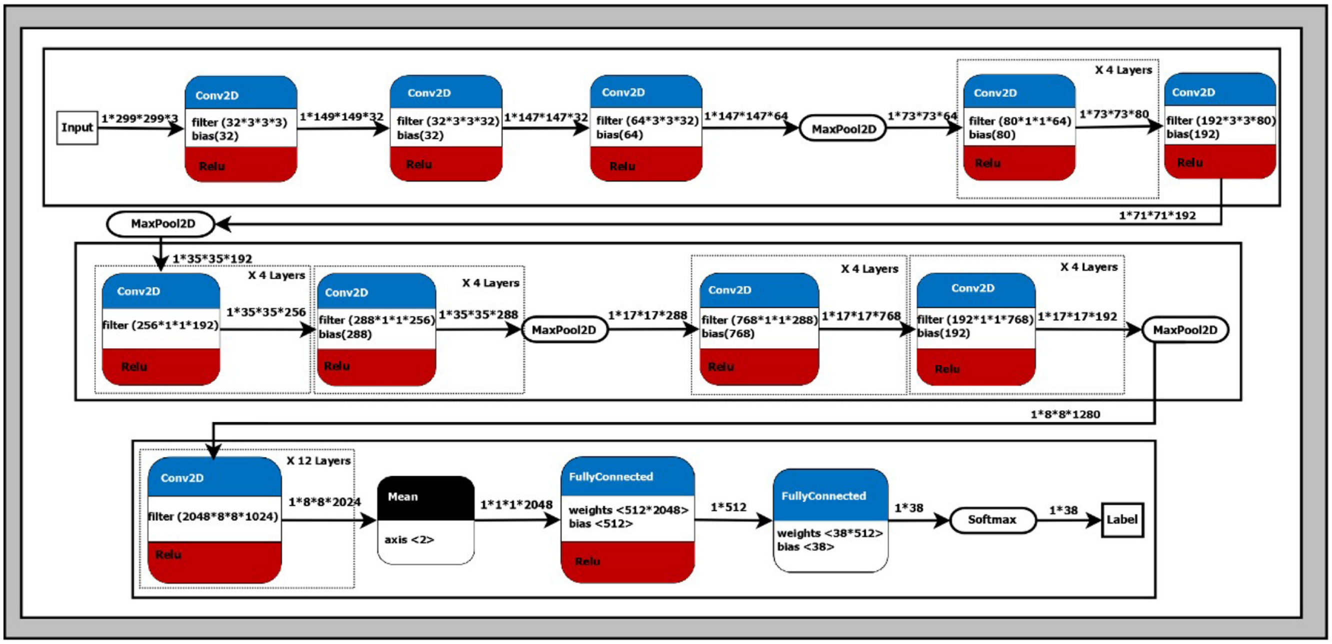

Fig. 5. MobileNet V2 architecture.

crop management practices with the assist of timely and correct weather forecasts. With the weather prediction, farmers can assist their plan for planting and harvesting of their crops. WF also has a feature of voice recognition and provides a reply in the form of a voice message. This voice message includes a description of the chances of rainfall in the upcoming days.

d) Recommendation on critical regions of focus: RCRF will send the information about the critical regions to the database. In critical regions, such as providing help in controlling pests in particular areas, providing fertilizer to the farmer, providing suggestions on certain fields that are not fulfilled by *CROPCARE*, etc., RCRF provides a feature of capturing the image of the problem, and it also provides voice typing so that illiterate farmers can easily handle it. This information will be sent to the central database, from their Ministry of Agriculture or Agronomist will look after farmer queries.

Cloud database service is also used to keep all relevant information about various crops such as baseline soil parameter values, such as temperature, soil moisture, soil type, pH value, etc., required for good productivity of specific crops. This information has been collected from various authentic sources. Furthermore, by comparing the baseline soil parameters with real-time soil parameters (collected using IoT sensors as described above), other important decisions about crops can be taken.

