The Journal of Supercomputing
https://doi.org/10.1007/s11227-023-05131-4

E-CropReco: a dew-edge-based multi-parametric crop recommendation framework for internet of agricultural things

Somnath Bera[1] · Tanushree Dey[1] · Anwesha Mukherjee[1] · Rajkumar Buyya[2]

Accepted: 21 February 2023

Abstract
Crop productivity prediction and recommendation is a significant research area of smart agriculture. This paper proposes an Internet of Things (IoT) framework based on dew computing, edge computing, and federated learning, where soil parameters, environmental parameters, and weather data are analysed to predict the crop productivity of a land, and then recommend suitable crop for the land. The dew layer preprocesses and accumulates the received sensor data, and forwards to the edge server. The edge server analyses the sensor data and the weather data, and then sends the result to the cloud along with the model characteristics and to the mobile device. The proposed framework is simulated in iFogSim. The theoretical analysis shows that the proposed framework has reduced the delay by 60–70% approximately and power consumption by 70–80% approximately than the conventional sensor-cloud framework. We also observe that the proposed framework has reduced the delay by 12–35% approximately and power consumption by 30–50% approximately than the edge-cloud framework. We compare four machine learning algorithms based on their performance in data analysis in terms of precision, recall, accuracy, and F-score. We observe that each classifier obtains more than 95% prediction accuracy. An Android application is also proposed for crop recommendation.

Keywords Crop recommendation · Dew computing · Edge computing · Internet of agricultural things

✉ Anwesha Mukherjee
anweshamukherjee2011@gmail.com

Extended author information available on the last page of the article

Published online: 04 March 2023

1 Introduction

IoT has become a key component of smart applications [1]. The use of IoT in agriculture has also gained the interest of the researchers [2]. Internet of Agricultural Things (IoAT) refers to the use of IoT in agriculture [3]. The crop productivity prediction of land plays a significant role in smart farming. The increasing population growth in the fixed land area has raised the necessity of predicting the right crop for the right land. The appropriate guidance to the farmers for efficient utilization of the land is highly important for their economical profit. The proper guidance consists of various issues like crop prediction, cost of farming, and preparation of soil. Previously, the crop productivity prediction was carried out manually by analysing the prior knowledge of the cultivator on the selected crop [4]. However, the manual analysis does not consider the dynamic behaviour of the attributes such as soil parameters and environmental parameters [4]. Hence, the use of artificial intelligence in crop productivity prediction is required for boosting the prediction of accuracy level, and to resolve the problems regarding the manual analysis.

The analysis of the data related to the weather and soil parameters can provide better crop productivity prediction [5]. In the present work, we perform crop productivity prediction of land using machine learning. In our proposed work, we consider soil, environmental, and weather parameters (Nitrogen (N), Potassium (K), and Phosphorus (P) levels of soil, pH, temperature, humidity, and rainfall) as the input features, and take various crop classes in the input dataset. Four different machine learning algorithms (Support Vector Machine (SVM), K-Nearest Neighbour (KNN), Linear Discriminant Analysis (LDA), and Decision Tree (D-Tree)) [5–7] are used in data analysis for classifying the input dataset, and achieved a standard level of prediction accuracy of crop yield.

1.1 Motivation and contributions

The primary purpose of agricultural planning is to maximize crop yields while utilizing a limited amount of land resources to facilitate the economy of a country. To meet this objective, a data analytics framework is required that will predict crop productivity of land, and recommend suitable crop for the land. In the conventional data analysis approaches, cloud servers are used to store and analyse the data. However, the use of remote cloud has constraints in terms of delay, network connectivity, security, etc. The conventional edge-cloud framework reduces the delay. However, the agricultural lands are located in rural regions, where the network connectivity is not usually good. In such a case, the data transmission from sensors to the edge server is affected. The motivation of the proposed work is to provide a crop productivity prediction and recommendation framework that will overcome these challenges. The key contributions of this paper are:

- A dew computing, edge computing, and federated learning-based crop productivity prediction and recommendation framework is proposed for IoAT. The proposed framework is named as *E-CropReco* (Dew-Edge Computing-based Crop

Recommendation). The theoretical analysis demonstrates that the proposed framework has lower delay and power consumption than the existing sensor-cloud and edge-cloud frameworks. The architecture is simulated in iFogSim to analyse its performance in terms of delay, network usage, and energy consumption.

- In the agricultural fields at rural regions, the network connectivity is a major issue. The sensor data transmission to the edge server is affected due to the poor network connectivity. To deal with this problem, we use dew computing in this framework. Between the microcontroller and edge server, a mobile device is used that provides the dew layer in our framework. The mobile device is allotted for the data collection, accumulation, and pre-processing. By using dew computing the temporary data storage is performed when the network connectivity is not available. When the network connectivity of the mobile device is available, the data from the dew layer are sent to the edge server. The edge server performs data analysis and generates the result. By using federated learning, the cloud overhead is reduced and the data privacy is enhanced.
- A comparative study among four machine learning algorithms is performed based on their performance in analysing soil parameters, environmental parameters, and weather data. Their performance is measured in terms of precision, recall, accuracy, and *F*-score.
- Based on the data analysis using machine learning crop productivity of a land is predicted, and the suitable crop for the land is recommended.

The rest of the paper is organized as follows. Section 2 discusses the related works, Sect. 3 presents the proposed framework. Section 4 evaluates the performance of E-CropReco. Finally, we conclude in Sect. 5.

2 Related work

Machine learning techniques were applied in crop yield study to enhance the agricultural growth of the country. We have performed a comprehensive literature study to acquire an overview of the application of machine learning in crop yield prediction.

2.1 Use of machine learning and deep learning

Over the past few decades, machine learning (ML) approaches made significant advancements. Instead of attempting to deduce the nature of the mechanical processes behind the outcomes, ML is primarily concerned with making predictions about the future [5]. ML applications are widely employed for addressing a number of issues that humans frequently fail at or take a long time to solve. ML techniques were applied in crop yield study to enhance the agricultural growth of the country. Usually, the ML methods are categorized into three types: supervised, unsupervised, and reinforcement learning. The use of these three types of ML methods in agriculture is explained as follows.

2.1.1 Use of supervised learning

Four ML algorithms namely linear regression, elastic net (EN), KNN, support vector regression (SVR) were used for predicting the potato tuber yield in [6]. Statistical parameters were generated from the data that was collected using modelling approaches to produce yield projections. Five ML algorithms were used for predicting the mustard crop yield by analysing the soil data in [7]. Among them KNN and Artificial Neural Network (ANN) both were found as effective algorithms to predict the mustard crop yield. Different weather and soil parameters such as temperature, rainfall, soil, seed, and wind speed data were collected and analysed in [8] to help the farmer for improving the crop production.

2.1.2 Use of unsupervised learning

Clustering divides a set of data into groups based on similarity and then labels a relatively small number of groups. Clustering is unsupervised learning because it does not rely on preset classes or training instances with class labels [9]. The K-means classification technique was used to classify the dataset and pick the optimal group or combination of precipitation and temperature for increasing crop yields [10]. The primary flaw in the current clustering algorithms was highlighted, including the uniform input of the number of clusters and the random initialization of cluster centres. In order to produce high-quality clusters and estimate the harvest with better accuracy, a modified version of the K-means clustering method was created in [9].

2.1.3 Use of reinforcement learning

Reinforcement learning uses the dynamic programming idea to create and train algorithms by using a strategy of reward and punishment. Deep reinforcement learning is a combination of reinforcement learning and deep learning. In order to anticipate crop yield, the suggested study [11] built a Deep Recurrent Q-Network model, which is a Recurrent Neural Network-based deep learning algorithm over Q-Learning reinforcement learning algorithm. In order to strengthen the mastering set of rules for yield prediction, the research [12] developed a version of the Deep Recurrent Q-Network SVM deep mastering set of rules over Q-Learning. By maintaining the baseline distribution of information, the suggested model predicted the trendy crop production by maintaining 93.7% accuracy level. A deep reinforcement learning-based plant industrial sensor data processing model was proposed in [13]. It offered a fresh concept for the intelligence of plant manufacturers and the processing of sensor data.

2.1.4 Use of deep learning

In [14], deep transfer learning framework was presented for predicting the crop yield in the developing countries by using the remote sensing data. A parametric statistical model augmented with the help of deep neural network (DNN), named as semi-parametric neural network (SNN), achieved better predictive performance, as presented

in [5]. A multi-parametric DNN was proposed to model the impact of changes on climate, as well as different soil and climate characteristics in order to predict crop production [4]. With the use of previous knowledge about different functional forms linked to the agricultural yield field, the multi-parametric DNN outperformed the DNN in terms of statistical efficiency. However, because of the usage of a hierarchy of features, the complexity grows and the quality of the concealed representation might be compromised. Another limitation of the system was that for medium-sized datasets the system might not be able to provide the optimal representation.

2.1.5 ML algorithms used in E-CropReco

The four ML algorithms (SVM, KNN, LDA, and D-Tree) used for data analysis in our proposed framework are briefly discussed as follows.

- *SVM* Support Vector Machine or SVM is the general discriminant classifier widely used in the area of pattern recognition. In SVM model, several classes are represented in the concept of surface named as hyperplane, where the boundary is drawn between the data instances and plotted in multidimensional space. SVM generates hyperplane in an iterative manner for minimizing the error. SVM depends on the kernel functions. Kernel functions are generally used to map the original features to the higher dimensional space nonlinearly. Non-separable problems are converted into separable problems [6]. SVM is a supervised ML algorithm that can be used for classification and regression tasks. The fundamental goal of SVM is to identify the optimal boundary for classifying the data into distinct groups. It can work with high-dimensional data and is effective with small datasets.
- *KNN* The key concept behind the K-Nearest Neighbour or KNN classifier is to find the similarities in close proximity that means how the similar data points are close to each other. KNN algorithm [7] uses data and it classifies new data points based on the distance function. The techniques belonging to KNN are based on single nearest neighbour classification and unfamiliar sample classification on the votes of K-nearest neighbour. Here, K is the user-defined constant. Based on Euclidean distance function the algorithm searches K neighbours closest to the unlabelled sample from the training space. KNN is one of the most straightforward ML algorithms, and it is based on the supervised learning method. This algorithm assumes that the new data and the existing data are similar, and it places the new data in the category that the existing data are most similar to. KNN properly works with labelled and error free data, and also works effectively on small dataset.
- *LDA* Linear discriminant analysis or LDA is a supervised classifier that is used for dimension reduction. It measures several categories of information. LDA is used for placing the extracted features from higher to lower dimensional space depending on the number of class for avoiding the dimensional curse, and also used for reducing the resource. LDA focuses on fetching the appropriate projection direction by using the Fisher function [7]:

- For maximizing among scattered classes.
- For minimizing within each class.

LDA is generally used for taking the class covariance within itself, mean of total data, mean of total class, and class probabilities. LDA is one of the most straightforward and efficient supervised ML algorithms for multi-class classification tasks with categorical output data. It is used when dataset is linearly separable.