C. User Interaction System—Mobile Vision

UIS module describes the concept of Mobile Vision, which is integrated with deep learning. The Mobile Vision approach can be achieved by deploying an app. This approach helps the user for real-time detection of objects (here, plant diseases) on Android devices. In the proposed system, the concept of Android Localization (multilanguage) of Strings is used to handle language issues. Generally, android considers English a default language, and it loads the string resources from /res/values/strings.xml. If the requirement is to incorporate some other supportive languages, then we need to create a values folder by appending the Hyphen and ISO language code. Android OS will check for the appropriate language resources available in the app. If the app supports a selected language, then android will look for the string resources in the project's values-(ISO language code) folder. All these modules are incorporated over a UIS. This component is further divided into subcomponents, which are IoT sensor controller (ISC), CDD, and recommendation module output (RMO).

1) IoT Sensor Controller: ISC module provides a facility to trigger the ISM module so that the user can get the baseline parameter of soil, such as pH value, temperature, soil moisture, etc. These values will be displayed on the app. The ISC module prevents the unnecessary computation power used in collecting the sensor's data at regular intervals and uploading it to the cloud database.

2) Crop Disease Detection: This module provides the feature for the real-time detection of the disease from the crop using the android phone. The user will capture the image, and the same will pass on to preprocessing like resizing the image (255 $\times$ 255) and scaling. After that, the image is passed to the decision model for further classification and the prediction of the disease. Following the detection of the disease(as discussed earlier), it will familiarize the farmer through displaying the message or through voice message and it will also be reported to the ministry of agriculture so that the government can apply further precautionary actions toward the disease. By informing the detection of disease to the government, the government may have early control of the disease in a particular area, and particular action can be taken.

3) Recommendation Module Output: The RMO module includes information that the user requires after the detection of disease in the crop, such as the description of the disease, the cause of the disease, treatment measures, and prevention measures. This module will also provide the menu for the recommendation for pesticides required to treat that particular disease. This module also contains an information bank that fetches data from a cloud database and acknowledges the farmer regarding updated technologies for using this application, the farmer can operate through the voice messages, and the corresponding module will be invoked based on the interpretation of the command. Farmers can take suggestions from the agronomist by dropping the audio or text message. During the implementation of such projects, the biggest challenge is to make a user-friendly application for the farmers as they are not well educated. *CROPCARE* considered and achieved this challenge by implementing a user-friendly and accessible mobile app user interface.

IV. Performance Analysis

A. Hardware Description

CROPCARE has been performed using a machine having Ubuntu 18.04.3 LTS, Intel Core i5-5200U CPU @ 2.20 GHz 8-GB RAM, 500-GB HDD having 2048-MB Nvidia GeForce 920M GPU. The programs are coded using Python 3.7.3 and having libraries as follows: sys, Keras-2.2.4, TensorFlow-2.0.0, cv2, skimage, Matplotlib-3.0.3, NumPy-1.16.2, tensor-flow hub-0.7.0 to make use of some predefined functions. For better performance and faster results, CROPCARE used a system with GPU.

B. Description of Data Set

CROPCARE has been analyzed through 54 306 images of plant leaves, which have a spread of 38 class labels assigned to them as shown in Table II and their control methods are also given. Each class label is a crop-disease pair and attempted to predict the crop-disease pair given just the image of the plant leaf. The approaches described in this article first resize the images to 224 × 224 pixels for MobileNet_v2 and 299 × 299 pixels for inception v3, and perform both the model optimization and predictions downscaled images.

C. Results

Results for Preprocessing: The prerequisite for training our deep learning model is performing preprocessing and DA. The augmented data set is passed to the SRCNN model so that a low-resolution image will be converted to a high-resolution image as shown in Fig. 6. One of the important factors is the time required for the conversion from low to high resolution. Conversion takes up to 1.1 s, which is practically tested.

Training and Validation Results: The preprocessed data, i.e., high-resolution data set images, were used for the training and validation. As stated in the earlier section, a large data set of images from the PlantVillage data set is used for experimentation. The overall data set was divided into 80–20 form means; 80% of images were used to train the model while the rest 20% images were used for model validation. The experimentation results using the proposed model are presented in Table III. The result suggests that the proposed methodology is better

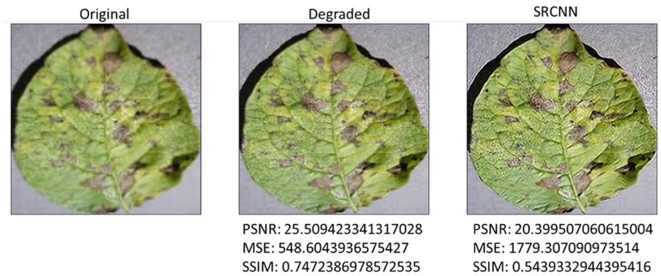

Fig. 6. Output of SRCNN.

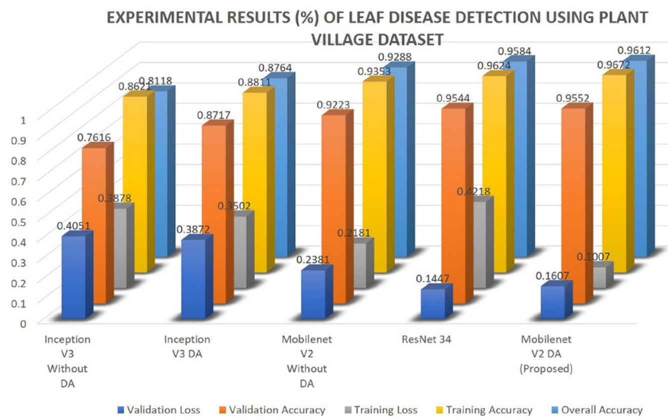

Fig. 7. Result analysis of proposed model with state-of-the-art models.

than other models as experimented with DA and with no DA. The best-achieved values using the proposed methodology is highlighted in Table III in bold (last row). It is being found that the proposed methodology is found better with an overall accuracy of 96.12% and a minimum loss of 10.07%.

The comparative result of the proposed model using DA with other models on Plant village data set is presented in Fig. 7. The analysis of Fig. 7 is given as follows.

1) The increment of val_loss and decrement in val_acc show that the model is cramming values, not learning.
2) The increment in val_loss starts and val_acc results in over fitting or diverse probability values for which the softmax function can be used in the output layer.
3) The decrement in val_loss and increment in val_acc show that the model, i.e., built, is learning and working fine.

The overall computation time of our proposed system is approximately 5 s. Total computation time comprises two tasks that are detection of plant disease and output of the recommendation system. The detection of disease takes 2 s as the model is embedded in the application itself, whereas the output of the recommendation system takes approximately 3 s as the module has to fetch data from a real-time cloud database.

D. Comparison With Existing Work

The experimentation was done using the proposed methodology and on similar experimental conditions, some of the existing methods were implemented on the same data set. The comparative analysis of the work is presented in Table IV. Most publishers compared using different models and compared their accuracy with each other like in [32], where

TABLE II
DISEASE CONTROL METHODS

S.No	Name	How to Control
1.	Apple Scab	Throughout the growing season for apple varieties fungicide sprays are needed healthy fruit.
2.	Apple Black Rot	Tree should be able to receive good air movement and light penetration and dead fruit should be removed from tree.
3.	Apple Cedar Rust	After petals have fallen fungicides can be applied for every 7-10 days to protect new leaves.
4.	Grape Black Rot	Remove all mummified fruit from vines during dormant pruning.
5.	Grape Black Measles (Esca)	Remove the infected berries, leaves and trunk and destroy them.
6.	Grape Leaf Blight	Fungicides sprayed for other diseases in the season may help to reduce this disease.
7.	Peach Bacterial Spot	Compounds available for use on peach and nectarine for bacterial spot include copper, oxytetracycline and syllit+captan; however, repeated applications are typically necessary for even minimal disease control.
8.	Strawberry Leaf Scorch	Keep moisture levels down and avoid long wetness periods by monitoring irrigation schedules
9.	Potato Early Blight	Try to avoid the overhead irrigation.
10.	Potato Late Blight	To prevent tuber infection keep tubers covered with soil throughout the season.
11.	Corn Gray Leaf Spot	Monitor high risk fields closely and consider the risk factors and weather conditions before making fungicide application decisions.
12.	Corn Common Rust	Cost-effective means to manage common rust is resistant varieties.
13.	Corn Northern Leaf Blight	Around the regions humidity should be less.
14.	Orange Huanglongbing(H,B)	Removal of symptomatic trees, protecting grove edges through intensive monitoring, use of pesticides can help in controlling the disease.
15.	Tomato Late Blight	Rotation of crops , like corn, beans and cabbage, for at least three years can provide some control but do not use sceptible plants like pepper, eggplant, potato, sunflower or cosmos in this rotation. Remove and destroy all infected plant material.
16.	Tomato Leaf Mold	Staking and pruning helps in increasing air circulation which decreases the disease rate. Don't water the crop when leaves are wet.