- *D-Tree* Decision Tree or D-Tree is a powerful classification tool used for prediction. From its tree structure, internal nodes are defined as test, leaf nodes are defined as class label, and branches are defined as features conjunction that indicates those particular classes. D-Tree [5] follows the technique by considering all probable decisions' outcomes and finds the conclusion by tracing every path. D-Tree is easy to comprehend since it uses the same steps that people do while making decisions in everyday life. D-Trees are simple to understand as it follows the tree like structure logic. It can be useful for solving decision-making problems.

In our work, the crop productivity of a land is predicted. The dataset is comparatively small, labelled data are considered, and decision-making regarding which crop is suitable for which land takes place. Thus, we have selected these four ML algorithms for data analysis in our framework.

2.2 IoT, cloud, edge, and fog computing

IoT is a principal component of smart solutions. A time-aware smart object recommendation model was proposed in [15] that considered both the social similarity of the smart object and the user's preference over time. The suggested model used events associated with user object usage to develop a latent probabilistic model that tracked user preferences over time. By inferring customer preferences from user-generated heterogeneous information, a recommendation model was proposed for physical businesses in [16]. In IoT, to analyse the high volume of sensor data, usually, cloud computing is used, where the cloud servers are used for data storage and processing. However, the use of remote cloud suffers from increase in delay, network traffic, and computation overhead on the cloud. As a solution, edge and fog computing come into the scenario. For smart agriculture, edge computing [17, 18] has a lot of potential. Smart farming benefits not only the scientists and agronomists, but also the cultivators by allowing them access to current technologies and gadgets, which help to maximize the product quality and quantity while lowering the agricultural cost [18, 19]. Edge computing allows smart agriculture services to be accessed and used more effectively. Fog computing is another approach that improves the quality of service for the modern information and communication technology. Unlike cloud computing, which processes all data on the cloud, the intermediate fog nodes also participate in data processing in fog computing. Thus, the use of fog computing reduces the network traffic and computation overhead on the cloud. An architecture

based on two levels of communication and processing nodes (edge and fog nodes) was proposed using low cost sensing technologies in [20]. A data sensing framework was developed in [21] for the complete crop lifetime by merging edge computing with IoT. The technique had the potential to greatly reduce the data collection time as well as energy consumption. In [17, 18], the use of edge computing in agriculture was highlighted. The use of IoT and edge computing in agriculture was explored in various research works, especially for soil quality monitoring, irrigation system, etc. [22–24]. However, crop productivity recommendation has not gained so much attention from the researchers, though the planting of suitable crop and proper utilization of land are highly significant for the economical profit of the farmers.

2.3 Dew computing

Dew computing is an emerging research trend [25–28]. Dew computing combines the use of end nodes such as mobile devices with cloud computing to provide local data storage and access to the data even in offline mode. A dew virtual machine is composed of a dew server, data analytics server, and artificial intelligence of the dew. The dew server is a lightweight server that provides micro-services according to the requirements and interacts with the cloud to periodically synchronize the content. The dew analytics server collects data regarding the use of the dew server. The artificial intelligence of the dew is used for data customization and update after the reception for improving the user experience. In developing countries, the agricultural fields are located usually in remote places, where network connectivity is not always available. In such a case, the use of dew computing can provide local storage and access to the data even in offline mode. In [29], dew computing was used for health care, where the authors focused on the early prediction of bronchial asthma. In [30], dew computing-based IoT was discussed for smart city applications. In [28], the use of dew computing and blockchain was highlighted for smart city applications. Though, dew computing was used for health care and smart city applications, its use in smart agriculture is not largely explored.

In our paper, we have used dew computing, edge computing, and federated learning for designing a crop productivity prediction and recommendation system. As per our knowledge, for the first time we are integrating IoAT with dew computing, edge computing, and federated learning, to develop a delay-aware and power-efficient paradigm for crop productivity prediction. Federated learning (FL) is a method in which an algorithm is trained across multiple decentralized edge servers containing local data [31, 32]. The local data are not exchanged in FL. Usually, the lower level nodes perform the local processing, and the result along with model characteristics are forwarded to the next level. This in turn helps in privacy protection. Table 1 presents a comparative study between the existing approaches and the proposed framework. We observe that the proposed framework (E-CropReco) is novel and beneficial than the existing approaches.

Table 1 Comparison of proposed framework with existing frameworks on crop yield prediction

Work	Multi-parametric crop data analysis	Multi-crop dataset-based analysis	Use of edge computing	Use of FL	Use of dew computing	Delay/time and power/energy calculation
Abbas et al. [6]	✓	✗	✗	✗	✗	✗
Crane et al. [5]	✓	✗	✗	✗	✗	✗
Pandith et al. [7]	✓	✗	✗	✗	✗	✗
Gupta et al. [8]	✓	✗	✗	✗	✗	✗
Kalaiarasi et al. [4]	✓	✓	✗	✗	✗	✗
Thilakarathne et al. [33]	✓	✓	✗	✗	✗	✗
Cruz et al. [34]	✓	✓	✓	✗	✗	✗
E-CropReco (proposed)	✓	✓	✓	✓	✓	✓

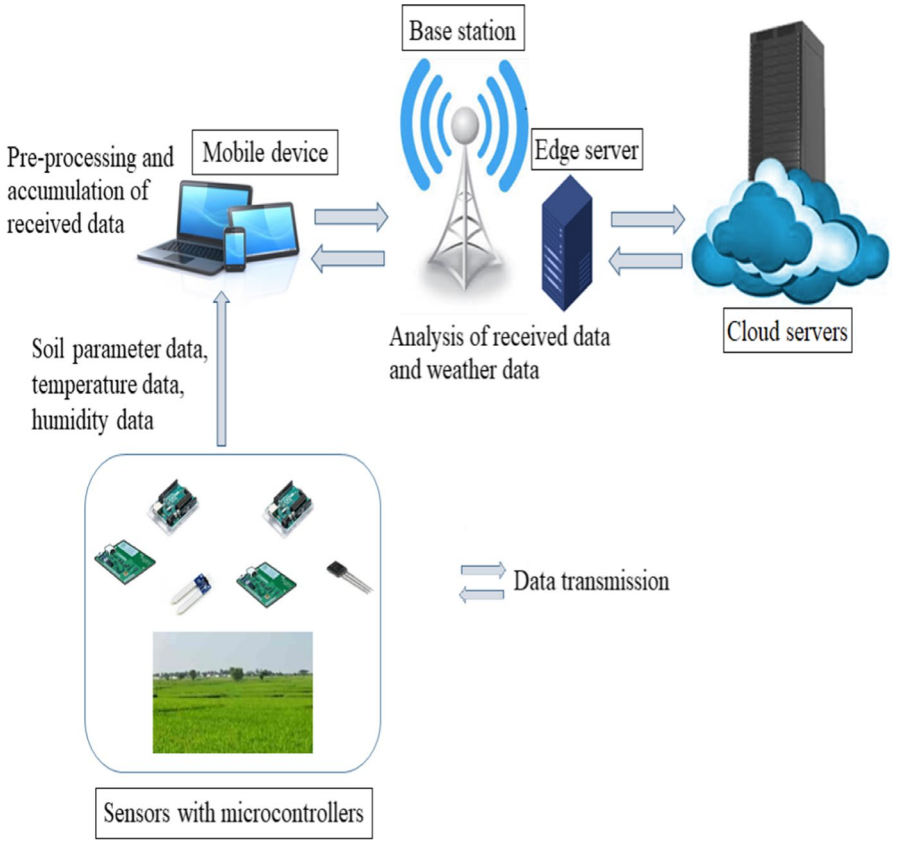


Fig. 1 Architecture of E-CropReco

3 E-CropReco: proposed framework for crop recommendation

This section presents the architecture and working model of E-CropReco, and its delay and power consumption.

3.1 Architecture and working model

The principal components of E-CropReco are (refer Fig. 1):

- Sensors with microcontrollers: A sensor node (S) is mathematically defined as:

$$S =< S_{\text{id}}, S_{\text{st}}, S_{\text{ob}} >,$$

where S_{id}, S_{st}, and S_{ob} denote the id of the sensor, status of the sensor (active or not), and the object type with which the sensor is attached (e.g. temperature, humidity, etc.), respectively. A microcontroller is mathematically defined as:

$$M =< M_{\text{id}}, M_{\text{st}} >,$$

where M_{id} and M_{st} denote the id and status of the microcontroller, respectively.

- Mobile device: A mobile device working as the dew layer is mathematically defined as:

$$D =< D_{\text{id}}, D_{\text{se}}, D_{\text{sp}} >,$$

where D_{id}, D_{se}, and D_{sp} denote the id of the mobile device, security methods used by the mobile device, and specification of the mobile device, respectively.

- Base station with edge server: An edge server is defined as:

$$E =< E_{\text{id}}, E_{\text{se}}, E_{\text{sp}} >,$$

where E_{id}, E_{se}, and E_{sp} denote the id of the edge server, security methods used by the edge server, and specification of the edge server, respectively.

- Cloud servers: A cloud computing instance is defined as:

$$C =< C_{id}, C_{pr} >,$$

where C_{id} and C_{pr} represent the id of the cloud computing instance and a set containing the processing unit IDs of the necessary cloud servers of the instance, respectively.

The working model of the proposed framework is described as follows, and the corresponding flow diagram is presented in Fig. 2.

- The sensor nodes collect the soil parameter data (such as Nitrogen (N), Potassium (K), Phosphorous (P) levels of the soil, and pH) and environmental parameter data (such as temperature and humidity), and send to the connected microcontroller.
- The microcontroller sends the data to a mobile device connected with the network.
- The mobile device pre-processes the received data, accumulates, and then sends the data to the base station of the region, where the land is located. Though we have used here mobile devices in our framework, instead of mobile device any processing device with good storage and computation abilities can be used as the dew layer.
- The base station after receiving the data stores it inside the attached edge server. We assume that the edge server already has the weather data (such as rainfall) of the region containing the land. The edge server analyses the soil parameters, environmental parameters, and the weather data using ML to predict the crop productivity and recommend suitable crop for the land under examination. The ML algorithm suggests which crop is suitable for the land. The result is sent to the cloud along with the model characteristics and to the mobile device.