TABLE III
EXPERIMENTAL RESULTS (%) OF LEAF DISEASE DETECTION USING PLANT VILLAGE DATA SET

Architecture	Validation Loss	Validation Accuracy	Training Loss	Training Accuracy	Overall Accuracy
Inception V3 Without DA	40.51	76.16	38.78	86.21	81.18
Inception V3 DA	38.72	87.17	35.02	88.11	87.64
Mobilenet V2 Without DA	23.81	92.23	21.81	93.53	92.88
ResNet 34	14.47	95.44	42.18	96.24	95.84
Mobilenet V2 DA (Proposed)	**16.07**	**95.52**	**10.07**	**96.72**	**96.12**

TABLE IV
COMPARISON OF PROPOSED ARCHITECTURE WITH EXISTING WORK

Parameters	Amara et al. [26]	Vin Gia Nhi et al. [30]	Brahimi et al. [31]	Wang et al.[13]	Proposed Architecture
Dataset	Banana Dataset(PlantVillage)	PlantVillage Dataset	PlantVillage Dataset	Apple Dataset(PlantVillage)	PlantVillage Dataset
No. of Species	1	14	14	1	14
No. of Classes	3	N/A	38	4	38
No. of images	3700	54306	54323	2086	54305
Preprocessing	Re	N/A	Cr, Re	Re, Fl, Ro, Zo	Re, Fl, NR, Sg, BR
CNN Architecture	(Modified)LeNet	N/A	Inception V3	VGG16	MobileNet V2
Transfer Learning	No	Yes	Yes	Yes	Yes
Learning rate	0.001	Yes	0.001	0.001	0.001
Activation Function	Sigmod	N/A	N/A	SGD	Softmax
Batch Size	10	N/A	20	N/A	64
Train - Test Ratio	60% - 40%	N/A	80% - 20%	80% - 20%	80% - 20%
Optimizer	SGD	N/A	N/A	N/A	Adam
No. of iterations	30	N/A	N/A	N/A	30
Overall Accuracy	0.9282	0.92	0.9976	0.904	**0.9672**
IoT sensor	N/A	Soil Moisture Sensor	N/A	N/A	Soil Moisture Sensor,DS18B20 sensor

the authors used pretrained models ResNet-50 and Inception ResNet-V2 [25] but did not provide a recommendation system for the farmers for the timely detection of diseases.

Amara *et al.* [26] proposed their work only on the banana crop and provided an accuracy of 92.82%. Nguyen *et al.* [30] and Brahimi *et al.* [31] proposed their work by using 14 number of species and achieved good accuracy.

E. Discussion About Experimental Results

In experimentation, initially, Inception V3 without the DA model experimented with the lowest training accuracy because of the highest validation loss and the training loss, which affects the overall accuracy. To overcome this, the same model was used but with DA as it helps to train the model from every angle of image, increasing accuracy. Inception V3 with DA will decrease the validation loss and training loss and improves the overall accuracy. Nevertheless, the accuracy was not satisfactory as the Inception V3 model size is 299 $\times$ 299 by default, which occupies a large space.

Later, as accuracy given by Inception was not acceptable, ResNet-34 without DA is used, which improves the overall accuracy and the training accuracy but this model is used for small networks only as it is for two or three layers. As a requirement, the whole model has to be deployed in the Android phone for which there was a need to optimize the machine learning algorithm. Next, to fulfill the requirement,