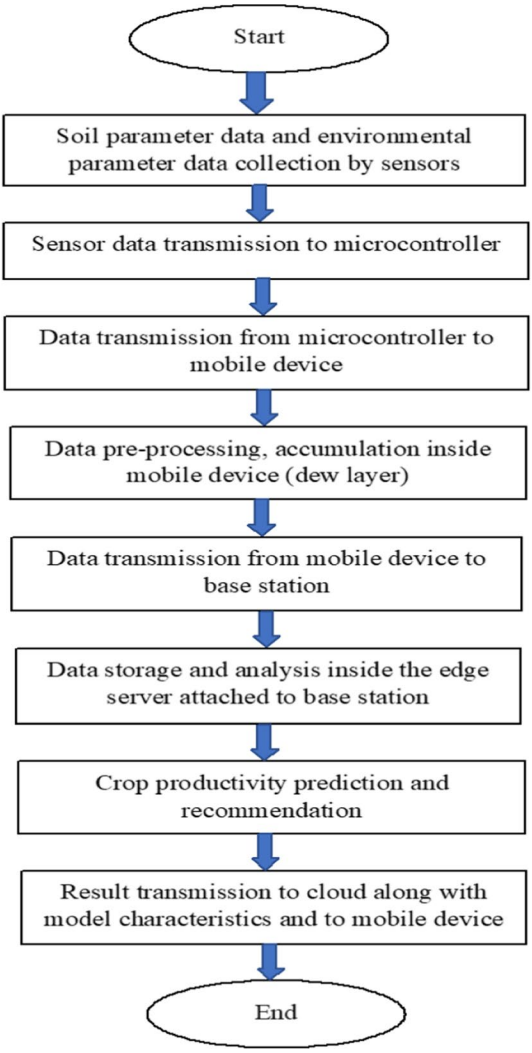


Fig. 2 Flow diagram of working model of E-CropReco

The cloud can also send data to the edge server if required. In our framework, the mobile device works as the dew layer by containing the data collected from the sensors. If the network connectivity is not available, then the dew layer i.e. the mobile device holds the data. When the connectivity is available, it forwards the data to the edge server. To connect the microcontroller to the mobile device Bluetooth can be used. As the data transmission takes place from microcontroller to mobile device, the microcontroller will act as the client and the mobile device will act as the server during data transmission between these two nodes. The Bluetooth address of the mobile device will be traced and a pairing agent will be executed. Then the data can be sent from the microcontroller to the mobile device.

3.1.1 Use of ML

In this paper, we have used four ML algorithms SVM, KNN, LDA, and D-Tree for analysing the data. In the performance analysis section, their performance is compared with respect to accuracy, precision, recall, and *F*-score.

3.1.2 Accuracy

The ratio of the number of proper predictions produced by the classifier to the total number of predicted crop yields made by the classifier is denoted as the classification accuracy. The classification accuracy is calculated as,

$$\text{Accuracy} = (X + Y)/(X + Y + Z + W)$$

where *X*, *Y*, *Z*, and *W* denote True Positive, True Negative, False Positive, and False Negative, respectively.

3.1.3 Precision

It is the volume of properly predicted crops over True Positive and False Positive, mathematically defined as,

$$\text{Precision} = X/(X + Z)$$

3.1.4 Recall

It is mathematically defined as,

$$\text{Recall} = X/(X + W)$$

where *X* and *W* denote True Positive and False Negative, respectively.

3.1.5 F-score

F-score helps to evaluate the recall and precision at the same time. The *F*-score is maximum if the recall is equal to the precision. The *F*-score is mathematically calculated as,

$$F_{\text{score}} = 2 * (\text{Precision*Recall})/(\text{Precision+Recall})$$

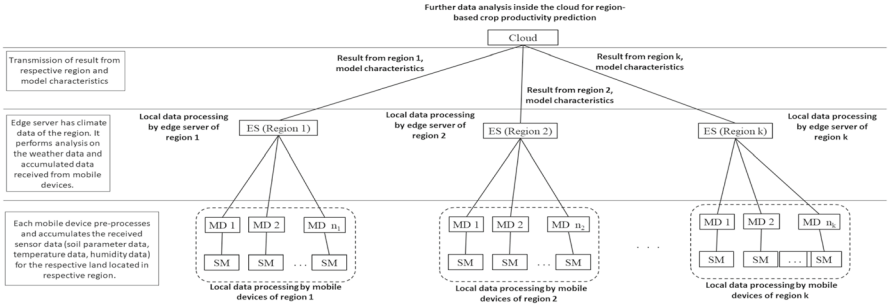


Fig. 3 Region-based data collection and processing in E-CropReco using FL

3.1.6 Use of dew computing

In our framework, we have used dew computing to allow the holding of data even if the network connectivity is lost. The mobile device is used as the dew layer here. The mobile device pre-processes and accumulates the data received from the sensor nodes, and then transmits to the edge server when the network connectivity is available. In the agricultural field specially located at the remote region of the developing countries, the good and seamless network connectivity is a major concern. Therefore, continuous data transmission from the sensor nodes to the edge server may not be possible. Now, if the mobile device is used as the dew layer, the data can be hold for a period of time if the connectivity is lost. When the network is again available, the dew layer transmits the data to the edge server.

3.1.7 Use of FL

In E-CropReco, we are using FL to maintain the local data inside the respective device instead of storing it inside the cloud. The mobile device holds the sensor data (received through the microcontroller), pre-processes, and accumulates. Here, the local data are stored inside the mobile device. The accumulated data are sent to the edge server for further analysis. The edge server has the weather data of the region, where it is located. The edge server analyses the weather data and received sensor data using ML to predict crop productivity, and generate recommendations regarding the suitable crop for the respective land. The edge servers send the result (recommendations) to the mobile device. The weather data as well as the received data are not forwarded to the cloud but rather stored and analysed by the edge server. Here, the local regional data processing and storage takes place inside the edge server. Only, the result generated after the analysis and the model characteristics (learned parameters, gradients) are sent to the cloud. The cloud deploys aggregate models to enhance the overall efficacy of the system. If the framework is considered as a three-tier model ((1) Tier 1: Sensors, microcontrollers, and mobile device, (2) Tier 2: Edge server, (3) Tier 3: Cloud), then at each tier the local data is maintained by the devices of the corresponding level, and the result after processing is forwarded to the next level. This in turn helps to reduce the data traffic, as well as

by maintaining the data locally, data privacy is enhanced compared to storing the entire data inside the cloud. In Fig. 3, we have demonstrated the use of FL in E-CropReco. In the figure, SM denotes sensors with microcontrollers, MD denotes mobile device, and ES denotes edge server. We assume there are k regions, each containing one edge server. Inside each region, there is a number of mobile devices. We assume region 1 contains n_1 number of mobile devices, region 2 contains n_2 number of mobile devices, and so on. As observed, the local data with respect to region 1 are maintained by edge server 1, the local data with respect to region 2 are maintained by edge server 2, and so on. The cloud receives the result data of each region along with the model characteristics, based on which it can perform region-based crop productivity prediction. The weather data of the regions are different from each other, as well as the values of the soil parameters and environmental parameters vary for different regions. Therefore, based on the results and model characteristics of different regions, the cloud can perform further analysis for region-based crop productivity prediction. In the present work, we have performed crop productivity prediction of a land. By collecting the data of different regions, region-based crop productivity prediction is a future research scope of the present work. In our framework, *horizontal FL* is used, and the features considered in the data set for each of the node are same. Though the features are same, the datasets hold different values based on the considered land.

3.1.7.1 Reason of using FL In our framework, we have used FL to increase the data privacy. If the entire data are sent to the cloud, then not only the traffic is increased but data privacy becomes a major concern. In such a case, the use of FL allows local data processing and transmission of only the result and model characteristics to the cloud. This in turn avoids huge data transmission to the cloud, reduces storage and computation overhead of the cloud, and enhances data privacy.

3.2 Delay and power consumption

The parameters used for calculating delay and power consumption of the E-CropReco framework are defined in Table 2. If g is a parameter whose value depends on the values of two parameters a and b, then g is a function of a and b, mathematically expressed as, $g = f(a, b)$.

3.2.1 Delay of the framework

Theoretically, the delay in data transmission from one node (node_1) to another node ($node_2$) is given as: $T_{12} = (1 + \phi_{12})(Data_{12}/R_{12})$, where Data_{12} is the amount of data transmitted, R_{12} is the data transmission rate, and ϕ_{12} is the link failure rate. In the practical scenario, the data transmission rate, link failure rate, etc. vary based on the present traffic, allocated bandwidth, etc. As the data transmission delay mainly depends on the amount of data transmitted, data transmission rate, and link failure rate, we have considered the data transmission delay as a function of the amount of data transmission, data transmission rate, and link failure rate, the function

Table 2 Parameters for delay and power calculation

Parameter	Definition
T_{SM}	Delay in data collection by sensors and transmission to microcontrollers (considering all sensors and microcontrollers)
P_{S}	Power consumption of all sensors in data collection and transmission
P_{M}	Power consumption of all microcontrollers in data reception and transmission
P_{Dr}	Power consumption of a mobile device per unit time in data reception
P_{Dt}	Power consumption of a mobile device per unit time in data transmission
P_{Da}	Power consumption of a mobile device per unit time in active mode
P_{Er}	Power consumption of an edge server per unit time in data reception
P_{Et}	Power consumption of an edge server per unit time in data transmission
P_{Ea}	Power consumption of an edge server per unit time in data analysis
P_{Cr}	Power consumption of cloud per unit time in data reception
P_{Ct}	Power consumption of cloud per unit time in data transmission
P_{Ca}	Power consumption of cloud per unit time in data analysis
$Data_{\mathrm{D}}$	Data amount pre-processed and accumulated by mobile device
$Data_{\mathrm{E}}$	Data amount analysed by edge server
$Data_{C}$	Data amount analysed by cloud
$Data_{\mathrm{MD}}$	Data amount received by mobile device from microcontrollers
$Data_{\mathrm{DE}}$	Data amount received by edge server from mobile device
$Data_{\mathrm{EC}}$	Data amount received by cloud from edge server
$Data_{\mathrm{ED}}$	Data amount received by mobile device from edge server
$Data_{\mathrm{CE}}$	Data amount received by edge server from cloud
Sp_{D}	Processing speed of a mobile device
Sp_{E}	Processing speed of an edge server
Sp_{C}	Processing speed of cloud
R_{MD}	Data transmission rate from microcontroller to mobile device
R_{DE}	Data transmission rate from mobile device to edge server
R_{EC}	Data transmission rate from edge server to cloud
R_{CE}	Data transmission rate from cloud to edge server
R_{ED}	Data transmission rate from edge server to mobile device

is denoted by f_t. The delay in data transmission from microcontrollers to mobile device is given as a function of respective data amount (Data_{MD}), data transmission rate (R_{MD}), and link failure rate (ϕ_{MD}):

$$T_{MD} = f_t(Data_{MD}, R_{MD}, \phi_{MD}).$$

The delay in data transmission from mobile device to edge server is given as a function of respective data amount ($Data_{DE}$), data transmission rate (R_{DE}), and link failure rate (ϕ_{DE}):

$$T_{DE} = f_t(Data_{DE}, R_{DE}, \phi_{DE}).$$

The delay in result data transmission from edge server to cloud is given as a function of respective data amount (Data_{EC}), data transmission rate (R_{EC}), and link failure rate (ϕ_{EC}):

$$T_{\text{EC}} = f_t(\text{Data}_{\text{EC}}, R_{\text{EC}}, \phi_{EC}).$$

The delay in result data transmission from edge server to mobile device is given as a function of respective data amount (Data_{ED}), data transmission rate (R_{ED}), and link failure rate (ϕ_{ED}):

$$T_{ED} = f_t(\text{Data}_{\text{ED}}, R_{\text{ED}}, \phi_{\text{ED}}).$$

The delay in data transmission from cloud to edge server is given as a function of respective data amount (Data_{CE}), data transmission rate (R_{CE}), and link failure rate (ϕ_{CE}):

$$T_{CE} = f_t(\text{Data}_{CE}, R_{CE}, \phi_{CE}).$$

Therefore, the total delay in data collection and transmission in E-CropReco is given as:

$$T_t = T_{SM} + T_{MD} + T_{DE} + T_{EC} + T_{ED} + T_{CE} \tag{1}$$

Theoretically, the delay in data processing of a node ($node_1$) is calculated as:

$$T_1 = \text{Data}_1 / Sp_1,$$

where Data_1 is the amount of data processed and Sp_1 is the processing speed of the node. In a practical scenario, a node executes different types of tasks, and the processor type, allotted memory, processing load, etc. can have an impact on the delay. However, the delay mainly depends on the amount of data processed and the data processing speed. Hence, we have considered the delay in data processing as a function of the amount of data and the processing speed, the function is denoted by f_p. The delay in data pre-processing and accumulation by the mobile device is given as a function of the respective data amount (Data_D) and processing speed (Sp_D):

$$T_{\text{Dp}} = f_p(\text{Data}_D, Sp_D).$$

The delay in data analysis by the edge server is given as a function of the respective data amount (Data_E) and processing speed (Sp_E):

$$T_{Ep} = f_p(\text{Data}_E, Sp_E).$$

The delay in data analysis by the cloud is given as a function of the respective data amount ($Data_C$) and processing speed (Sp_C):

$$T_{Cp} = f_p(\text{Data}_C, Sp_C).$$

Therefore, the total delay in data processing (pre-processing, accumulation, and analysis) in E-CropReco is given as:

$$T_p = T_{Dp} + T_{Ep} + T_{Cp} \tag{2}$$

The total delay in E-CropReco is given as the sum of the data transmission and processing delays as follows:

$$T = T_t + T_p \tag{3}$$

In conventional edge-cloud framework, the microcontrollers send the data to the edge server, and the edge server sends the data to the cloud after pre-processing. Therefore, the delay in data transmission in the conventional edge-cloud framework is given as,

$$T_{t_{ec}} = T_{SM} + T_{ME} + T_{pEC} + T_{ED} + T_{CE} \tag{4}$$

where T_{ME} is the data transmission delay from microcontrollers to the edge server, T_{pEC} is the data transmission delay from edge server to the cloud, T_{CE} is the data transmission delay from cloud to the edge server, and T_{ED} is the delay in data transmission from edge server to the user's mobile device. These delays depend on the respective data amount, data transmission rate, and link failure rate. The total delay in data processing in conventional edge-cloud framework is given as,

$$T_{p_{ec}} = T_{Ep_{ec}} + T_{Cp_{ec}} \tag{5}$$

where $T_{Ep_{ec}}$ is the delay in data pre-processing by the edge server and $T_{Cp_{ec}}$ is the delay in data processing by the cloud. These delays depend on the respective data amount and data processing speed. The total delay in conventional edge-cloud framework is given as the sum of the data transmission and processing delays as follows:

$$T_{ec} = T_{t_{ec}} + T_{p_{ec}} \tag{6}$$

In the rural regions, the network connectivity is poor, and most of the agricultural lands are located in the rural regions. In such a scenario, the data transmission to the edge server is affected in conventional edge-cloud framework due to poor network connectivity. As a result, the delay is increased. In conventional sensor-cloud framework, the data are sent to the cloud from the microcontroller, and the entire data are processed inside the cloud. A user using his or her mobile device can access the data. Therefore, the delay in data transmission in the conventional sensor-cloud framework is given as,

$$T_{t_{sc}} = T_{SM} + T_{MC} + T_{CD} \tag{7}$$

where T_{MC} is the data transmission delay from microcontrollers to the cloud and T_{CD} is the data transmission delay from cloud to the user's mobile device. These delays depend on the respective data amount, data transmission rate, and link failure rate. In sensor-cloud framework, the entire data processing happens inside the cloud. Therefore, the total delay in data processing in sensor-cloud framework is given as,

$$T_{p_{sc}} = T_{Cp_{sc}} \tag{8}$$

where $T_{Cp_{sc}}$ is the delay in data processing by the cloud. This delay depend on the respective data amount and data processing speed. The total delay in sensor-cloud framework is given as the sum of the data transmission and processing delays as follows:

$$T_{sc} = T_{t_{sc}} + T_{p_{sc}} \tag{9}$$

As most of the agricultural lands are located in rural regions, which suffer from poor network connectivity, the data transmission to the cloud is highly affected in conventional sensor-cloud framework. As a result, the delay is increased.

3.2.2 Power consumption of the framework

If $node_1$ denotes a node, then theoretically, the power consumption of the node (Pow_{node_1}) during a time period is given as the product of the time duration (let T_{node_1}) and the power consumption of the node per unit time (let P_{node_1}), i.e. $Pow_{node_1} = P_{node_1} \dot{T}_{node_1}$. Therefore, we can consider the power consumption of a node as a function of the power consumption per unit time and the time duration, the function is denoted by f_{pow}. The power consumption of the mobile device for data reception from microcontrollers is given as a function of its respective power consumption per unit time (P_{Dr}) and the delay (T_{MD}):

$$P_{D_{11}} = f_{pow}(P_{Dr}, T_{MD}).$$

The power consumption of the mobile device for data transmission to edge server is given as a function of its respective power consumption per unit time (P_{Dt}) and the delay (T_{DE}):

$$P_{D_{12}} = f_{pow}(P_{Dt}, T_{DE}).$$

The power consumption of the mobile device for data reception from edge server is given as a function of its respective power consumption per unit time (P_{Dr}) and the delay (T_{ED}):

$$P_{D_{13}} = f_{pow}(P_{Dr}, T_{ED}).$$

Therefore, the total power consumption of the mobile device for data transmission and reception is given as,

$$P_{D_1} = P_{D_{11}} + P_{D_{12}} + P_{D_{13}} \tag{10}$$

The power consumption of the mobile device for data pre-processing and accumulation is given as a function of its respective power consumption per unit time (P_{Da}) and the delay (T_{Dp}):

$$P_{D_2} = f_{pow}(P_{Da}, T_{Dp}).$$

The total power consumption of the mobile device is given as the sum of its power consumption during data transmission, reception, pre-processing, and accumulation, as follows:

$$P_D = P_{D_1} + P_{D_2} \tag{11}$$

The power consumption of the edge server for data reception from mobile device is given as a function of its respective power consumption per unit time (P_{Er}) and the delay (T_{DE}):

$$P_{E_{11}} = f_{pow}(P_{Er}, T_{DE}).$$

The power consumption of the edge server for data transmission to cloud is given as a function of its respective power consumption per unit time (P_{Et}) and the delay (T_{EC}):

$$P_{E_{12}} = f_{pow}(P_{Et}, T_{EC}).$$

The power consumption of the edge server for data transmission to mobile device is given as a function of its respective power consumption per unit time (P_{Et}) and the delay (T_{ED}):

$$P_{E_{13}} = f_{pow}(P_{Et}, T_{ED}).$$

The power consumption of the edge server for data reception from cloud is given as a function of its respective power consumption per unit time (P_{Er}) and the delay (T_{CE}):

$$P_{E_{14}} = f_{pow}(P_{Er}, T_{CE}).$$

Therefore, the total power consumption of the edge server for data transmission and reception is given as,

$$P_{E_1} = P_{E_{11}} + P_{E_{12}} + P_{E_{13}} + P_{E_{14}} \tag{12}$$

The power consumption of the edge server for data analysis is given as a function of its respective power consumption per unit time (P_{Ea}) and the delay (T_{Ep}):

$$P_{E_2} = f_{pow}(P_{Ea}, T_{Ep}).$$

The total power consumption of the edge server is given as the sum of its power consumption during data transmission, reception, and analysis, as follows:

$$P_E = P_{E_1} + P_{E_2} \tag{13}$$

The power consumption of the cloud for data reception from edge server is given as a function of its respective power consumption per unit time (P_{Cr}) and the delay (T_{EC}):

$$P_{C_{11}} = f_{pow}(P_{Cr}, T_{EC}).$$

The power consumption of cloud for data transmission to edge server is given as a function of its respective power consumption per unit time (P_{Ct}) and the delay (T_{CE}):

$$P_{C_{12}} = f_{pow}(P_{Ct}, T_{CE}).$$

Therefore, the total power consumption of the cloud for data transmission and reception is given as,

$$P_{C_1} = P_{C_{11}} + P_{C_{12}} \tag{14}$$

The power consumption of the cloud for data analysis is given as a function of its respective power consumption per unit time (P_{Ca}) and the delay (T_{Cp}):

$$P_{C_2} = f_{pow}(P_{Ca}, T_{Cp})$$

The total power consumption of the cloud is given as the sum of its power consumption during data transmission, reception, and analysis, as follows:

$$P_C = P_{C_1} + P_{C_2} \tag{15}$$

The total power consumption of the E-CropReco framework is given as the sum of the power consumption by its components, as follows:

$$P = P_S + P_M + P_D + P_E + P_C \tag{16}$$

In the conventional edge-cloud framework, the power consumption is given as,

$$P_{ec} = P_S + P_M + P_{E_{ec}} + P_{C_{ec}} \tag{17}$$

where $P_{E_{ec}}$ and $P_{C_{ec}}$ are the power consumption of the edge server and cloud, respectively, for data transmission, reception, and analysis. The power consumption of the edge server in different modes (data transmission or reception or analysis) are calculated depending on the respective delays and power consumption per unit time depending on the modes, and then added to calculate $P_{E_{ec}}$. The power consumption of the cloud in different modes (data transmission or reception or analysis) is calculated depending on the respective delays and power consumption per unit time depending on the modes, and then added to calculate $P_{C_{ec}}$. In the conventional sensor-cloud framework, the power consumption is given as,

$$P_{sc} = P_S + P_M + P_{C_{sc}} \tag{18}$$

where $P_{C_{sc}}$ is the power consumption of the cloud for data transmission, reception, and analysis. The power consumption of the cloud in different modes (data transmission or reception or analysis) is calculated depending on the respective delays and power consumption per unit time depending on the modes, and then added to calculate $P_{C_{sc}}$.