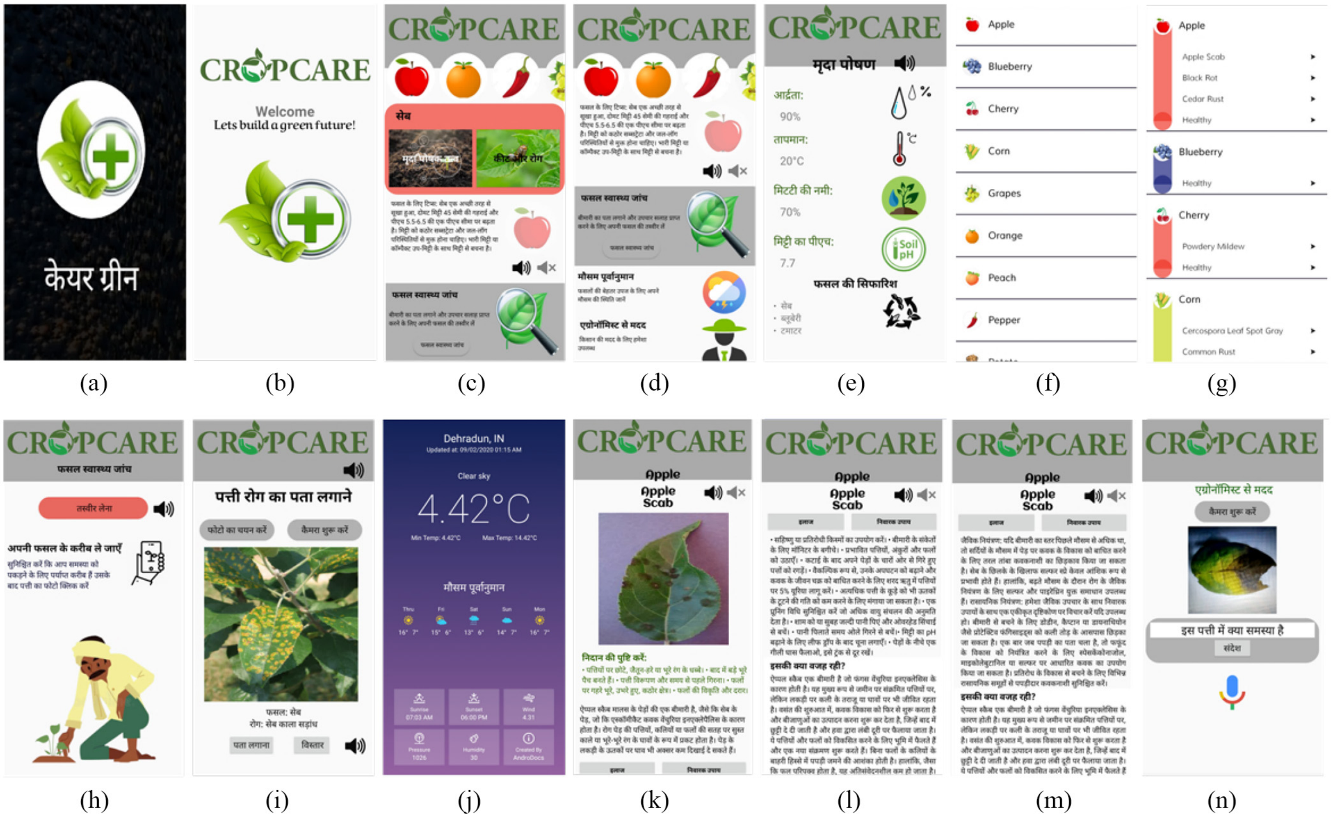

Fig. 8. CROPCARE: Stepwise execution stages. (a) Corp Care. (b) Front Screen. (c) Home Page. (d) Home Page 2. (e) Soli Nutrition. (f) Crop List. (g) Crop Diseases. (h) Health Check. (i) Disease Detection. (j) Weather Forcast. (k) Disease Description. (l) Preventive Measure. (m) Treatment. (n) Help from Agronomist.

the MobileNet V2 model was used, which performed very well on the dynamic data set splitting. We have considered Mobilenet V2 into two variations; first, MobileNet V2 without DA and another with DA.

MobileNet V2 without DA was proposed, which has 224 × 224 default size of the model. Even without the DA process, this model gives the accuracy better than the Inception model as validation accuracy was improved by 7% with less validation loss and less training loss and gave the accuracy 92%, which was not satisfactory.

Finally, Mobilenet V2 with DA was proposed in CROPCARE, i.e., the model was trained with every angle of the image so the accuracy can be improved with the validation loss 0.1607 and the training loss 0.1007, which is the least losses till the proposed models. Even the overall accuracy is 96% at an epoch of 30 only as shown in Fig. 7.

The comparative analysis of the proposed methodology with the existing published work also suggests that the proposed model is better when models working on a multiclass problem with an accuracy of 96.72% compared with other models, as shown in Table IV. Moreover, it was found that the best-gained results were 99.76%, but the authors have not used real-time data analysis using IoT.

F. Recommendation Module

The recommendation module provides several information regarding the disease of the plant. Through this module, the farmer or the client could know about the disease, the cause of the disease, and prevention measures. Our proposed system incorporates the recommendation in two different ways. The first way is the stand-alone way. In this way, the recommendation module plays a role of a knowledge book on plant disease. In this, the farmer can know every detail about any plant disease without using a detection module. The second way is description after disease detection. In this, the farmer will capture the image of a plant leaf, and if any disease is found, then the recommendation module will give a detailed description of that disease, the cause of the disease, and prevention measures.

The recommendation module also recommends about the best suitable crop based on soil nutrition values, which has been calculated using an ISM Module. The complete data for the recommendation module are store in firebase real-time database. Every time user asked for the description, the application fetches the data from the real-time database. The results of the recommendation module were given in the Table IV. The table suggest the methods about how to control the disease based on the different crop type.

G. Mobile Vision-End User Application

Once the model was successfully integrated into the mobile app and installed on an Android device, it was tested with real-time leaf disease prediction by the farmers. The end-user GUI-based overall analysis is given in Fig. 8. The mobile app takes the input image using the inbuilt camera, which was later tested with trained MobileNet v2 lite API and cloud network. The GUI has several features, which help the farmers based on the runtime query provided.

H. Practical Application and Advantages

In the architecture, as explained in Section III, all the data after processing are stored in the cloud database server, i.e., Google Firebase. After evaluating the real-time values, they have been compared in the database itself (as true values for each crop have been taken from Internet sources), which helps to know the region's situation in terms of weather conditions. This database will help the government institute for the proper survey, and they can provide help to farmers by supplying pesticides and fertilizers as per the requirement.

1) Application:

1) CROPCARE will help in increasing business efficiency. By using smart devices, the government can automate production crop processes, and that government bodies will provide fertilizers or pesticides to the farmers.
2) Improve the quality and quantity of the crops. Automation will help provide better quality of crops as the needs of pesticides will be provided on time and early detection of diseases.

2) Advantages:

1) *Data Collection:* Data can be collected and maintained by government bodies so that diseases in crops can be known and prevention measures can be taken. This process will help the farmers to grow better quality crops.
2) *Reduction of Risks:* When farmers can collect the information up to date, they can understand and predict problems that may arise in the future. Moreover, farmers can use this data to improve their business processes.
3) *Business Goes Automated:* Many business processes become automated, which provides many extra features and helps increase efficiency. Thus, farmers can pay attention to other important processes.
4) *Higher Quality:* Smart, sustainable agriculture makes it possible to avoid challenges, and consumers can get a good product of high quality.

V. Conclusion and Future Work

Smart agriculture is one of the strong requirements for today's era, making farming more efficient, sustainable, and profitable. In this article, one of the research problems of smart farming is addressed. We proposed an efficient smart CDD and prevention system, called CROPCARE, to deal with the crop diseases-related problems faced by farmers in a more appropriate way by using various sustainable technologies, such as the IoT cloud computing, mobile vision, etc. The IoT sensors are used to collect various important data from the monitoring fields, which are stored in cloud servers. The collected data of various images are analyzed using deep learning models in which the Google cloud platform has been used for training purposes. The trained model is optimized and integrated with the mobile app (a prototype of the proposed framework). In this proposed work, a recommendation system module is also designed to provide suggestions to farmers to take remedial actions on the occurrence of a specific disease. The possible recommendations can also be updated by the administrative authorities time to time. It also provides the pesticide calculator that tells the amount of pesticide and fertilizer to be applied in crops based on the number of trees and areas in hectare or an acre. The weather station feature is also embedded with the system so the farmer can know the weather conditions before watering the crops. In the future, we will extend the functionalities of the system and incorporate pest detection features as well. As our model is trained on 38 classes only so in the future to increase the data set and to use some advance technologies such as NLP to perform validation techniques.

References

[1] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, 2018, pp. 4510–4520.

[2] W.-J. Hu, J. Fan, Y.-X. Du, B.-S. Li, N. N. Xiong, and E. Bekkering, "MDFC–ResNet: An agricultural IoT system to accurately recognize crop diseases," *IEEE Access*, vol. 8, pp. 115287–115298, 2020.

[3] M. Islam, A. Dinh, K. Wahid, and P. Bhowmik, "Detection of potato diseases using image segmentation and multiclass support vector machine," in *Proc. IEEE 30th Can. Conf. Elect. Comput. Eng. (CCECE)*, Windsor, ON, Canada, 2017, pp. 1–4.

[4] D. Das, M. Singh, S. S. Mohanty, and S. Chakravarty, "Leaf disease detection using support vector machine," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Chennai, India, 2020, pp. 1036–1040.

[5] S. Savary, L. Willocquet, S. J. Pethybridge, P. Esker, N. McRoberts, and A. Nelson, "The global burden of pathogens and pests on major food crops," *Nat. Ecol. Evol.*, vol. 3, pp. 430–439, Feb. 2019.

[6] S. Ramesh *et al.*, "Plant disease detection using machine learning," in *Proc. Int. Conf. Design Innovat. 3Cs Comput. Commun. Control (ICDI3C)*, Bangalore, India, 2018, pp. 41–45.

[7] M. Hassan and M. S. Uddin, "Project report on aerial agricultural vehicle," Ph.D. dissertation, Dept. Elect. Electron. Eng., East West Univ., Dhaka, Bangladesh, 2018.

[8] G. Shrestha, Deepsikha, M. Das, and N. Dey, "Plant disease detection using CNN," in *Proc. IEEE Appl. Signal Process. Conf. (ASPCON)*, Kolkata, India, 2020, pp. 109–113.

[9] H. Wang, G. Li, Z. Ma, and X. Li, "Application of neural networks to image recognition of plant diseases," in *Proc. Int. Conf. Syst. Informat. (ICSAI)*, Yantai, China, 2012, pp. 2159–2164.

[10] U. Mokhtar, M. A. S. Ali, A. E. Hassanien, and H. Hefny, "Identifying two of tomatoes leaf viruses using support vector machine," in *Information Systems Design and Intelligent Applications*. New Delhi, India: Springer, 2015, pp. 771–782.

[11] B. S. Kusumo, A. Heryana, O. Mahendra, and H. F. Pardede, "Machine learning-based for automatic detection of corn-plant diseases using image processing," in *Proc. Int. Conf. Comput. Control Informat. Appl. Recent Challenges Mach. Learn. Comput. Appl.*, Tangerang, Indonesia, 2018, pp. 93–97.

[12] Y. Tian, G. Yang, Z. Wang, E. Li, and Z. Liang, "Detection of apple lesions in orchards based on deep learning methods of CycleGAN and YOLOV3-dense," *J. Sens.*, vol. 2019, pp. 1–13, Apr. 2019. [Online]. Available: https://doi.org/10.1155/2019/7630926

[13] W. Guan, S. Yu, and W. Jianxin, "Automatic image-based plant disease severity estimation using deep learning," *Comput. Intell. Neurosci.*, vol. 2017, Jul. 2017, Art. no. 2917536. [Online]. Available: https://doi.org/10.1155/2017/2917536

[14] Y.-H. Hu, X.-W. Ping, M.-Z. Xu, W.-X. Shan, and Y. He, "Detection of late blight disease on potato leaves using hyperspectral imaging technique," *Guang Pu Xue Yu Guang Pu Fen Xi*, vol. 36, no. 2, pp. 515–519, Feb. 2016.

[15] R. Yakkundimath, G. Saunshi, and K. Vishwanath, "Plant disease detection using IoT," *Int. J. Eng. Sci. Comput.*, vol. 8, no. 9, pp. 18902–18906, 2018.

[16] N. Petrellis, "A smart phone image processing application for plant disease diagnosis," in *Proc. 6th Int. Conf. Mod. Circuits Syst. Technol. (MOCAST)*, Thessaloniki, Greece, 2017, pp. 1–4.

[17] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, Dec. 2017.

[18] J. G. M. Esgario, P. B. C. Castro, L. M. Tassis, and R. A. Krohling, "An app to assist farmers in the identification of diseases and pests of coffee leaves using deep learning," *Inf. Process. Agr.*, to be published.

[19] V. Singh and A. K. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques," *Inf. Process. Agr.*, vol. 4, no. 1, pp. 41–49, 2016.

[20] J. Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosyst. Eng.*, vol. 180. pp. 96–107, Apr. 2019.

[21] S. Arivazhagan, S. N. Shebiah, S. Ananthi, and S. V. Varthini, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features," *Agr. Eng Int. CIGR J.*, vol. 15, no. 1, pp. 211–217, 2013.

[22] T. O. Emmanuel. (2018). *PlantVillage Dataset 2018*. [Online]. Available: https://www.kaggle.com/emmarex/plantdisease

[23] S. H. Tsang, *Review: MobileNetV2—Light Weight Model (Image Classification)*, Towards Data Sci., 2019. [Online]. Available: https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c

[24] B. Raj, *A Simple Guide to the Versions of the Inception Network*, Towards Data Sci., 2018. [Online]. Available: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

[25] V. Fung. (2017). *An Overview of ResNet and Its Variants*. [Online]. Available: https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035

[26] J. Amara, B. Bouaziz, and A. Algergawy, "A deep learning-based approach for banana leaf diseases classification," in *Lecture Notes in Informatics (LNI)*. Bonn, Germany: Gesellschaft für Informatik, 2017, pp. 79–88.

[27] J. D. Pujari, R. Yakkundimath, and A. S. Byadgi, "Image processing based detection of fungal diseases in plants," *Procedia Comput. Sci.*, vol. 46, no. 1, pp. 1802–1808, 2015.

[28] A. A. Bharate and M. S. Shirdhonkar, "A review on plant disease detection using image processing," in *Proc. Int. Conf. Intell. Sustain. Syst. (ICISS)*, Palladam, India, 2017, pp. 103–109.

[29] T. B. Thakur and A. K. Mittal, "Real time IoT application for classification of crop diseases using machine learning in cloud environment," *Int. J. Innovat. Sci. Mod. Eng.*, vol. 6, no. 4, pp. 1–4, 2020.

[30] V. G. N. Nguyen, A. Drogoul, and H. X. Huynh, "Toward an agent-based multi-scale recommendation system for brown plant hopper control," in *Proc. 6th UKSim/AMSS Eur. Symp. Comput. Model. Simulat.*, 2012, pp. 9–14.

[31] M. Brahimi, M. Arsenovic, S. Laraba, S. Sladojevic, K. Boukhalfa, and A. Moussaoui, *Deep Learning for Plant Diseases: Detection and Saliency Map Visualisation* (Human–Computer Interaction Series), J. Zhou, F. Chen, Eds. Cham, Switzerland: Springer, 2018, pp. 93–117.

[32] M. H. Saleem, S. Khanchi, J. Potgieter, and K. M. Arif, "Image-based plant disease identification by deep learning meta-architectures," *Plants*, vol. 9, no. 11, p. 1451, 2020.

[33] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, "Deep learning for single image super-resolution: A brief review," *IEEE Trans. Multimedia*, vol. 21, no. 12, pp. 3106–3121, Dec. 2019.

[34] S. K. Tichkule and D. H. Gawali, "Plant diseases detection using image processing techniques," in *Proc. Online Int. Conf. Green Eng. Technol. (IC-GET)*, Coimbatore, India, 2016, pp. 1–6.

[35] S. Yoganand, P. R. Narasingaperumal, and S. Rahul, "Prevention of crop disease in plants (groundnut) using IoT and machine learning models," *Int. Res. J. Eng. Technol.*, vol. 7, no. 3, pp. 1164–1169, 2020.

[36] S. K. Roy, S. Misra, N. S. Raghuwanshi, and S. K. Das, "AgriSens: IoT-based dynamic irrigation scheduling system for water management of irrigated crops," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 5023–5030, Mar. 2021.

[37] A. Bhattacharya, S. Misra, N. Pathak, and A. Mukherjee, "IDeA: IoT-based autonomous aerial demarcation and path planning for precision agriculture with UAVs," *ACM Trans. Internet Things*, vol. 3, no. 1, p. 16, 2020.

[38] A. Mukherjee, S. Misra, N. S. Raghuwanshi, and S. Mitra, "Blind entity identification for agricultural IoT deployments," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3156–3163, Apr. 2019.

[39] R. Saha, S. Misra, and P. K. Deb, "FogFL: Fog-assisted federated learning for resource-constrained IoT devices," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8456–8463, May 2021.