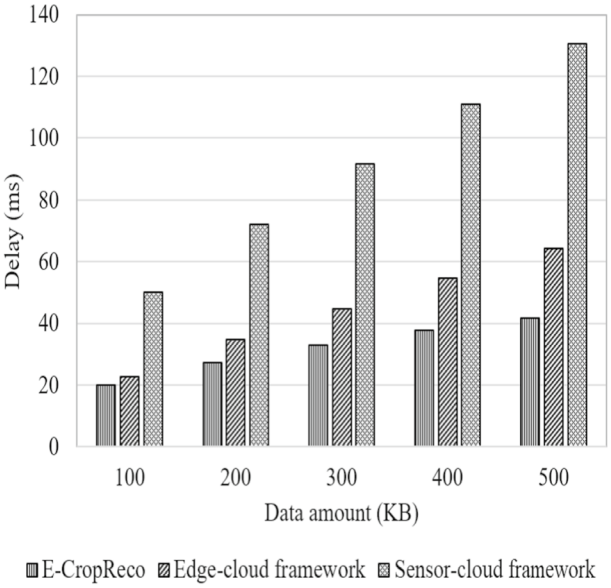


Fig. 4 Delay in proposed, edge-cloud, and sensor-cloud frameworks

4 Performance evaluation

In this section, we have performed theoretical analysis using MATLABR2022b, simulation using iFogSim, experimental data analysis using MATLABR2022b, and finally proposed an Android APP for crop recommendation.

4.1 Theoretical analysis

The theoretical analysis is performed to demonstrate the delay and power consumption of the framework depending on the data transmission rate, amount of data, data processing speed, power consumption per unit time, etc. (the equations are presented in Sect. 3.2). Though the data transmission delay depends on the amount of data transmission, data transmission rate, link failure rate, etc. in the practical scenario, the data transmission rate, link failure rate, etc. depends on various factors such as the network traffic and allocated bandwidth. Similarly, though, the data processing delay depends on the data amount and data processing speed, in the practical scenario, the data processing speed depends on different factors such as the number of tasks under execution by the device, allocated memory, and processor type. Here, we have performed the theoretical analysis using MATLABR2022b. The uplink and downlink data transmission rate (between mobile device and edge server, and edge server and cloud) are assumed 1–2 Mbps and 2–4 Mbps, respectively. The data amount to be processed is assumed 100–500 KB. The processing speed of the mobile device, edge server, and cloud are considered 500 MHz, 1 GHz, and 2 GHz,

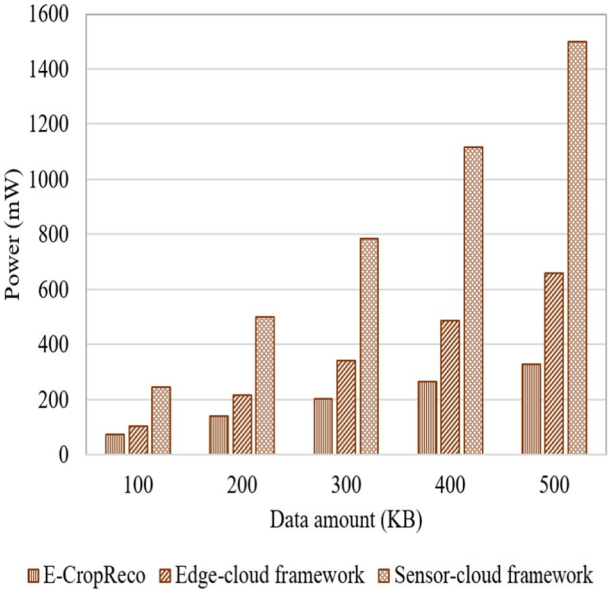


Fig. 5 Power consumption in proposed, edge-cloud, and sensor-cloud frameworks

respectively. The power consumption of sensor nodes is considered 5–100 mW/s, microcontroller is considered 4.2 mW/s (5 V, 3 A), mobile device, edge server, and cloud is considered 0.11 W/s (220 V, 1.8 A). Figure 4 presents a graphical comparison of the delays of E-CropReco framework, conventional edge-cloud framework, and sensor-cloud framework. Figure 5 presents a graphical comparison of the power consumption of E-CropReco framework, edge-cloud framework, and sensor-cloud framework. We observe that the proposed E-CropReco framework has approximately 60–70% lower delay and 70–80% lower power consumption compared to the sensor-cloud framework. We observe that the proposed E-CropReco framework has approximately 12–35% lower delay and 30–50% lower power consumption compared to the edge-cloud framework. Usually, most of the agricultural lands are located in the rural regions, and in the rural regions the network connectivity is poor. Hence, the sensor data transmission to the edge server or to the cloud (for processing and storage) is affected in conventional edge-cloud and sensor-cloud frameworks. But in E-CropReco, the dew layer temporarily holds the sensor data to prevent data loss, pre-processes the data, accumulates the data, and sends to the edge server for processing. After processing the data the edge server sends the result to the mobile device and to the cloud. By using dew computing and FL, E-CropReco outperforms the conventional edge-cloud and sensor-cloud frameworks in terms of delay and power consumption.

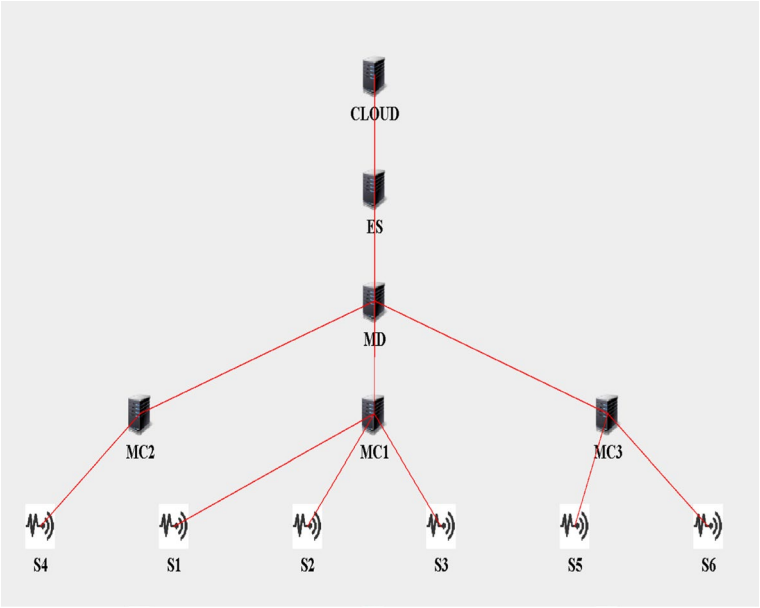
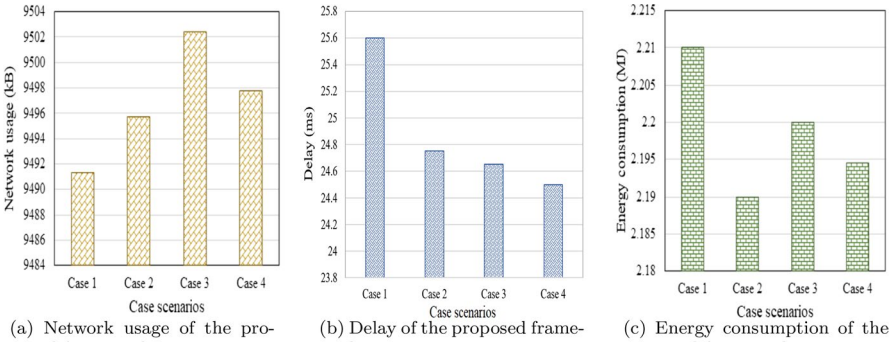


Fig. 6 Proposed topology implemented using iFogSim

4.2 Simulation using iFogSim

The simulation is performed to determine the network usage, delay, and the energy consumption of the proposed framework. To match with the real-world scenario: (i) the sensors' input values are provided based on the real dataset used in the experiment 4.3, the maximum and minimum sensor values are considered accordingly, (ii) uplink and downlink bandwidth in terms of data transmission speed are considered, (iii) various parameters are considered for the microcontrollers, mobile device, edge server, and cloud, such as RAM size and CPU length. To simulate the proposed framework, iFogSim [35] has been used. In Fig. 6, the proposed topology is presented. As observed, we have used three microcontrollers: MC1, MC2, and MC3. Under MC1, there are three sensor nodes (S1 (Nitrogen level measurement, maximum value: 140, minimum value: 0), S2 (Phosphorous level measurement, maximum value: 145, minimum value: 5), S3 (Potassium level measurement, maximum value: 205, minimum value: 15)). Under MC2, there is one sensor node (S4 (pH measurement, maximum value: 8, minimum value: 5)). Under MC3, there are two sensors nodes (S5 (temperature measurement, maximum value: 45, minimum value: 8), S6 (humidity measurement, maximum value: 100, minimum value: 45)). The microcontrollers (MC1, MC2, and MC3) are connected with the mobile device (MD). The mobile device is connected with the edge server (ES). The edge server is connected with the cloud. In the simulation, it is assumed that: (i) each microcontroller has 500 MB RAM, CPU length: 1000 MIPS, (ii) the mobile device has 1 GB RAM, CPU length: 3000 MIPS, (iii) the edge server has 4 GB RAM, CPU length: 40,000



(a) Network usage of the proposed framework

(b) Delay of the proposed framework

(c) Energy consumption of the proposed framework

Fig. 7 Results from iFogSim

Table 3 Case scenarios considered in simulation

Case scenarios	Uplink bandwidth	Downlink bandwidth
Case 1	1 Mbps	1.2 Mbps
Case 2	1.2 Mbps	1.5 Mbps
Case 3	1.5 Mbps	1.8 Mbps
Case 4	1.8 Mbps	2 Mbps

MIPS, and (iv) the cloud virtual machine (VM) has 4 GB RAM, CPU length: 44,800 MIPS. The idle power of mobile device, edge server, and cloud VM are 83.4333 W, 16 x 83.25 W, and 16 x 83.25 W, respectively. The busy power of mobile device, edge server, and cloud VM are 107.339 W, 16 x 103 W, and 16 x 103 W, respectively. We have considered four case scenarios (refer to Table 3). The uplink and downlink bandwidth (between mobile device and edge server, and between edge server and cloud) in terms of data transmission rate are presented in the table. The data processing delay is measured from the experimental analysis (here, we have considered a real dataset of 1000 samples, size: 151 KB) and added to measure the total delay. The network usage, delay, and the energy consumption of the proposed framework are presented in Fig. 7a–c, respectively. We observe that the network usage, delay, and energy consumption of the simulated framework are 9490–9504 kB, 24.4−25.6 ms, and 2.19−2.21 MJ, respectively.

Comparison between the theoretical and simulation results: In the theoretical analysis, we have measured the total delay and power consumption of the E-CropReco framework. In the simulation, we have measured the network usage, delay, and energy consumption of the framework. The theoretical analysis shows that the total delay of the E-CropReco framework for analysing 100–500 KB data is ~ 20–40 ms. The simulation results present that the delay is ~ 24–26 ms (dataset contains

Table 4 Statistical summary of the considered dataset

Statistical measure	N	P	K	Temperature	Humidity	pH	Rainfall
Number of entries	1000	1000	1000	1000	1000	1000	1000
Minimum	0	5	15	8.83	45.02	4.51	35.04
Maximum	140	145	205	43.68	99.98	7.99	298.56
Mean	60.96	54.37	52.98	26.63	79.14	6.4	117.51
Standard deviation	38.54	33.73	50.41	5.16	12.69	0.63	64.84

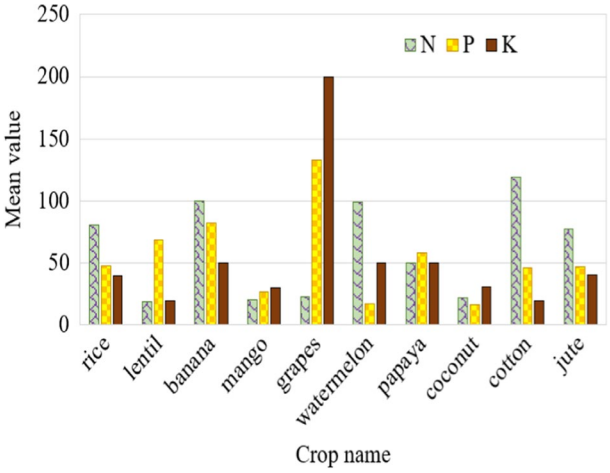


Fig. 8 N, P, K requirements of different crops

1000 samples, size: 151 KB). As observed, the delay values obtained from the simulation and theoretical results are quite similar.

4.3 Experimental analysis

For experimental analysis, we have considered 1000 samples from the real dataset,[1] and prepared our input dataset. We have applied four ML methods (SVM, LDA, D-Tree, and KNN) on the input dataset to observe their efficacy in crop yield prediction. We have performed the experimental analysis using MATLABR2022b. In this experiment, soil, environmental, and weather parameters (N, P, K levels of the soil, temperature, humidity, pH, and rainfall), and crop are taken in the input dataset. Table 4 presents the statistical summary of the dataset of 1000 samples. The requirements of N, P, and K for different crops are presented in Fig. 8. This is observed that the level of N is highest for cotton, whereas the levels of P and K are highest for grapes. The requirements of pH and rainfall

[1] https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset.

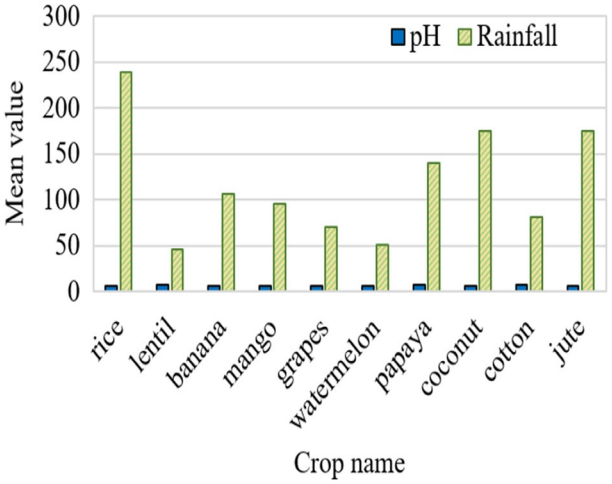


Fig. 9 pH and rainfall requirements of different crops

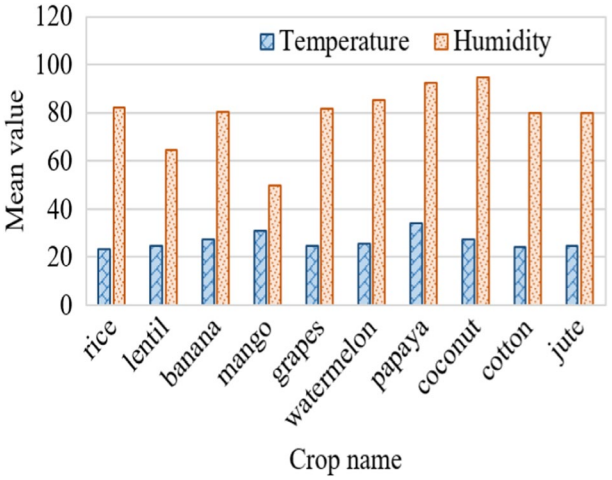


Fig. 10 Temperature and humidity requirements of different crops

for different crops are presented in Fig. 9. The results present that the pH level in case of lentil is highest, whereas the rainfall is highest for rice. The requirements of temperature and humidity for different crops are presented in Fig. 10. The results present that humidity is highest for coconut and temperature is highest for papaya.

From the dataset of 1000 samples, 800 samples are taken as training dataset and 200 samples are taken as test dataset. The test dataset's actual and predicted class labels are compared to determine the classification accuracy. We have considered the test cases by reducing the number of features, but we observe that considering all the features we are achieving the highest accuracy. If we consider

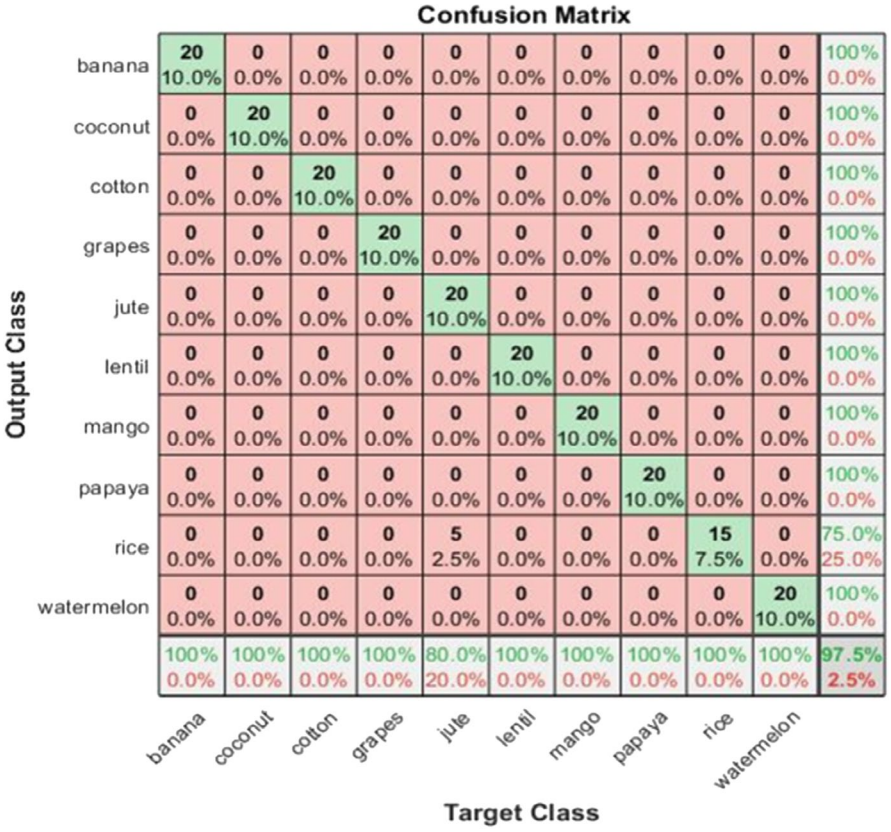


Fig. 11 Confusion matrix obtained using SVM classifier

only pH and rainfall, the SVM, KNN, LDA, and D-Tree achieve 68%, 67.5%, 60.5%, and 74% accuracy, respectively. If we consider only temperature and humidity, the SVM, KNN, LDA, and D-Tree achieve 66%, 73.5%, 61.5%, and 70.5% accuracy, respectively. If we consider only N, P, K, the SVM, KNN, LDA, and D-Tree achieve 81.5%, 88.5%, 82%, and 85.5% accuracy, respectively. When we consider all seven features, the SVM, KNN, LDA, and D-Tree achieve 97.5%, 96.5%, 96.5%, and 98% accuracy, respectively. Thus, it is observed that considering all features better prediction accuracy is achieved.

According to the classifiers, the obtained prediction accuracy of ten different classes (banana, coconut, cotton, grapes, jute, lentil, mango, papaya, rice, and watermelon) considering all seven features, are presented along with the respective confusion matrix. Each entry in a confusion matrix represents the number of predictions made by the model, which correctly or incorrectly classify the classes.

Figure 11 shows the comparative accuracy obtained for each of the different classes according to the training features used in SVM classifier. Figure 12 illustrates the comparative accuracy achieved for each of the different classes according to the

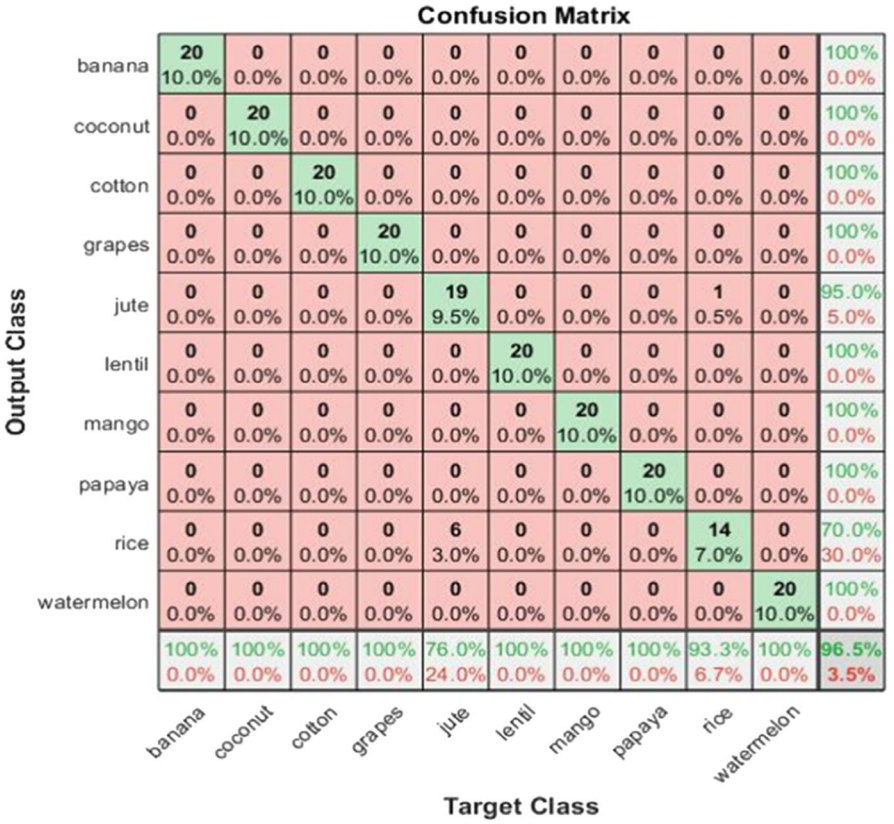


Fig. 12 Confusion matrix obtained using KNN classifier

training features used in KNN classifier. Figure 13 presents the comparative accuracy obtained for each of the different classes based on the training features used in LDA classifier. Figure 14 illustrates the comparative accuracy obtained for each of the different classes according to the training features used in D-Tree classifier.

To compare the performance of SVM, KNN, LDA, and D-Tree classifiers in crop productivity prediction, we consider accuracy, precision, recall, and *F*-score, as the performance metrics.

Based on the confusion metrics (refer Figs. 11, 12, 13, and 14), we have generated Fig. 15, that presents the accuracy achieved by the four classifiers for different crops. Table 5 presents the accuracy, precision, recall, and *F*-score obtained by each of the four classifiers considering all crops in the test dataset. The graphical representation of the performance metrics for the four classifiers is demonstrated in Fig. 16. We observe that each classifier has above 95% prediction accuracy, precision, recall, and *F*-score. We observe that D-Tree has higher accuracy, recall, precision, and *F*-score (98%) compared to the other three methods. From the results, we conclude that considering all performance metrics (accuracy, precision, recall, and

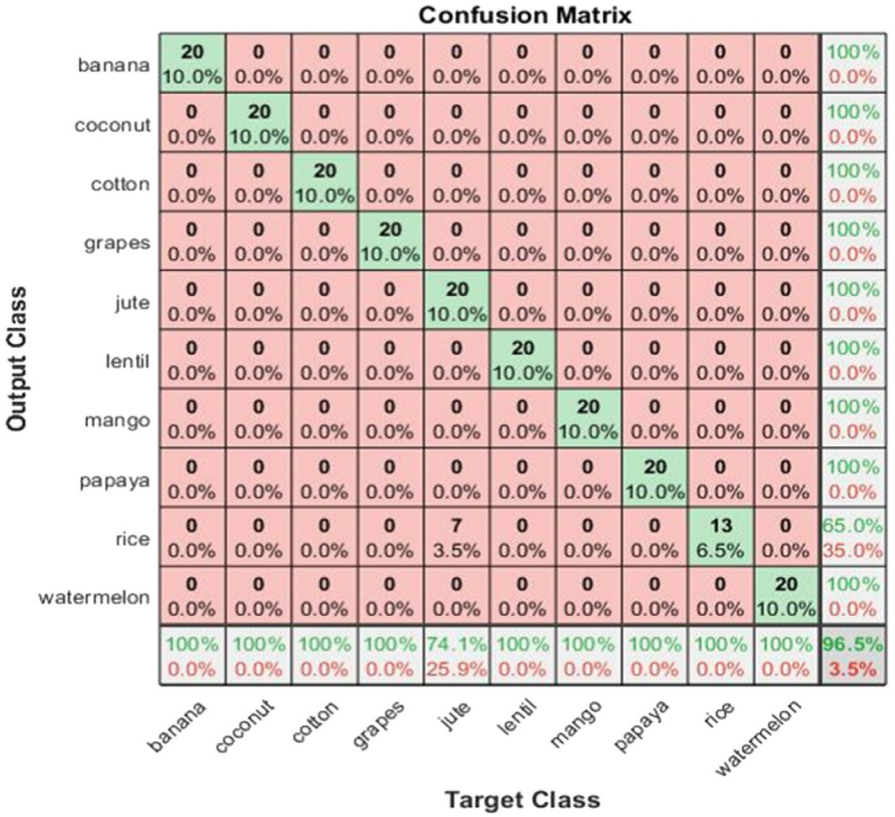


Fig. 13 Confusion matrix obtained using LDA classifier

F-score) and all crops, D-Tree performs better in E-CropReco than the other methods. As the input dataset used in the experiment has only 1000 samples, we have generated a large data set containing 50,000 samples also to evaluate the performance. The accuracy, precision, recall, and *F*-score measures in analysing the large dataset by using the four classifiers are presented in Fig. 17. We observe that for large dataset SVM and D-Tree outperforms KNN and LDA. However, the accuracy for each classifier is above 95%.

4.3.1 Comparison with existing approaches

Table 6 draws a comparison among the existing and proposed frameworks based on the accuracy, precision, recall, and *F*-score. This is observed that in [33], five ML algorithms were used for crop data analysis, and among them random forest (RF) achieved highest accuracy of 97.18%. In [34], KNN was used for crop data analysis, and achieved accuracy of 92.62%. In our E-CropReco, we have used four ML algorithms for crop data analysis, and among them D-Tree has achieved highest accuracy

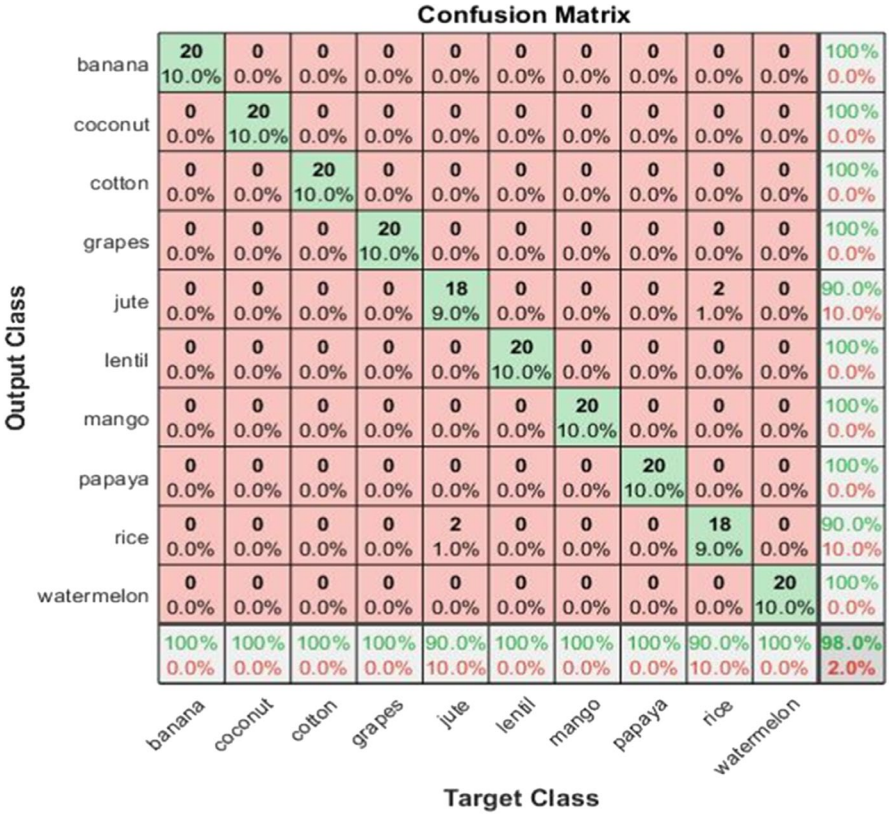


Fig. 14 Confusion matrix obtained using D-Tree classifier

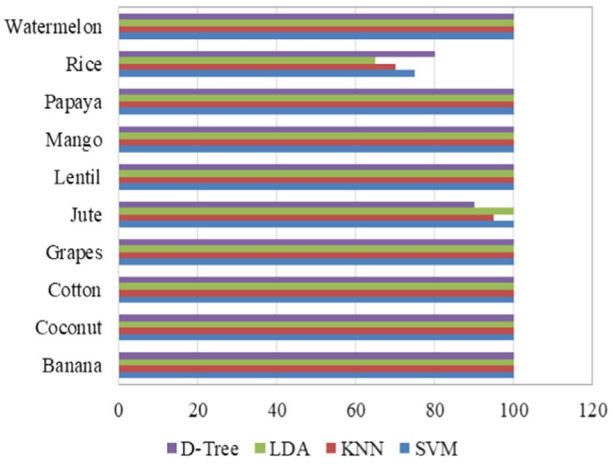


Fig. 15 Accuracy obtained by the classifiers for different crops

Table 5 Comparison of four classifiers based on accuracy, precision, recall, and *F*-score

Classifier	Accuracy	Precision	Recall	*F*-score
SVM	97.5%	98%	97.5%	97.46%
KNN	96.5%	96.93%	96.5%	96.44%
LDA	96.5%	97.41%	96.5%	96.39%
D-Tree	98%	98%	98%	98%

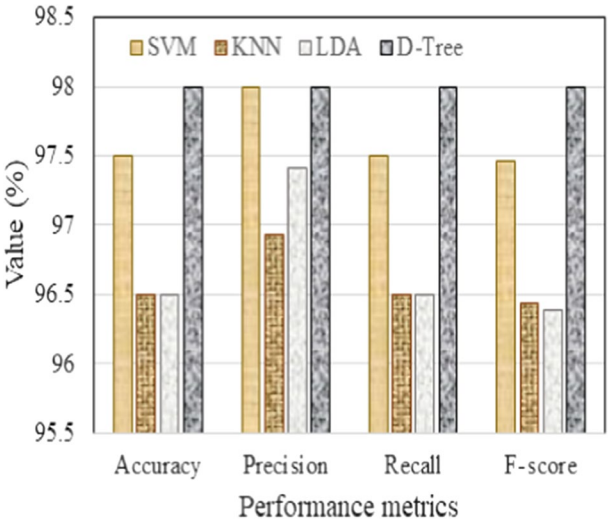


Fig. 16 Comparison of four classifiers based on accuracy, precision, recall, and *F*-score, in analysing the real dataset containing 1000 samples

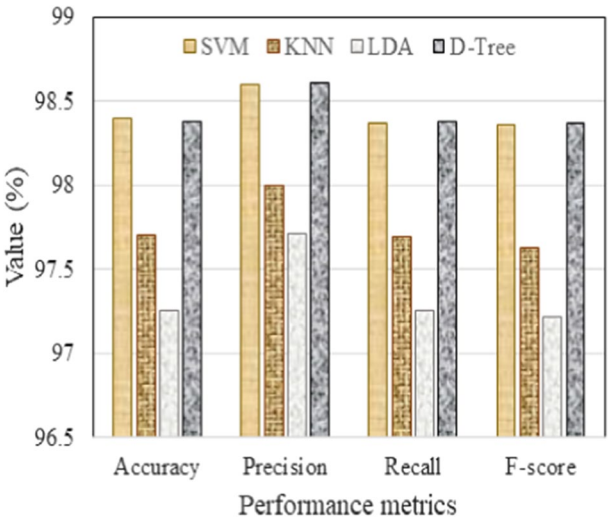
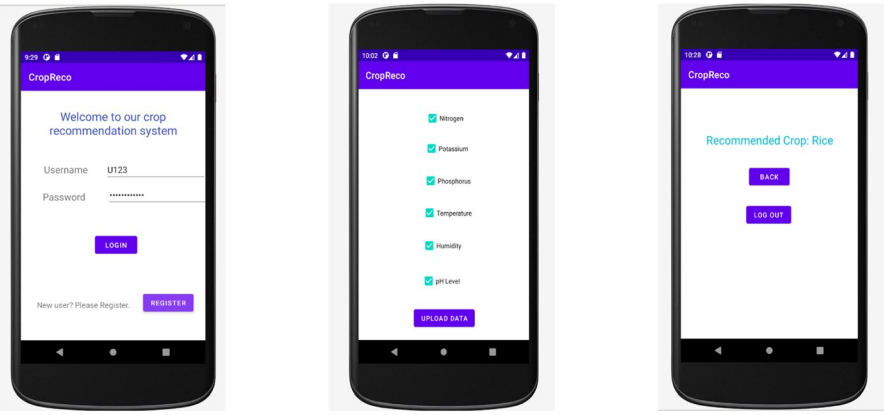


Fig. 17 Comparison of four classifiers based on accuracy, precision, recall, and *F*-score, in analysing the dataset containing 50,000 samples

Table 6 Comparison among the proposed and existing frameworks based on accuracy, precision, recall, and *F*-score

Work	Classifier	Accuracy	Precision	Recall	*F*-score
Thilakarathne et al. [33]	RF (Highest), KNN, D-Tree, Extreme Gradient Boosting (XGBoost), SVM	97.18%	97%	97%	97%
Cruz et al. [34]	KNN	92.62%	96.74%	92.62%	95.46%
E-CropReco	D-Tree (highest), SVM, KNN, LDA	98%	98%	98%	98%



(a) Screenshot of CropReco: User login

(b) Screenshot of CropReco: Upload data

(c) Screenshot of CropReco: Display Recommendation

Fig. 18 Screenshots of CropReco

of 98%. We also observe that the proposed framework has outperformed the existing approaches in terms of precision, recall, and *F*-score also.

4.4 CropReco: an app for crop recommendation

The objective of the proposed framework E-CropReco is to help the farmers to select the suitable crop for their land that will be beneficial in terms of their economical profit. Therefore, a user interface or mobile app is required using which the farmer can upload the data related to the land (sensor data), and receive recommendation. For this purpose, we have proposed an Android app named CropReco. Figure 18 shows the screenshots of the app. The mobile device works as the dew layer in our framework, and it has the received sensor data. Using the proposed app, the farmer can login or register if he/she is a new user (refer to Fig. 18a). After successful login, the farmer can select the checkbox for uploading the data from the mobile device (refer to Fig. 18b). The uploaded data and the weather data (stored inside

the edge server) will be analysed, and the recommendation will be generated by the edge server. The recommendation will be displayed on the screen (refer to Fig. 18c). Using this app, the farmer can receive the guidance regarding which crop he/she should plant depending on the soil data, environmental data, and weather data.

5 Conclusions and future work

The application of IoT in agriculture has gained the attention of the researchers in last few years. From the economical perspective agriculture is a vital sector of a country. Suitable crop identification for land plays a significant role in the production. Crop productivity prediction and recommendation are highly required for better productivity. In this paper, we propose a dew computing, edge computing, and FL-based IoAT framework that analyses the soil parameters, environmental parameters, and weather data, for predicting suitable crop for land. In the framework, the data regarding the soil parameters and environmental parameters, are collected using sensor nodes and sent to the mobile device. The mobile device working as the dew layer pre-processes and accumulates the sensor data, and forwards it to the edge server. The edge server holds the weather data of the region, where it is located. After analysing the weather data and the received sensor data using ML, the edge server sends the result to the cloud along with the model characteristics and to the mobile device. Four ML algorithms are used in the data analysis, and compared based on the performance metrics: precision, recall, accuracy, and *F*-score. We observe from the results that D-Tree outperforms SVM, KNN, and LDA, though each of the classifiers has more than 95% prediction accuracy. We have simulated the proposed architecture in iFogSim, and analysed its performance in terms of network usage, delay, and energy consumption. Theoretical results present that the proposed E-CropReco framework has reduced the delay by 60–70% and power consumption by 70–80% approximately than the conventional sensor-cloud framework. It is also observed from the results that the proposed E-CropReco framework has reduced the delay by 12–35% and power consumption by 30–50% approximately than the edge-cloud framework. We have also proposed an Android application for crop recommendation in this paper. In the future, we wish to extend our work for region-based crop productivity prediction. In the future, we also wish to use deep learning in the proposed E-CropReco framework. The use of blockchain is another future scope of this work.

Author Contributions SB contributed to conceptualization, formal analysis, methodology, and writing—original draft. TD contributed to conceptualization, methodology, data analysis, and writing—original draft. AM contributed to conceptualization, methodology, supervision, and writing—review and editing. RB contributed to supervision and writing— review and editing.

Availability of data and materials The datasets analysed during the current study are available from the corresponding author on request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Funding Not applicable.

References

1. Gavrilović N, Mishra A (2021) Software architecture of the internet of things (iot) for smart city, healthcare and agriculture: analysis and improvement directions. J Ambient Intell Humaniz Comput 12(1):1315–1336
2. Hitesh Mohapatra and Amiya Kumar Rath (2022) Ioe based framework for smart agriculture. J Ambient Intell Humaniz Comput 13(1):407–424
3. Javaid N (2021) Integration of context awareness in internet of agricultural things. ICT Express
4. Kalaiarasi E, Anbarasi A (2021) Crop yield prediction using multi-parametric deep neural networks. Indian J Sci Technol 14(2):131–140
5. Crane-Droesch A(2018) Machine learning methods for crop yield prediction and climate change impact assessment in agriculture. Environ Res Lett 13(11):114003
6. Abbas F, Afzaal H, Farooque AA, Tang S (2020) Crop yield prediction through proximal sensing and machine learning algorithms. Agronomy 10(7):1046
7. Pandith V, Kour H, Singh S, Manhas J, Sharma V (2020) Performance evaluation of machine learning techniques for mustard crop yield prediction from soil analysis. J Sci Res 64(2):394–398
8. Rishi G, Kumar SA, Oorja G, Krishna M, Shahreen K, Zirawani B, Hairulnizam M, Mostafa SA (2021) Wb-cpi: weather based crop prediction in india using big data analytics. IEEE Access 9:137869–137885
9. Narkhede UP, Adhiya KP (2014) Evaluation of modified k-means clustering algorithm in crop prediction. Int J Adv Comput Res 4(3):799
10. Gbadamosi B, Abidemi EA, Roseline OO, Bukola BO, Ehiedu PA (2019) Impact of climatic change on agricultural product yield using k-means and multiple linear regressions. Int J Educ Manag Eng (IJEME) 9(3):16–26
11. Elavarasan D, Vincent DP (2020) Crop yield prediction using deep reinforcement learning model for sustainable agrarian applications. IEEE access 8:86886–86901
12. Jayakumar D, Srinivasan S, Prithi P, Vemula Sreelekha, Sri Narashena (2021) Application of machine learning on crop yield prediction in agriculture enforcement. Revista Geintec-gestao Inovacao e Tecnologias 11(2):2142–2155
13. Zheng Z, Yan P, Chen Y, Cai J, Zhu F (2021) Increasing crop yield using agriculture sensing data in smart plant factory. In Security, Privacy, and Anonymity in Computation, Communication, and Storage: SpaCCS 2020 International Workshops, Nanjing, China, December 18-20, 2020, Proceedings 13, pp 345–356. Springer
14. Wang AX, Tran C, Desai N, Lobell D, Ermon S (2018) Deep transfer learning for crop yield prediction with remote sensing data. In: Proceedings of the 1st ACM Sigcas Conference on Computing and Sustainable Societies, pp 1–5
15. Chen Y, Zhou M, Zheng Z, Chen D (2019) Time-aware smart object recommendation in social internet of things. IEEE Internet Things J 7(3):2014–2027
16. Chen Y, Zhang J, Guo M, Cao J (2017) Learning user preference from heterogeneous information for store-type recommendation. IEEE Trans Serv Comput 13(6):1100–1114
17. Guillén MA, Llanes A, Imbernón B, Martínez-España R, Bueno-Crespo A, Cano J-C, Cecilia JM (2021) Performance evaluation of edge-computing platforms for the prediction of low temperatures in agriculture using deep learning. J Supercomput 77(1):818–840
18. O'Grady MJ, Langton D, O'Hare GMP (2019) Edge computing: a tractable model for smart agriculture? Artif Intell Agric 3:42–51
19. Kalyani Y, Collier R (2021) A systematic survey on the role of cloud, fog, and edge computing combination in smart agriculture. Sensors 21(17):5922
20. Ferrández-Pastor FJ, García-Chamizo JM, Nieto-Hidalgo M, Mora-Martínez J (2018) Precision agriculture design method using a distributed computing architecture on internet of things context. Sensors 18(6):1731
21. Zhang R, Li X (2021) Edge computing driven data sensing strategy in the entire crop lifecycle for smart agriculture. Sensors 21(22):7502

22. Farooq Muhammad Shoaib, Riaz Shamyla, Abid Adnan, Abid Kamran, Naeem Muhammad Azhar (2019) A survey on the role of iot in agriculture for the implementation of smart farming. IEEE Access 7:156237–156271
23. Ananthi N, Divya J, Divya M, Janani V (2017) IoT based smart soil monitoring system for agricultural production. In 2017 IEEE technological innovations in ICT for agriculture and rural development (TIAR), pp 209–214. IEEE
24. Ahmed N, De D, Hussain I (2018) Internet of things (IoT) for smart precision agriculture and farming in rural areas. IEEE Internet Things J 5(6):4890–4899
25. Partha Pratim Ray (2017) An introduction to dew computing: definition, concept and implications. IEEE Access 6:723–737
26. Rana S, Obaidat MS, Mishra D, Mishra A, Rao SY (2022) Efficient design of an authenticated key agreement protocol for dew-assisted IoT systems. J Supercomput 78(3):3696–3714
27. Javadzadeh G, Rahmani AM, Kamarposhti MS (2022) Mathematical model for the scheduling of real-time applications in IoT using dew computing. J Supercomput 78(5):7464–7488
28. Hati S, De D, Mukherjee A (2022) Dewbcity: blockchain network-based dew-cloud modeling for distributed and decentralized smart cities. J Supercomput 78(6):8977–8997
29. Manocha A, Bhatia M, Kumar G (2021) Dew computing-inspired health-meteorological factor analysis for early prediction of bronchial asthma. J Netw Comput Appl 179:102995
30. Ghosh S, De D (2022) Dewcitygame: dew computing-based 5g iot for smart city using coalition formation game. IETE J Res, pp 1–10
31. Li T, Sahu AK, Talwalkar A, Smith V (2020) Federated learning: challenges, methods, and future directions. IEEE Signal Process Magaz 37(3):50–60
32. Kumar P, Gupta GP, Tripathi R (2021) Deep privacy-encoding based federated learning framework for smart agriculture. IEEE Micro, Pefl
33. Thilakarathne NN, Bakar MSA, Abas PE, Yassin H (2022) A cloud enabled crop recommendation platform for machine learning-driven precision farming. Sensors 22(16):6299
34. Cruz M, Mafra S, Teixeira E (2022) An iot crop recommendation system with k-nn and lora for precision farming
35. Gupta H, Vahid DA, Ghosh SK, Buyya R (2017) ifogsim: a toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. Softw Practice Exp 47(9):1275–1296

Authors and Affiliations

Somnath Bera[1] · Tanushree Dey[1] · Anwesha Mukherjee[1] · Rajkumar Buyya[2]

Somnath Bera
somcs.research@gmail.com

Tanushree Dey
tanushreedeycs1987@gmail.com

Rajkumar Buyya
rbuyya@unimelb.edu.au

[1] Department of Computer Science, Mahishadal Raj College, Mahishadal, West Bengal, India

[2] Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia