A Blockchain-Assisted Trusted Federated Learning for Smart Agriculture

Article *in* SN Computer Science · February 2025

DOI: 10.1007/s42979-025-03672-4

3 authors, including:

Manoj Taleka
Manipal Academy of Higher Education
13 PUBLICATIONS **200** CITATIONS
SEE PROFILE

Narendra V G
Manipal Academy of Higher Education
37 PUBLICATIONS **571** CITATIONS
SEE PROFILE

SN Computer Science (2025) 6:221
https://doi.org/10.1007/s42979-025-03672-4

ORIGINAL RESEARCH

A Blockchain-Assisted Trusted Federated Learning for Smart Agriculture

T Manoj[1] · Krishnamoorthi Makkithaya[1] · V G Narendra[1]

Received: 30 May 2024 / Accepted: 5 January 2025

Abstract

Smart agriculture promises to alleviate the burden of climate risks on crop production by leveraging machine learning tasks. These tasks act as a decision support instrument for making well-informed choices by stakeholders in the agricultural value chain. Currently, predictive models in smart agriculture demand a centralized collection of diverse data, fragmented across multiple information systems leading to a single point of failure. The application of the Federated Learning (FL) technique restricts the movement of raw data and trains the model at the data source. However, the FL approach does not ensure trust factors like privacy, authentication, data provenance, transparency and traceability. To address this, a decentralized federated learning framework built on blockchain can be a potential solution. In this study, we introduce a blockchain-based framework called AgriFLChain for trusted federated learning in the context of smart agriculture. We focus on crop yield prediction as an illustrative use case, initially discussing centralized deep learning models (ResNet-16, ResNet-28, CNN-DNN, and CNN-LSTM). We then detail the authentication and data provenance mechanisms for federated learning participants, utilizing blockchain-based Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs). We implement these models using vanilla federated learning and Differential Privacy (DP) federated learning approaches, achieving transparency and traceability through smart contracts by recording metadata of model updates into the blockchain. Finally, detailed evaluation demonstrates that AgriFLChain achieves comparable efficiency to centralized models while maintaining scalability in blockchain transactions for higher data volumes.

Keywords Blockchain · Crop yield prediction · Deep learning · Federated learning · Smart agriculture · Smart contracts

Introduction

Agriculture plays a pivotal role in the economy, guaranteeing food security and a source of livelihood for the global population. Climate change risks such as rising temperatures, heavy floods, greenhouse gas emissions and frequent variability in weather events pose significant challenges to global food production, impeding its capability to keep up with growing demand. A United Nations (UN) report found that around 30 percent of the world's population suffers from moderate-to-severe food insecurity [1]. According to World Bank data, approximately 1.6 billion people in low and middle-income countries depend on food production to earn their livelihoods [2]. It is necessary to double the crop production output by 2050 compared to 2009 to meet the food security needs of the spiraling global population [3]. Smart agriculture has immense potential to address these challenges by exploiting large volumes of data generated through tasks such as crop yield prediction, monitoring, disease detection, weed identification, and price prediction. Crop yield prediction is the predominant and challenging activity in smart agriculture, as it contributes to maintaining a region's food security and aids farmers in boosting their agricultural productivity. Generally, crop yield is affected by numerous risks from sowing to harvesting. The influence of risk events on crop yield can be classified as catastrophic

T Manoj and V G Narendra have contributed equally to this work.

✉ Krishnamoorthi Makkithaya
k.moorthi@manipal.edu

T Manoj
t.manoj@manipal.edu

V G Narendra
narendra.vg@manipal.edu

[1] Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka, India 576104

Published online: 24 February 2025

events (droughts, floods, cyclones, hailstorms, earthquakes) with far reaching consequences and non-catastrophic events (weather fluctuations, soil conditions, pest infestation) with ephemeral impacts [4]. An accurate yield prediction model is essential for global and regional decision-making to effectively tide over short-term and long-term implications of risks.

Currently, economically developing countries adopt manual methods like Crop Cutting Experiments (CCE) to assess the crop yield in a particular region. It assists insurers and financial institutions in offering risk coverage or extending loans to farmers while governments use the data for policy planning [5]. However, CCEs are time consuming and labor intensive because they depend on field staff, administrative facilities, farmers' coordination and harvest conditions [6]. In recent years, with the upsurge in smart agricultural practices, many predictive models based on machine and deep learning have become available for crop yield prediction. However, yield prediction is determined by various factors such as weather, soil, crop management, crop phenotype, and crop genotype [7]. Hence, conventional machine learning techniques are required to collect spatio-temporally varying agricultural data from diverse backgrounds and concentrate on centralized systems to train prediction models. However, centralized learning suffers from two key trust issues. First, owing to data privacy restrictions and regulations, agricultural data remains siloed in organization servers and must be shared to achieve the common good. Second, security risks emerge due to single point of failure, identity theft, data falsification and unauthorized access [8]. Moreover, the absence of trusted data sharing mechanisms, associated overhead costs and lack of incentive schemes make farmers reluctant to share their farm data [9].

A novel training approach called Federated Learning (FL) has opened a new avenue in machine intelligence, which trains the model with decentralized data under the supervision of a centralized server. Specifically, it allows the geographically distributed organizations/entities to train the model collaboratively without sharing raw data with a centralized server. It is a suitable method for training yield prediction models as diverse data is siloed with stakeholders such as meteorological departments, Agricultural Technology Providers (ATPs), crop monitoring devices in farmlands, and soil testing laboratories. However, simple federated learning still needs to be improved with the limitations of a single point of failure and loss of privacy as a centralized server overlooks the training and aggregation of model parameters. Besides this, multiple security issues like data poisoning, man-in-the-middle attacks and denial of service attacks may hamper the application of federated learning in many scenarios. A blockchain-assisted decentralized federated learning offers a promising solution to overcome the pitfalls associated with centralized processing, [10]. Here, the key idea is to replace the existing many-to-one communication between clients and server with peer-to-peer communication between clients and aggregator, which is a part of a decentralized network. Blockchain is a decentralized peer-to-peer network that writes the transactions into an immutable and transparent distributed ledger upon consensus among the participants. The application of blockchain in federated learning helps achieve trust factors such as privacy, authentication, data provenance, transparency, traceability and anonymity [11]. Hence, this study proposes a framework called AgriFLChain, which effectively combines blockchain-based Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs), Differential Privacy (DP), smart contracts and blockchain to realize trustworthy federated learning for smart agriculture with the following contributions:

- Initially, we describe deep learning models like ResNet-16, ResNet-28, CNN-DNN, CNN-LSTM in centralized settings for crop yield prediction using the soybean yield dataset.
- We illustrate a method for authentication and data provenance of client organizations in a federated learning ecosystem using blockchain-based DIDs and VCs.
- We implement deep learning models for crop yield prediction using vanilla and DP-federated learning schemes.
- We design a smart contract to publish the metadata associated with federated model updates into the blockchain.
- We conduct a detailed performance evaluation of federated learning models and smart contract transactions' scalability.

The remainder of the article is organized into the following sections: Sect. Background discusses the background to the research study with a preliminary introduction to federated learning, blockchain, differential privacy concepts, and a detailed subsection on related work; Sect. Methods offers an overview of the AgriFLChain framework, description of the dataset, feature selection, deep learning models and algorithms for vanilla federated learning and blockchain-based DP-federated learning; Sect. Experimental Setup and Implementation Details provides a detailed experimental setup and implementation details; Sect. Results and Analysis provides the results and analysis of centralized and federated learning models in

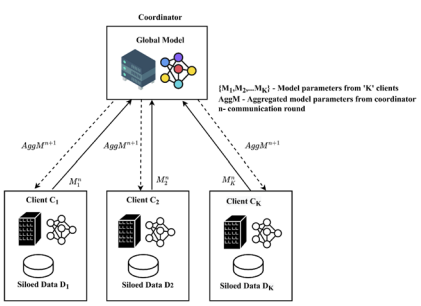


Fig. 1 An overview of federated learning

different settings; Sect. Discussion presents the discussion of the current study and Sect. Conclusion concludes the article with limitations and future work.

Background

Preliminaries

Federated Learning

A team led by McMahan et al., [12] coined the term Federated Learning for a distributed machine learning paradigm in Google. This is a learning approach in which the collection of client agents collaboratively trains the model with distributed data under the supervision of a centralized server. These client agents do not share their training data with any centralized server or trusted third parties. Model training occurs at the client's end, with parameter updates sent to the centralized server or coordinator for aggregation. Upon receiving the updates from the client agent's, the coordinator merges the model updates into a common global model and redistributes aggregated updates back to clients as illustrated in Fig. 1. Early applications of federated learning were seen in mobile and edge devices, which have now made a foray into mainstream scenarios where data is spread across devices, silos, and geographical boundaries. Federated learning techniques are characterized by their focus on several key aspects: the type of available data distribution, the privacy-preserving mechanisms used for model update exchange and aggregation, the choice of the learning model, efficient methods for communicating weights, and the degree of heterogeneity among contributing devices. Based on the distribution of data samples and their features, federated learning is broadly classified into horizontal federated learning, vertical federated learning and federated transfer learning [13].

In the case of horizontal federated learning, different client devices possess similar feature sets but differ in their samples. Vertical federated learning is suitable where client devices share a similar sample space but vary in their feature sets. For scenarios in which the feature set and client device samples do not overlap, federated transfer learning is preferred. Frequently employed techniques for privacy-based sharing of model parameters during distributed learning include DP, Secure Multi-party Computation (SMPC) and Homomorphic Encryption (HE). Prevailing machine learning algorithms defined on the principles of linear models, tree-based and neural network models are more appropriate to be deployed in the federated setting. Conventional machine learning models are inappropriate for training in a distributed manner as the communication efficiency of updates may significantly hinder performance. Federated Averaging (FedAvg) is the widely adopted optimization method to achieve communication efficiency by sharing updates. This method averages the decreasing gradient value at the global level, and the updated average value is distributed back to the training local models [12]. In addition, better optimization techniques such as FedProx, QFedAvg and FedOptim have been proposed to accommodate heterogeneity, fairness and server-side efficiency factors. The variation in the device's capability and data generated leads to the inefficiency of the training process. Asynchronous communication of updates, sub-sampling of devices, and devising fault-tolerant mechanisms are possible solutions to system heterogeneity.

Blockchain for Enabling Trust Factors

Blockchain is a fully decentralized and disintermediated network in which participating peers collaborate to reach a consensus and record the transactions into an immutable, transparent, traceable distributed ledger. The primary motivation behind conceiving a blockchain was to achieve decentralized financial transactions with anonymity. The first significant application of blockchain was seen in the transaction of a cryptocurrency named Bitcoin, introduced by Satoshi Nakamoto [14]. The introduction of programming and scripting into blockchain led to a notable expansion of its horizon into multifarious sectors, including governance, education, healthcare, supply chains, agriculture, logistics, entertainment, and more [15]. The synergy of blockchain features and cryptographic primitives makes it appropriate for enabling trust factors in transactions related to federated learning [16]. An agent-controlled identity and credential system are necessary to ensure that only authenticated agents (devices/organizations) become part of the federated setup and share model updates during learning. Recent advancements in digital identity techniques have led to the creation of blockchain-based Self-Sovereign Identity (SSI) which is promoted by the World Wide Web Consortium (W3C), Trust Over Internet Protocol (TOIP) and the Sovrin foundation. In SSI, an agent controls its identity-related data and decides which data to share with others without relying on centralized systems. With SSI, anyone can verify the claims regarding identity with cryptographic validity. SSI encompasses an ecosystem in which the identifier is decentralized, secure and human-readable, satisfying the needs of Zooko's triangle [17]. SSI meets identity principles such as user-controllability, portability, interoperability, privacy and minimal disclosure, persistence, decentralization, transparency, authenticity and verifiability as elucidated by Christoper Allen [18]. The true potential of SSI is realized for digital identity by adding a missing identity layer to the existing Internet Protocol (IP) architecture. Two essential layers of

the SSI stack, namely DIDs and VCs, form the basis for authentication and data provenance [19].

DIDs act as identifiers in the SSI environment and are used to authenticate users/agents by using blockchain as a verifiable identity registry for a decentralized source of trust. A DID is a user-controlled, cryptographically verifiable and decentralized universal identifier that can be assigned to any human, device or organization [20]. The syntax of DID is similar to that of a Uniform Resource Identifier (URI) comprising a DID schema, method and method specific string. In general, DID document encompass fields like global identifier, authenticity proof, digital signature, service endpoints and metadata. Based on the context of its usage DIDs can be public DIDs (visible and registered on blockchain) or private/peer DIDs (non-disclosed and used in one-to-one relationship). VCs are tamper-proof digital attestation for the claims about identity owner that is verifiable cryptographically without any intervention from trusted-third parties [21]. The VCs data model standardized by W3C can be used to assert data provenance from various devices or organizations in a multi-stakeholder federated environment. The issue-hold-verify lifecycle in the VCs data model includes an issuer assigning a credential to the claims of the subject, a holder is a subject about whom the claims are made, and a verifier receiving the proof of credential for claims about the subject. Smart contracts are the application layer component of blockchain that hosts self-executing code (digital contract), which is triggered upon satisfying the predetermined conditions agreed by transacting agents. Transactions conducted by smart contracts are immutably recorded on a distributed ledger of the blockchain and validated by a consensus mechanism. Hence, smart contracts are an obvious choice for communicating updates in federated training process [22]. All participants on the blockchain have visibility to smart contract code, making sharing model updates infeasible. However, integrating suitable privacy-preserving mechanisms with smart contracts can help to realize a secure and privacy-based sharing of model updates in decentralized learning. Frequently used privacy protecting approaches such as DP [23] and SMPC are successfully implemented in smart contracts to share local model updates and incentivize client devices' contribution [24]. In this way, blockchain aids in enabling trust factors such as authentication, data provenance, privacy and availability for federated learning.

Differential Privacy for Sharing Model Parameters

Federated learning circumvents the need to share raw data from the participating devices and organizations. However, it does not guarantee the privacy of model updates during training. It is required to believe that all the transacting agents (clients and coordinator) distrust each other and share computed updates without any possibility of inferring something about actual data to ensure the privacy of model parameters. It can be attained through a technique called DP, which was initially proposed by [25]. The underlying idea behind DP uses a randomized mechanism to add noise/perturbation to actual output data or model parameters so that the original data and differentially added noisy data become indistinguishable, and privacy is preserved. The commonly used perturbation mechanisms in DP are Laplace and Gaussian mechanisms. DP specifies how much information about a dataset can be made available to other parties for analysis. Conventionally, these limits are defined with the help of ϵ (eplison) and δ(delta). ϵ represents a privacy budget offering insight into privacy provided by a DP mechanism. δ denotes the probability of the mechanism which fails to render the required privacy guarantees. The smaller value of ϵ obtained for a dataset results in stronger privacy with little scope for its utility. Hence, the privacy versus utility trade-off must be carefully managed based on the application domain in which the dataset is being used [26]. Generally, DP can be divided into Global Differential Privacy (GDP) and Local Differential Privacy (LDP). In GDP, the perturbation is added to data after it is collected from the source where the users trust the data curator. However, in LDP, it is assumed that data curators are not trustworthy and noise is added to the data before analysis. The seminal work conducted by [27] established the suitability of using the DP framework for training deep learning models. In the recent past, the DP-FedAvg algorithm by [28] demonstrated that DP is well-suited for federated learning.

Related Work

The advent of Information and Communication Technologies (ICT) in agricultural activities has transformed conventional farming into smart farming. Smart farming plays a critical role in leveraging the effectiveness of agricultural practices to increase crop productivity, improve food quality, manage production risks, reduce costs, and control greenhouse gas emissions. In particular, crop yield prediction is a significant decision tool that aids farmers in making informed decisions on crop selection in a particular season, helping governments and policymakers to plan their food security needs, buyers to fix a fair and remunerative price for crops, insurers and bankers to estimate and compensate losses to farmers due to natural calamities. To facilitate decision-making, statistical approaches and crop simulation models were extensively utilized for yield prediction [29]. Early studies used Copula-based statistical measures, probability distributions, and crop health monitoring to estimate crop yield losses and overcome production risks [30, 31]. However, inferential statistical methods are suitable for drawing inferences regarding probability or likelihood measures rather than generalizing predictive patterns [32]. For better performance in prediction models, researchers have

started to embrace machine learning which enables models to learn from data, forecast the future and make decisions.

Machine Learning and Deep Learning for Yield Prediction

In the literature, the prediction of corn, rice, wheat, soybean, and cotton crop yield using machine learning has been exhaustively studied. Emphasis is placed on modeling weather, soil and crop management (sowing time, fertilizer input, crop height, crop maturity, plant growth, and irrigation) features obtained from remote sensing data, meteorological data, in situ measurements, and agricultural IoT devices [7]. Most of the existing studies have widely used variants of regression techniques and Artificial Neural Networks (ANN) for yield estimation. Furthermore, ensemble learning (bagging and boosting), and Support Vector Machine (SVM) are frequently used to forecast the yield. Research works by [33] and [34] applied linear regression to predict paddy and soybean yields, whereas [35] utilized Quantile Regression (QR) for wheat yield forecasting for risk estimation. Multiple studies have successfully applied ANN [36, 37], ensemble methods [38, 39] and SVM [40, 41] for the prediction of major crops like paddy, wheat and corn. The success of deep learning in different domains, coupled with the availability of computational resources, has paved the way for its use in yield prediction. Neural networks such as Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM), Reinforcement Learning (RL) and hybrid learning approaches have been widely adopted for yield estimation. Khaki et al., [42] developed a DNN based on residual shortcuts that included weather, soil and crop genotype in a feature set to predict the maize yield performance. Kim et al., [43] studied the effect of extreme weather events on corn yield and proposed a DNN-based model for its prediction using meteorological data. A model for winter wheat yield prediction was proposed based on a one-dimensional CNN [44], and its performance was compared with eight different supervised machine learning models used as baselines. For corn and soybean yield estimation, a CNN-RNN framework was designed by [45] which incorporates one-dimensional CNN to extract relevant weather and soil features and an RNN to capture the temporal dependencies of yield over a period of time. To effectively model the long-term dependencies of yield, Jiang et al., [46] applied LSTM to county-level corn yield prediction. Elavarasan et al., [47] integrated the traditional Random Forest (RF) technique with RL for paddy yield prediction to enhance crop productivity and address agrarian application-based issues. However, these efficient prediction models currently adopt a centralized architecture for data collection and training, which is vulnerable to a single point of failure and data breaches that violate data privacy concerns. In addition, data sharing in the agricultural sector is less extensive than in healthcare and genomic research [48]. These security, privacy and data sharing shortcomings lead to a lack of trust in centralized models, severely limiting their application in agricultural decision-making processes.

Federated Learning for Smart Agricultural Tasks

The adoption of federated learning in various domains such as healthcare and genomics, smartphones, advertising, finance, autonomous vehicles, and manufacturing is on the rise, but its application in smart agriculture is still in its infancy. Durrant et al., [49] successfully demonstrated the application of cross-silo federated learning to soybean yield prediction with differential privacy. They trained CNN and CNN-RNN models with remote sensing data and the LSTM model with a tabular dataset for yield prediction using FedAvg and Federated Batch Normalization (FedBN) methods to aggregate model weights. The results showed an improvement of 22.75 percent and 39.81 percent in Root Mean Square Error (RMSE) values for the image and tabular data modalities, respectively, compared to local baselines. In the context of smart farming, [50] proposed an FL model that combines Partial Least Square (PLS) regression with neural networks for milk quality prediction. Kumar et al., [51] observed that security and data privacy are major impediments to the implementation of IoT systems in smart agriculture. Hence, they proposed a privacy-preserving FL framework for smart agriculture that encompasses modules for achieving privacy (perturbation-based encoding, LSTM-AutoEncoder (AE) transformation) and detects intrusion in IoT devices with Fed-GatedRecurrentUnit (GRU). This framework was experimentally evaluated using federated methods and compared to non-federated techniques such as Naïve Bayes (NB), decision trees, and RF to check its effectiveness in detecting intrusion. The extensive performance evaluation revealed that federated techniques exhibited superior performance compared to their non-federated counterparts.

To protect and secure agricultural IoT devices from cyber-attacks, a federated intrusion detection framework called FIELDS was developed using real-time network traffic datasets [52]. Within the FIELDS framework, deep learning models, namely DNN, CNN, and RNN, were developed and deployed in federated settings for intrusion detection using real-world network traffic datasets such as MQTTset, InSDN and CSE-CIC-IDS2018. The comprehensive evaluation of the FIELDS framework using classification metrics including Precision, Recall and F_1 score showed that deep learning classifiers trained in a federated manner performed comparably to and, in some instances, outperformed centralized machine learning models. Patros et al., [53] integrated serverless computing with federated learning to leverage

rural Artificial Intelligence (AI) for weed detection using the FedAvg algorithm. The evaluation of the full participatory federated model based on ANN showed improved accuracy compared to the “local-only" model. However, it exhibited relatively poorer performance compared to the “cloud-only" model. Zhang et al., [54] proposed a federated framework for smart breeding solutions incorporated an RF model with vertical data partitioning for maize yield prediction using tabular phenological and weather data. Confidentiality was achieved through one-way encryption schemes to ensure data security within the framework. The experimentation of the model involved three different settings: centralized data, individual site data, and federated approach. Their corresponding results were compared using Leave-One-Year-Out (LOYO) cross-validation values of the RMSE, revealing that the conventional RF model displayed reduced performance in centralized and individual site settings compared to the federated approach. Li et al., [55] developed a communication efficient and privacy-preserved federated pruning approach for yield forecasting and data sharing. Specifically, they aggregated the pruned and localized model updates for soybean yield prediction to enhance the model’s prediction accuracy and enhance inference efficiency.

Idoje et al., [56] investigated the role of federated learning in crop classification tasks using a Gaussian NB model with the FedAvg algorithm. They tested Stochastic Gradient Descent (SGD) and Adam optimizers during experimentation with learning rate combinations of 0.01 and 0.001. Their results concluded that SGD optimization led to faster model convergence after 100 epochs and displayed better accuracy than centralized models. Aggarwal et al., [57] designed a privacy-protected and resource efficient federated learning approach integrated with the Internet of Agriculture Things (IoAT) for rice-leaf disease detection. Two key techniques namely federated transfer learning and federated feature extraction were experimented and compared on performance metrics such as accuracy and resource utilization (GPU, CPU, and memory). EfficientNetB3 was identified as the best model with a validation accuracy of 99 percent. Devraj et al., [58] introduced a hierarchical federated learning framework for crop health monitoring of tomato plantation. The framework integrates IoT sensors that collect soil, crop, and weather data, and distributes machine learning computation across local devices, edge nodes, and cloud servers. The hierarchical learning approach allows for privacy-preserved and adaptive learning across farms enabling efficient model aggregation and decision making.

While the aforementioned studies on federated learning in smart agriculture have decentralized data processing to preserve privacy, they still rely on centralized model aggregation, which introduces significant security and privacy risks. This centralization also lacks transparency and traceability, making it difficult to track model contributions and updates. To address some of these limitations, Qammar et al. [59] proposed integrating blockchain with federated learning to enable decentralized, secure model updates. Arachchige et al. [60] developed a framework combining federated learning, blockchain, smart contracts, and privacy-preserving mechanisms to enhance trust and security in industrial IoT systems. Shen et al., [61] formulated a blockchain assisted federated learning framework for rice pest and disease detection. To ensure transparency and privacy the framework incorporates homomorphic encryption and secret sharing schemes for model update sharing while training SVM model. The experimental results demonstrate that federated training of SVM achieves accuracy of 93 percent in pest detection with privacy protection. Nie et al., [62] designed a blockchain enabled federated system for distributed multi-regional jujube orchard yield estimation. It utilizes adaptive aggregation algorithm to prevent malicious attacks with underlying blockchain for transparency and traceability of model updates. However, while these studies show promise in improving transparency and privacy, they do not fully address authentication, data provenance, and the trustworthiness of participating organizations in the federated learning process. Furthermore, the mechanisms for verifying the authenticity and integrity of model updates remain underdeveloped. Table 1 summarizes some notable studies conducted with respect to federated learning in smart agriculture for regression tasks and compares it with centralized learning with their limitations in meeting required trust factors.

Methods

AgriFLChain Framework

This section proposes a trusted and reliable federated learning framework named AgriFLChain assisted by blockchain for crop yield prediction task in smart agriculture as shown in Fig. 2. It ensures that participants in the federated setup are authenticated and that the model update process happens in a decentralized and privacy-preserved manner. The framework constitutes two layers: a data layer and blockchain layer. The primary components of the data layer include different organizational silos with agricultural data and a coordinator with a global model for learning from its federated members. A blockchain layer comprises two elements: a permissioned identity blockchain and public blockchain. The SSI-based identity blockchain achieves authentication and data provenance for federated actors with the help of DIDs and VCs. A public blockchain encompasses a smart contract that creates a transaction record in the blockchain for the traceability and transparency of the federated learning process. The stakeholders, such as agricultural government

Table 1 Summary of Centralized and Federated Learning Approaches for Regression Tasks in the Smart Agricultural Domain

Sl.No	Related work	Training mode	Description	Evaluation metrics	Security, privacy & Transparency support
1	Khaki et al., 2020 [45]	Centralized	Utilized CNN-RNN model for soybean yield prediction	RMSE, Correlation coefficient(r)	No
2.	Elavarasan et al., 2021 [47]	Centralized	Implemented reinforcement random forest for paddy yield prediction	Mean Squared Error(MSE), Mean Absolute Error(MAE), r, RMSE	No
3.	Srivastava et al. 2022 [44]	Centralized	Utilized one dimensional CNN model for winter wheat yield prediction	MSE, RMSE, MAE,r	No
4.	Vimalajeewa et al., 2021 [50]	Decentralized	Implemented NN-Partial Least Squares(PLS) model using federated learning for milk quality prediction	Residual Sum of Squares(RSS), R^2	Data Ownership, Privacy
5.	Durrant et al., 2022 [49]	Decentralized	Experimented CNN-RNN and LSTM models using federated learning for soybean yield prediction	RMSE	Privacy
6.	Zhang et al., 2023 [54]	Decentralized	Deployed federated random forest model for maize yield prediction	MSE, RMSE, MAE, R^2	Authentication, Privacy
7.	Li et al., 2024 [55]	Decentralized	Proposed federated pruning approach aggregating pruned and localized client model updates for soybean yield prediction	RMSE	Privacy

agencies, data organizations, planners, insurers, and bankers involved in the value chain for deriving insights from yield prediction are identified and included in the AgriFLChain framework. A government agency at the national level, like the Department of Agriculture and Farmers Welfare, can be a steward that defines a governance model for blockchain-enabled cross-silo federated learning. Stewards deploy valid actors as issuers (Government agricultural agencies/Farmers' organizations at the regional level) and verifiers (Agricultural policy planners/insurers/bankers) for DID and VC-based authentication and data provenance of Agricultural Data Organizations (ADOs) participating in the federated learning process. ADOs willing to participate in the federated process get the credential from the government agricultural agency/farmers' organization at the district or county level. The coordinator (planners/insurers/bankers) deploys a global model for crop yield prediction requests and verifies the proof of credential from ADOs before uploading the training model to local clients. The verified client organizations receive a machine learning model and train it with the local dataset with the DP algorithm to guarantee the privacy of the model parameters. Further, the model parameters exchanged between client organizations and coordinator are encrypted for secure aggregation. The metadata related to the model update, which exchanges local or global parameters during each communication round, is verified and added as a transaction into the public blockchain through a smart contract. The transfer of model updates between clients and the coordinator is iterated over multiple rounds until the global model converges to the least error.

Centralized Learning for Yield Prediction

Dataset Description

This research study utilizes the Soybean yield dataset collected from different counties across nine states in the United States (US) [45]. The dataset comprises of 25,345 rows of tabular data where weather, soil, and crop management features were observed from 1980 to 2018, totaling 392 features. The weather features, six in total (precipitation, vapor pressure, solar radiation, minimum temperature, maximum temperature, and snow water equivalent), were measured weekly throughout the year, resulting in 312 features (6 features × 52 weeks) and were acquired from the Daymet Service with a 1 km^2 resolution [63]. Soil component features, measured at six different depths (0–5, 5–15, 15–30, 30–60, 60–100, and 100–200 cm), include organic carbon density, organic carbon stock, soil organic carbon, clay, nitrogen, coarse fragments, pH in H_2O, cation exchange capacity at pH7, bulk density, silt, and sand. This results in 66 soil features (11 features × 6 depths), obtained from the SoilGrids250m database [64]. Furthermore, the dataset includes fourteen columns representing

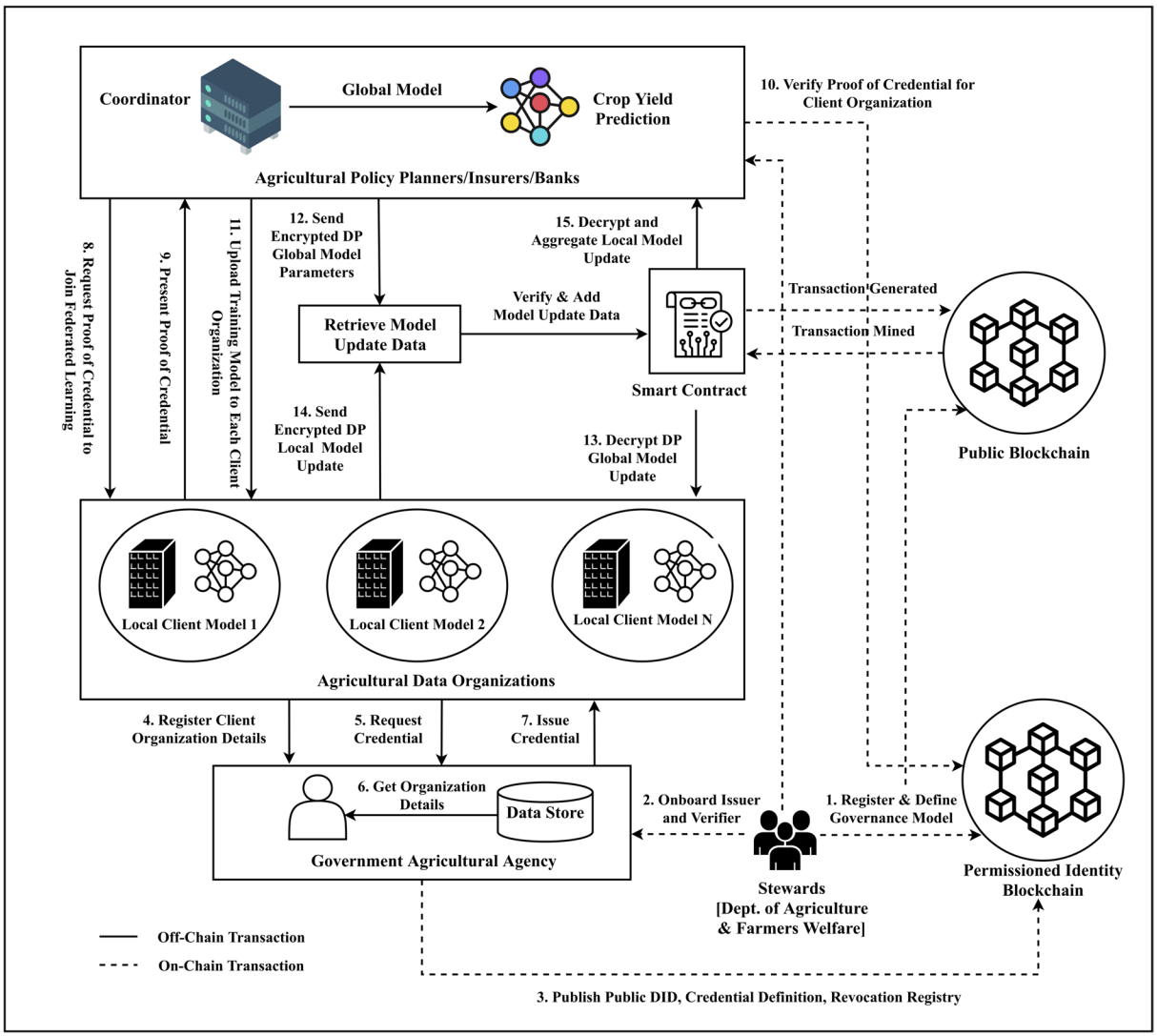


Fig. 2 An overview of AgriFLChain framework

the cumulative percentage of soybean sown every week from April each year as crop management features, gathered from the US National Agricultural Statistics Service (NASS) [65]. The target variable for the task of yield prediction is soybean yield expressed in bushels per acre.

Data Preprocessing and Feature Selection

The dataset used in this study is initially screened for missing values and outliers. Only a very small percentage of missing values (less than 1 percent of the total dataset) are observed, and these are imputed with zeros, which did not significantly affect the overall data distribution or prediction accuracy. All instances with a target yield value of less than five bushels are removed. In addition, the feature column labelled “Loc_ID" is dropped as it does not influence target yield. Most of the features have data values in different scales. Hence, all values in the feature set are rescaled using either StandardScaler or RobustScaler, depending upon the requirement. StandardScaler subtracts the mean value and transforms the feature values between 0 and 1. RobustScaler deducts the median and rescales the feature values by the Interquartile Range(IQR) so that it becomes robust to outliers. In the dataset of 392 features, not all features significantly contribute to yield prediction. Hence, feature selection is performed based on their significance in determining the yield value. Both filter-based and intrinsic methods are utilized for efficient feature selection. Specifically, correlation measure and mutual information score are selected as filter methods, while the XGBRegressor model is used to obtain feature importance as an intrinsic method.

Deep Learning Techniques

Deep learning techniques are extensively used for approximating continuous-valued non-linear functions in regression problems like yield prediction with a large and wide variety of data collected from multiple sources. The techniques derived from deep learning considered in this study include DNNs, CNNs, LSTMs and Residual Networks (ResNets). DNNs represent an advanced form of ANNs comprising multiple hidden layers in contrast to the single hidden layer found in traditional ANNs. These hidden layers incorporate numerous neurons equipped with non-linear activation functions that are capable of learning from underlying data. DNNs operate based on a feedforward network architecture, where information flows sequentially from the input layers through the hidden layers and ultimately reaches the output layer. At each hidden layer, the input data is subject to transformations involving the application of weights and biases before being passed to the subsequent layer. The training process for DNNs utilizes the backpropagation algorithm. The computed output is compared to the actual target values during forward propagation to calculate a loss. This loss is then iteratively minimized by applying various optimization techniques, effectively tuning the network's weights and biases [66]. CNNs are specialized DNNs designed to handle complex, non-linear data with fewer weights and biases. They are less susceptible to overfitting and particularly well-suited for data with grid-like structures, including time-series data (1D) and image data (2D). CNNs consist of distinctive layers, including convolutional layers, pooling layers, and fully connected layers, each dedicated to learning specific features within the data. The key characteristics of CNNs, such as sparse connectivity and weight sharing, contribute to a significant reduction in computational requirements compared to fully connected DNNs [67]. RNNs are an extension of DNNs designed for modeling sequential and time-series data with interdependent attribute values. They can handle variable-length input data by maintaining a recurrent hidden state that retains information from past elements in the sequence. However, RNNs face challenges in handling long-term dependencies due to the vanishing and exploding gradient problem. To address these challenges, a variant of RNNs called LSTM networks was introduced by Hochreiter and Schmidhuber in 1997. LSTMs incorporate a fundamental unit known as a cell state, which includes input, forget, and output gates. These gates control the flow of sequential information and maintain the necessary dependencies over extended periods [68]. With the increase in computational capabilities, a recent trend in deep learning involves training neural networks with hundreds of layers to better approximate and generalize data. One practical approach to achieving this is through specialized deep neural networks known as ResNets, which utilize skip connections to address the vanishing or exploding gradient problem [69]. Generally, the ResNets behave well for the problems involving the processing of images due to the presence of local convolutional kernels. However, convolution kernels are not suitable for approximating non-linear functions in regression. Based on the original form of ResNet, a neural network for non-linear regression is built by replacing convolutional kernel layers with fully connected layers in the residual block, as illustrated in [70].

Model Description

As deep learning models are good at estimating the non-linear functions with multiple input features, this study proposes to use ResNet-16, ResNet-28, CNN-DNN and CNN-LSTM for soybean yield prediction as shown in Fig. 3a–d. The ResNet model implemented for regression is obtained from the original implementation of ResNet that consists of an input layer connected with multiple identity blocks and an output layer [70]. Each of these identity blocks is made up of two dense layers with a residual shortcut for two hidden layers. The dense layer comprises a Rectified Linear Unit (ReLU) for activation and batch normalization for regularization instead of dropout. In the course of experimentation with deeper networks, it was determined that the most favorable approach for comparing regression models involved utilizing a depth of 16 layers and 28 layers, excluding the input layer. This configuration was found to strike an effective balance between minimizing loss and mitigating overfitting, as illustrated in Fig. 3a and b. The ResNet regression model, designed to meet the specified depth requirement, is constructed by sequentially stacking identity blocks, each containing 50 neurons in every layer. The proposed CNN-DNN model incorporates three stacked one-dimensional convolutional layers (Conv1D) to extract the temporal patterns in input features used for prediction as detailed in Fig. 3c. The Conv1D layer is followed by batch normalization, max pooling, dropout, five dense layers, and output with linear activation. The filters used in the first, second, and third Conv1D layers are 64,128 and 1024, respectively. Batch normalization is used to standardize the inputs from Conv1D layers and dropout for regularization. The first four fully connected dense layers comprise 128, 64, 768, and 32 units, followed by one unit in the last dense layer. As the target yield value used in this study is a progression of values over the years, prediction can be best formulated as a time-series forecasting problem. LSTMs are appropriate to capture the sequential dependencies among attribute values. Based on the idea presented in [45], a CNN-LSTM model is designed in the study that encompasses two-dimensional convolutional layers (Conv2D) to extract relevant features and LSTM for depicting temporal dependencies. In particular, CNN modules named W-Conv2D and S-Conv2D are defined to extract weather and soil features. To model the linear and non-linear impact of weather features, W-Conv2D is made up of four Conv2D layers with 8, 12, 16, and 20 filters and an average pooling layer placed between them to downsample the

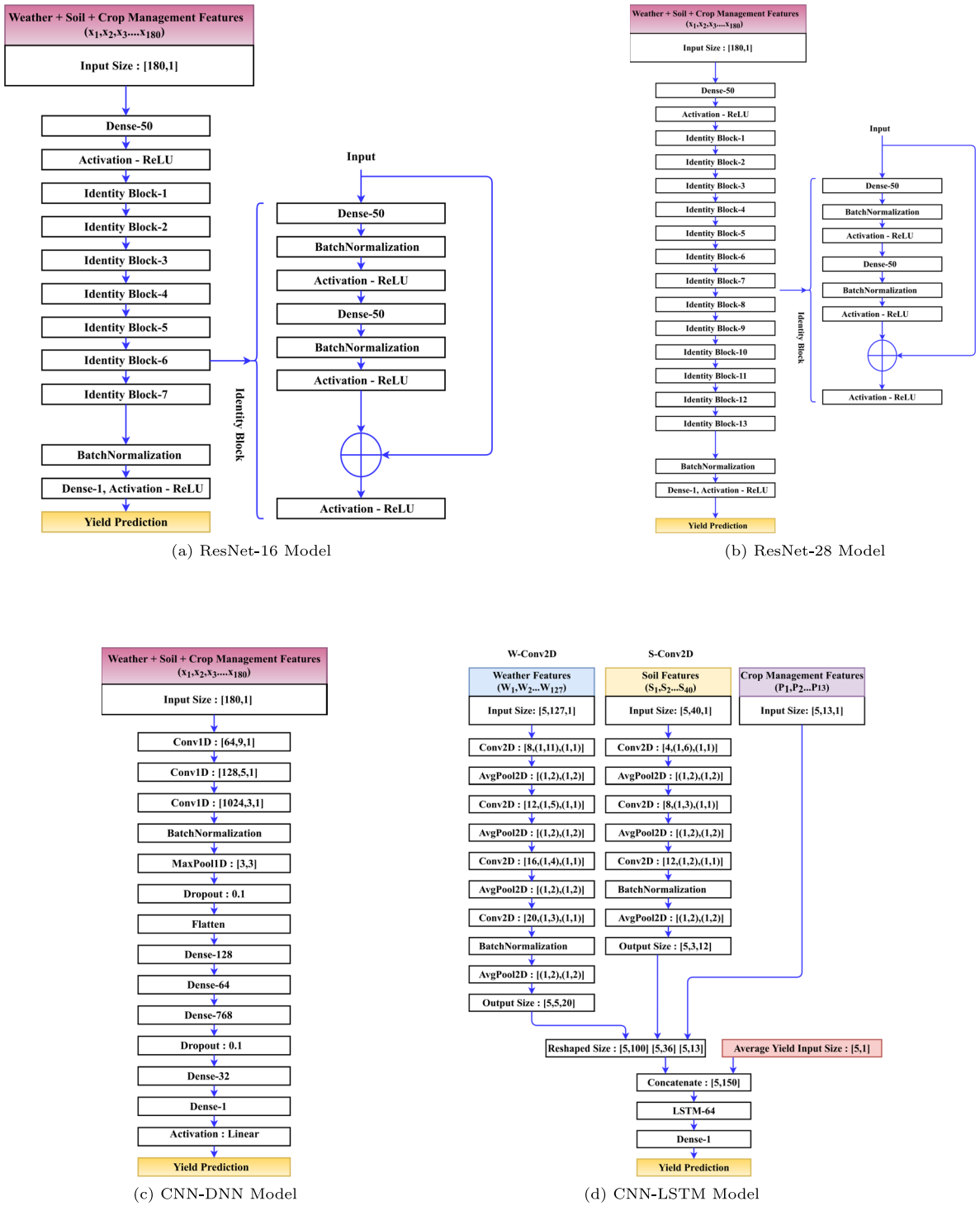


Fig. 3 Architecture Diagram for ResNet-16, ResNet-28, CNN-DNN and CNN-LSTM Models. Convolutional layer parameters are shown as "[No. of filters, Kernel Size, Stride Size]". Pool layer parameters are shown as "[Pool Size, Pool Stride]" Dense layer is represented as "Dense followed by No of Units"

representation size. In the end, the W-Conv2D is appended with the batch normalization layer. Similarly, S-Conv2D models the impact of soil features that encompass three Conv2D layers with 4,8,12 filters, average pooling, and batch normalization layers. The extracted features from W-Conv2D, S-Conv2D, crop management data, and average yield in a particular year are input to LSTM cells for prediction. LSTM layer consisting of 64 units predicts the value of crop yield for year 't' by using information from (t-n) years. In particular, t=5 is used in the model description shown in Fig. 3d.

SSI-Based DID and VC in Federated Learning

This section illustrates how federated learning employs SSI-based DIDs and VCs to achieve the issue-hold-verify trust triangle for authentication and data provenance of client organizations based on the method developed in our earlier work [71]. The present study has a scenario where an agricultural policy planner/insurer/bank (verifier) is interested in training its yield prediction model with data held by ADOs (holder) in a federated manner. ADOs become a part of a federated ecosystem after being authenticated by VCs issued by the local Government Agricultural Agency (issuer). Figure 4 illustrates the sequence of messages exchanged among the actors. Initially, the issuer publishes a DID, public keys, schema and credential definition into the identity blockchain. Any ADOs willing to request credential or present proof of credential establishes the connection with the issuer and verifier with the help of DIDs. ADOs agree with the issuer regarding the attributes to be a part of the credential. In addition, it generates a blinded master secret (m_b) and commits to attribute values it has shared with the issuer. The verifier generates a Presentation Definition (P_d) and writes it to the identity blockchain. ADOs retrieve P_d and prepare a data organization proof (P_r) for verification on receiving the presentation of proof request from the verifier. Finally, the verifier confirms the correctness of the proof using issuer public keys.

Vanilla Federated Learning for Yield Prediction

This study has employed the vanilla federated averaging scheme for predicting soybean yield. The data samples are horizontally distributed, ensuring that all independent siloed clients possess datasets with the same set of features. These data samples utilized for federated training are partitioned into Identical and Independently Distributed (IID) and Non-IID categories. Suppose a client contributing to federated training provides equal data samples. In that case, it results in IID partitioning, while an unequal number of data samples leads to Non-IID partitioning. The process used to train a model with the federated averaging method is adapted from [12] and is described in Algorithm 1. The

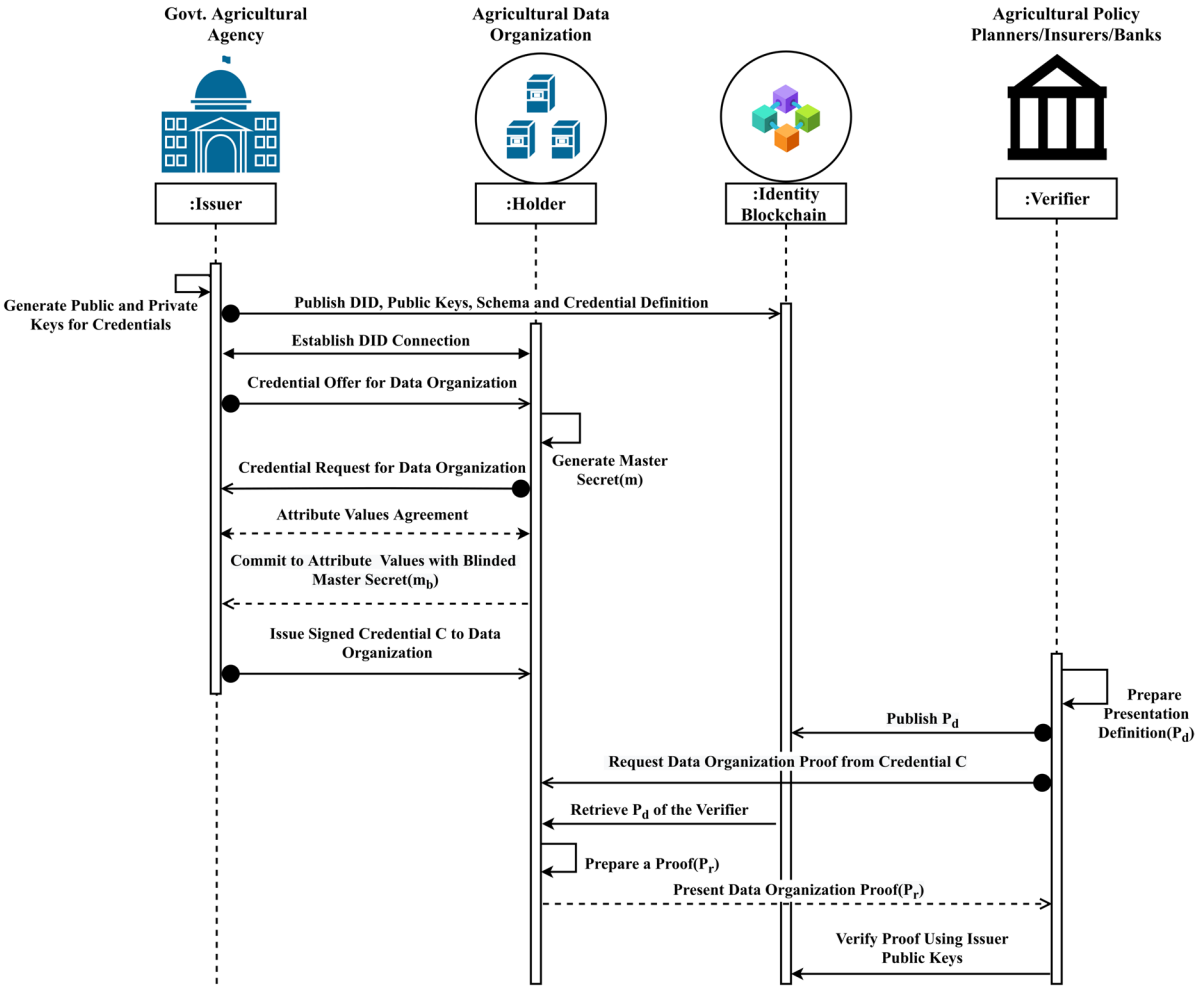


Fig. 4 Sequence diagram for authentication and data provenance of federated participants using DIDs and VCs

federated averaging algorithm is particularly well-suited for parameterized learning in deep learning, accommodating a diverse range of gradient descent optimization methods.

Algorithm 1 Federated Averaging Algorithm

Input: **N** denotes the total number of communication rounds, **K** signifies the overall count of ADO clients, **D** corresponds to the total number of data points, $\mathbf{D}_k$ corresponds to data points held by the $\mathbf{k}^{th}$ ADO client.
Output: Trained Global Model

1: Initialize the global model with the number of rounds (N), relevant metrics, hyperparameters, and weight configurations.

2: **for** n in N **do**

- Choose **K** clients from various ADOs and distribute the global model to each selected client k.
- Each client k initializes its local model with global weights and trains it using a data subset held by it.
- Calculate the scaling factor $\frac{D_k}{D}$ based on the proportion of data points in each client k and then adjust the local model's weights accordingly.
- Aggregate the local models by summing the scaled weights as defined by the equation.

$$f(w) = \sum_{k=1}^{K} \frac{D_k}{D} F_k(w)$$

where $F_k(w) = \frac{1}{D_k} \sum_i f_i(w)$

- Update the global model with the aggregated weights.

3: **end for**

Blockchain-Based Differentially Private Federated Learning

This section describes the workflow for federated training of ML models (neural networks) in different client organizations (ADOs) and aggregation of DP model updates to achieve the global prediction model using blockchain-enabled smart contracts. Algorithm 2 details the role of a client for DP-federated learning, and Algorithm 3 presents a coordinator's part in collecting and aggregating model updates using DP. The summary of the key notations used in the algorithms are given in Table 2. The main steps in the workflow include (i) Setup the Cryptographic parameters, (ii) Publish global model to clients with initial parameters, (iii) Local Model Training with DP, (iv) Creating a record/transaction of model updates using smart contracts (v) Collection and aggregation of DP local model updates to global model (vi) Redistribute global model updates to local training models. In the AgriFLChain framework, ADOs are authenticated and verified by the coordinator with the help of SSI-based verifiable credentials. Then, the coordinator generates a public-private key pair, creates a global ML model, encrypts initial parameters, and broadcasts it to all clients (ADOs). The clients also compute the public-private key pair required to exchange model parameters securely with the coordinator. Each client taking part in a particular round of federated training gets a copy of the model, initializes the model parameters, and trains it with the local dataset using DP. A record of local model updates is created in the blockchain using smart contracts, and an encrypted update is transmitted. The coordinator collects and decrypts DP local model updates from different clients and aggregates them into the global model using a federated averaging scheme.

Table 2 Summary of key notations

Notation	Description	Notation	Description	Notation	Description
B	Batch Size	C	Norm Bound	C_k^{PK}	Public Key of k^{th} client
C_k^{PK}	Secret Key of k^{th} client	CO^{PK}	Public key of coordinator	CO^{SK}	Secret key of coordinator
D	Size of entire dataset	D_k	Size of dataset held by k^{th} client	E	Number of Epochs
σ	Noise Scale	η	Learning Rate	$\mathcal{N}$	Gaussian Noise
ϵ	Privacy Budget	w_n^k	Weight of k^th client in n^th round	GM_{Update}	Global Model Update
LM_{Update}	Local Model Update				

Algorithm 2 Client Side Algorithm for Differentially Private Federated Learning

Input: C_1,C_2....C_K are credential authenticated ADO clients.
D_k=<x_i,y_i> - Input dataset for clients k=1,2....K.
N- No. of communication rounds, B- Batch Size
E- No. of Epochs,σ- Noise Scale, C- Norm Bound
η- Learning rate, GM_{Update}- Encrypted Global Model Update
Output: Encrypted Differentially Private Local Model Parameters($LM_{Update}^{C_k}$)

Generate a public-private key pairs. <C_k^{PK},C_k^{SK}> for kth client.
Broadcast the client public key(C_k^{PK}) to the coordinator.
Get the Machine Learning (ML) model for training into the client(C_k).
for round n in $1, 2, \ldots N$ **do**
Decrypt the global model update(GM_{Update}) published by the coordinator using corresponding private key (C_k^{SK}).
Set client-side ML model with GM_{Update} parameters.
Train the client-side ML model with DP(σ,C) using its local dataset $D_k = <x_i, y_i>$
for epoch e in $1, 2, \ldots E$ **do**
for each $<x_i, y_i>$ in Batch B **do**
Compute the gradient $g_e(x_i) = \nabla L(x_i; y_i, \hat{y_i})$
Clip the gradient $g_e(x_i)$ by factor 'C' using L2 norm

$$\overline{g_e}(x_i) = \frac{g_e(x_i)}{\max(1, \|g_e(x_i)\|_2/C)}$$

Add the noise with scale σ

$$\tilde{g_e}(x_i) = \overline{g_e}(x_i) + \mathcal{N}(0, \sigma^2 C^2 I)$$

where $\mathcal{N}(0, \sigma^2 C^2 I)$ is gaussian noise with mean 0 and covariance $\sigma^2 C^2 I$
end for
Compute the average of noisy gradients

$$\tilde{g_e} = \frac{1}{B} \sum_{i=1}^{B} \tilde{g_e}(x_i)$$

Update the weights $w_n^k = w_{n-1}^k - \eta * \widetilde{g_e}$
end for
Compute the privacy budget ϵ using the accounting method.
Retrieve the DP local model update($LM_{Update}^{C_k}$) from client C_k.
Invoke smart contract 'SC' that records $LM_{Update}^{C_k}$ metadata <ClientID,Timestamp,$LM_{updatesize}$,Hashvalue,Roundnumber> into the blockchain.
Encrypt $LM_{Update}^{C_k}$ using coordinator's public key (CO^{PK}) and publish it to coordinator.
end for

Algorithm 3 Coordinator Side Algorithm for Differentially Private Federated Learning

Input: Encrypted DP local model updates from different clients k = 1,2....K
N – No. of communication rounds
Output: Trained Global Model, Encrypted Global Model Parameters[GM_{Update}]

Generate a public-private key pairs $<CO^{PK}, CO^{SK}>$ required for communication.
Create a ML model and encrypt initial model parameters with respective client's public key (C_k^{PK}) and broadcast it to all clients.
for round n in $1, 2, \ldots N$ **do**
  for each ADO client in k in $1, 2, \ldots K$ **do**
    Decrypt the local model update($LM_{Update}^{C_k}$) using the coordinator's private key (CO^{SK}).
    Aggregate the $LM_{Update}^{C_k}$ for 'K' clients using federated averaging scheme

$$w_n = \sum_{k=1}^{K} \frac{D_k}{D} w_n^k$$

    Set the global model with aggregated local model weights(w_n).
    Retrieve the global model update(GM_{Update}).
  Invoke the smart contract 'SC' that records GM_{Update} metadata <CoordinatorID, Timestamp,$GM_{updatesize}$,Hashvalue,Rounndnumber> into the blockchain.
    Encrypt GM_{Update} using their respective public key (C_k^{PK}) and publish it to all clients.
  end for
end for

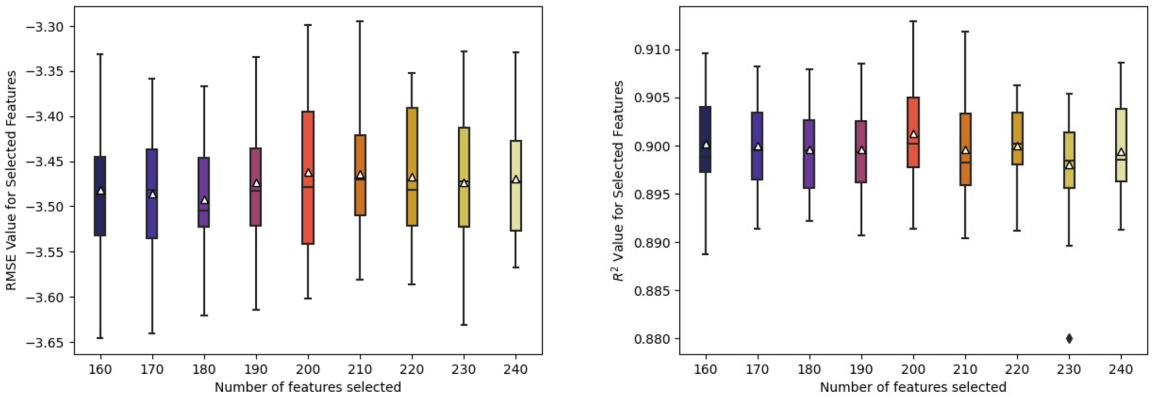
Experimental Setup and Implementation Details

This section outlines the details of the experimental setup used for training and evaluating deep learning models in a centralized and federated way. The study focuses on federated learning with blockchain to enhance transparency and privacy and provide a reliable prediction. All the experiments on training the models are carried out in a cloud server based on AMD EPYC 32-core processor with NVIDIA A100 40 GB GPU and Ubuntu 22.04 operating system. The primary programming language used for coding models is Python3, while Solidity is used for writing smart contracts. Libraries such as Pandas, NumPy, Scikit-learn, and Tensorflow 2.0 are employed to accomplish tasks such as data preprocessing and implementing deep learning models. The framework named Flower 1.3.0 is used to conduct the federated training via Google Remote Procedure Call (gRPC). PyNaCl and Tensorflow-Privacy libraries are utilized for asymmetric encryption and enabling differential privacy, respectively. A Python-based blockchain framework called Brownie is used for deploying smart contracts on Ethereum Ganache, a simulated public blockchain platform. To build and evaluate centralized and federated learning models for soybean yield prediction, we aggregate training data from 1980 up to the year preceding each prediction year. We have considered the test data from 2018, 2017, and 2016 as prediction years. Experiments involving vanilla and DP-federated learning are conducted using three to five client organizations, with data horizontally partitioned in IID and non-IID manner. Performance is evaluated using metrics including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R^2 score. This study uses RMSE as the key evaluation metric for comparing models experimented in different centralized and federated settings. Specifically, centralized (coordinator-side) and federated (client-side) evaluation metrics are computed for federated learning model evaluation. In centralized evaluation, the global model at the coordinator-side is evaluated using test data, while federated evaluation involves evaluating each client-side model with their respective test data.

Results and Analysis

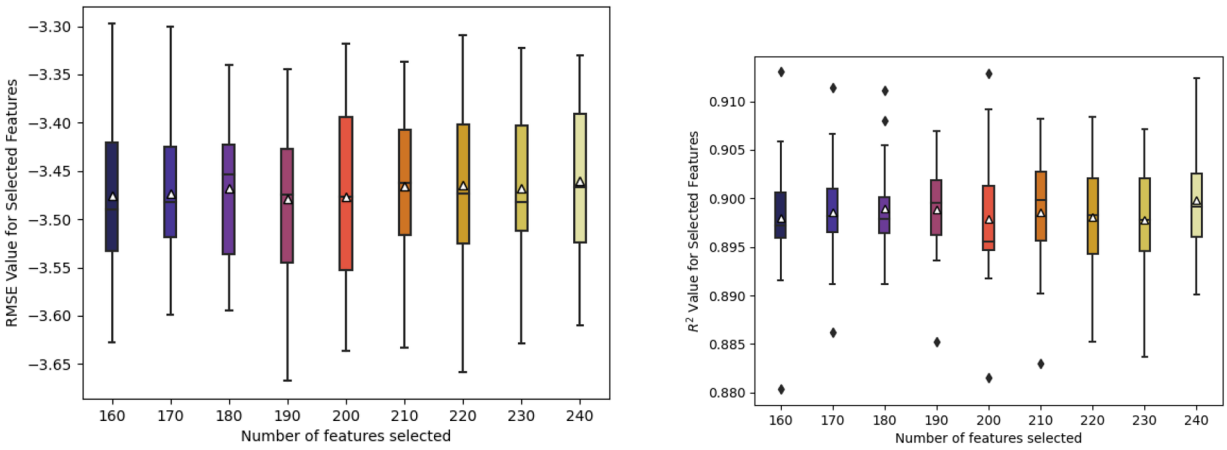
Feature Selection

The feature selection experiments are conducted with the use of filter-based methods that include the correlation measure and mutual information score, as well as an intrinsic method based on the XGBRegressor model. Initially, a broader range of features is tested using RMSE and R^2 scores as performance metrics. These experiments showed a significant drop in prediction performance when fewer than 160 or more than 240 features are used. Consequently, further experimentation investigated features within the 160 to 240 range, chosen from a total of 392. In filter-based methods, the 'n' best features selected by correlation measure and mutual information score are evaluated on the XGBRegressor model, which has been fine-tuned after parameter optimization through grid search. Furthermore, ten-fold cross-validation is carried out for different feature counts, and performance is assessed using two metrics: RMSE and R^2 score, with the results presented in Figures 5 and 6. For the intrinsic



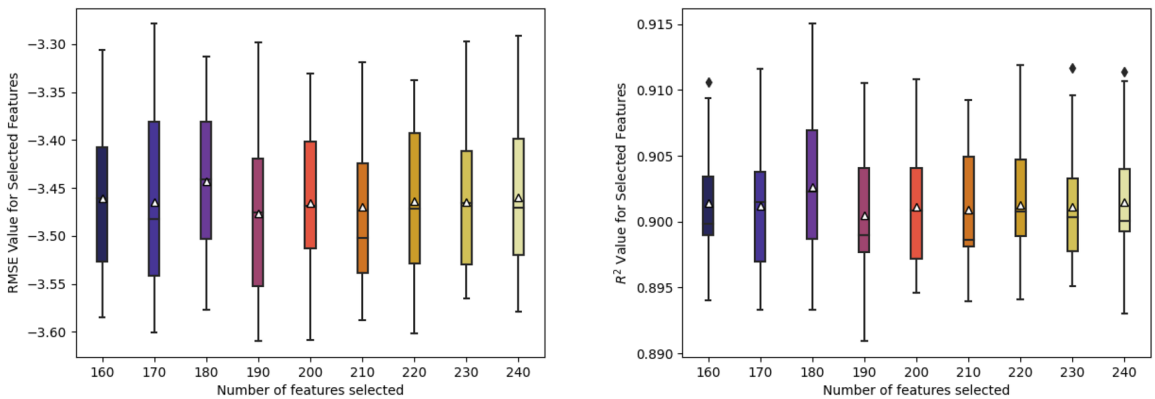
(a) RMSE Vs Selected Features (b) R^2 Score Vs Selected Features

Fig. 5 Box and Whisker Plots of RMSE and R^2 Score Selected Features Using Correlation Measure



(a) RMSE Vs Selected Features (b) R^2 Score Vs Selected Features

Fig. 6 Box and Whisker Plots of RMSE and R^2 Score Selected Features Using Mutual Information Score



(a) RMSE Vs Selected Features (b) R^2 Score Vs Selected Features

Fig. 7 Box and Whisker Plots of RMSE and R^2 Score Selected Features Using Intrinsic Method - XGBRegressor Model

method employing the XGBRegressor model, feature importance is calculated, and its performance is evaluated using RMSE and R^2 score, as shown in Fig. 7. Comparing the two feature selection methods, the intrinsic method using the XGBRegressor model demonstrates a favorable RMSE value of 3.446 with 180 features. This approach also exhibits excellent model interpretability as evidenced by a high R^2 score in Fig. 7. While the differences in RMSE and R^2 scores between the methods might appear minor, these small variations can have a significant impact in precision-driven task like yield prediction. Among the three methods tested, XGBRegressor consistently demonstrated the best performance in both RMSE and R^2 scores across different feature counts, particularly within the range of 160 to 240 features.

Centralized Learning With Deep Learning Models for Yield Prediction

The implementation and evaluation of deep learning models such as ResNet-16, ResNet-28, CNN-DNN, and CNN-LSTM are carried out for three different prediction years, 2018, 2017, and 2016 in centralized settings. Since hyperparameter selection is critical for effective model training and accuracy, we considered six key hyperparameters-weight initialization method, learning optimizer, learning rate (η), batch size (B), number of epochs (E), and activation function-based on an extensive literature review. To ensure the optimal performance, a grid search is employed to explore various configurations for these hyperparameters. For weight initialization methods like {'glorot uniform,' 'glorot normal,' 'he uniform,' 'he normal'} are considered, as they are known to perform well for deep learning models. Learning optimizers such as {'SGD,' 'Adam,' 'AdaGrad,' 'RMSProp'} are evaluated to determine their convergence speed and adaptability to noisy gradients. Learning rates of {0.01, 0.001, 0.0001} are tested to find the optimal balance between stability and speed. Various batch sizes {32, 64, 128, 256} are examined to balance computational efficiency and convergence. For epochs, multiple values are tested {60,100,150,200,250} to avoid overfitting or underfitting while allowing sufficient learning time. Different activation functions like {'ReLU,' 'Leaky ReLU,' 'Sigmoid', 'ELU'} are explored to capture non-linear relationships effectively in different architectures of ResNet-16, ResNet-28, CNN-DNN, and CNN-LSTM. These configurations are chosen to ensure thorough exploration and optimal model performance based on validation RMSE loss value. After tuning, the optimal hyperparameter values for weight initialization (glorot uniform), learning optimizer (Adam), η (0.0001), and activation function (ReLU) are consistent across all four models, while the B and E varied as shown in Table 3. Additionally, the number of filters, filter sizes, stride sizes, pool sizes, and dropout rates configured for CNN layers in CNN-DNN and CNN-LSTM models are depicted in Figs. 3c and 3d. The experimentation is carried out with deep learning models using the inferred hyperparameter values. The performance evaluation obtained for deep learning models in centralized learning is listed in Table 4. The analysis of RMSE performance metrics across prediction years 2018, 2017, and 2016 in centralized learning reveals that CNN-LSTM outperforms the other three models. Training the ResNet model with 28 layers does not perform better than ResNet-16. Therefore, three models-namely, ResNet-16, CNN-DNN, and

Table 3 Hyperparameter values for centralized deep learning models

Model	Batch size	Epochs
ResNet-16	64	120
ResNet-28	128	150
CNN-DNN	128	200
CNN-LSTM	64	180

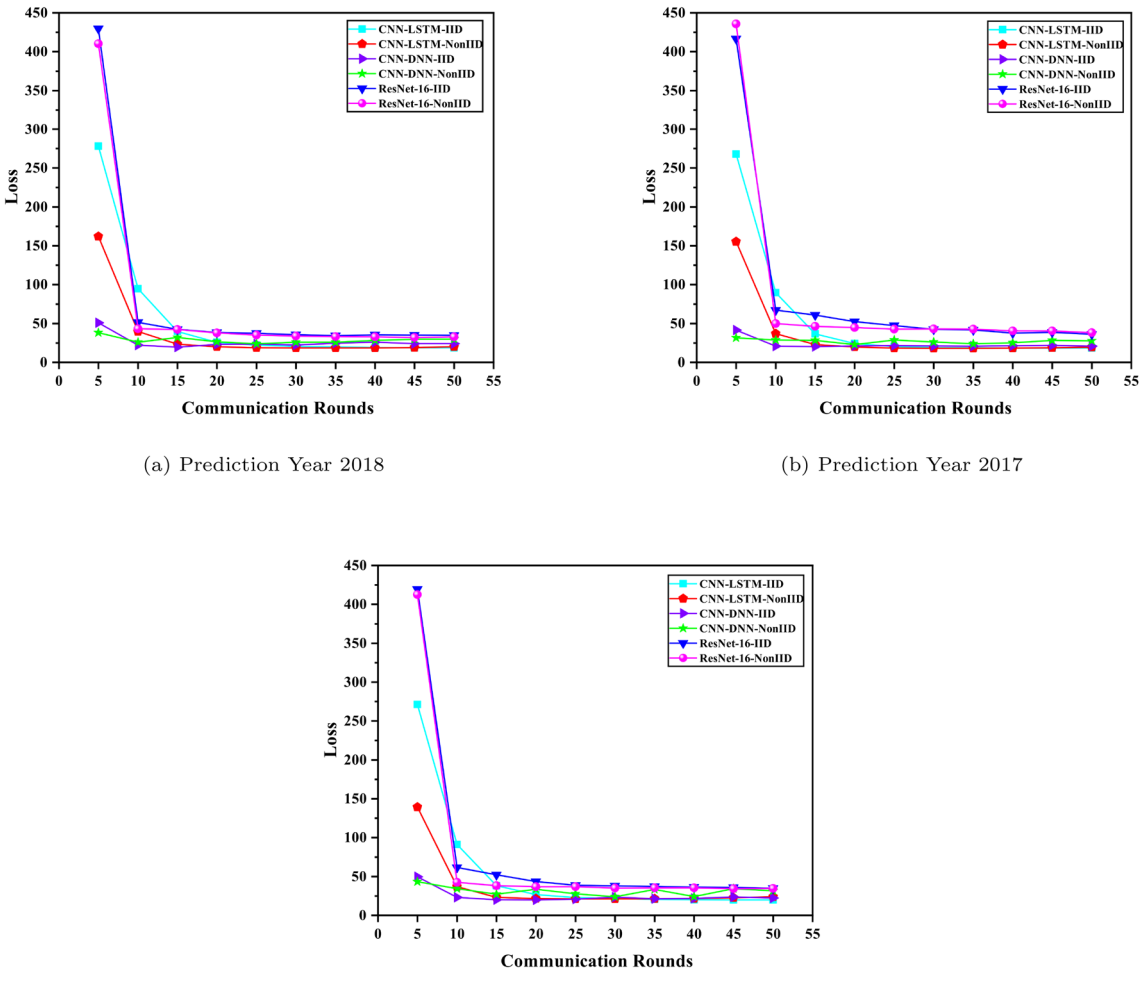
Table 4 Performance evaluation for soybean yield prediction (bushels/acre) using centralized deep learning models

Learning Method	Learning model	Prediction year	Test set evaluation metrics		
			MSE	RMSE	R^2
Centralized Learning	ResNet-16	2018	27.3259	5.2274	0.77
		2017	22.4849	4.7418	0.75
		2016	26.6382	5.1612	0.56
	ResNet-28	2018	34.5532	5.8782	0.71
		2017	28.3706	5.3264	0.73
		2016	30.1603	5.4918	0.63
	CNN-DNN	2018	22.7753	4.7723	0.80
		2017	20.8540	4.5666	0.75
		2016	19.6388	4.4315	0.75
	CNN-LSTM	2018	22.0666	4.6975	0.84
		2017	19.4803	4.4136	0.84
		2016	19.2216	4.3842	0.81

CNN-LSTM- are considered for further experimentation in the federated setting.

Vanilla Federated Learning with IID and Non-IID Data for Yield Prediction

The federated training experiments for soybean yield prediction are conducted by horizontal partitioning of data in IID and Non-IID fashion. We have considered three and five client organizations in the federated experimental setup. In the context of IID data distribution, data samples obtained from nine states of the US and their corresponding counties are equally shuffled and redistributed among three and five clients. While considering Non-IID data distribution, data samples are distributed based on the geographical locations of their respective states. In the case of non-IID data, data samples from the states <Illinois, Indiana, Missouri>, <Iowa, Minnesota, North Dakota>, and <Kansas, Nebraska, South Dakota> are grouped and assigned to three clients. Similarly, for the scenario with five clients, the data is redistributed by combining states such as <Illinois, Indiana>, <Iowa, Missouri>, <Minnesota, North Dakota>, <Nebraska, South Dakota>, and <Kansas> thereby creating non-IID data subsets for each client. The federated training is carried out using the previously discussed deep learning models like ResNet-16, CNN-DNN, and CNN-LSTM. At the client level, the key hyperparameters such as batch size (B), local epochs (E), and communication rounds (N) are tuned for various combinations of number of clients (K) and data distributions (IID/Non-IID). However, for the three models, the B and E differ depending on the data distribution. With IID data and three clients, the observed values are: ResNet-16 (B: 64, E: 10), CNN-DNN (B: 128, E: 10), and CNN-LSTM (B: 256, E: 20). In the case of five clients with IID data these values remain same. For Non-IID data and three

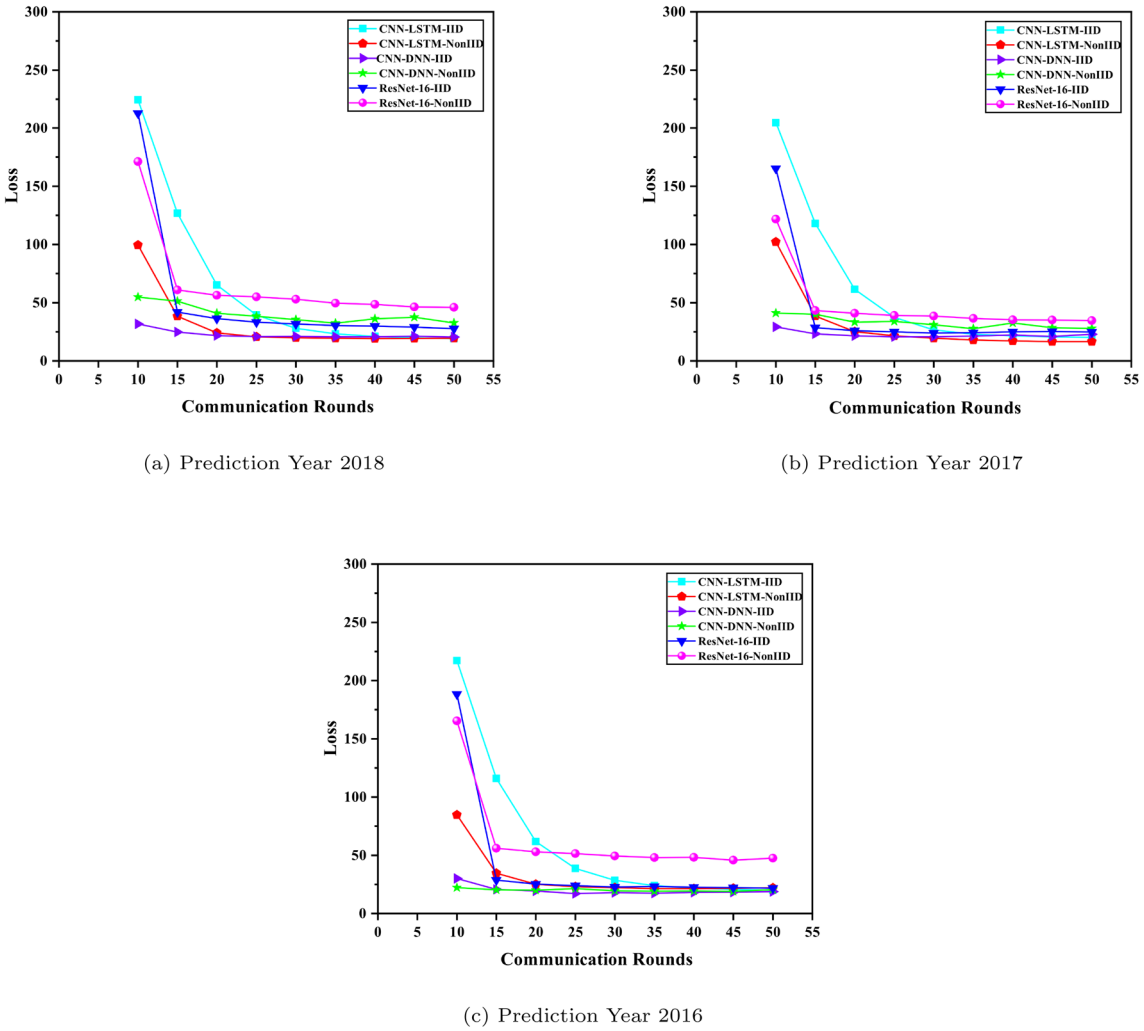


(a) Prediction Year 2018

(b) Prediction Year 2017

(c) Prediction Year 2016

Fig. 8 Global loss versus rounds for deep learning models with three clients using vanilla federated learning



(a) Prediction Year 2018

(b) Prediction Year 2017

(c) Prediction Year 2016

Fig. 9 Global loss versus rounds for deep learning models with five clients using vanilla federated learning

clients, suggested values are ResNet-16 (B: 128, E: 20), CNN-DNN (B: 128, E: 20), and CNN-LSTM (B: 256, E: 25). Specifically, for five clients with Non-IID data, the values of B and E remain same. Figure 8a–c illustrates the convergence process for all three models with three clients (IID/Non-IID) and test data from different prediction years. In addition, Fig. 9a, b details the convergence process for multiple models with five clients (IID/Non-IID), employing test data from various prediction years. Based on experimental observations, all three models (ResNet-16, CNN-DNN, CNN-LSTM) trained on three and five clients with both IID and Non-IID data converge approximately after 50 communication rounds. Tables 5 and 7 present the centralized and federated evaluation metrics for three learning models considering various parameters such as the number of clients, different data distributions, and prediction years. Based on analysis of experimental results from Tables 5 and 7, it can be inferred that in the case of three and five clients with IID and Non-IID data, the CNN-LSTM model consistently outperforms ResNet-16 and CNN-DNN models for prediction years 2018, 2017, and 2016 in vanilla federated learning. The percentage change in RMSE values is calculated for the CNN-LSTM model between centralized and vanilla federated setting values. This evaluation aims to assess its suitability for federated learning. The comparison of the CNN-LSTM model tested using vanilla federated learning (centralized evaluation) with its centralized counterpart shows a marginal variation of −8 percent to +12 percent in RMSE value, where a negative percentage indicates a decrease in RMSE value in the federated setting compared to the centralized setting, and positive percentage indicate an increase. Similarly, comparing the CNN-LSTM model tested using vanilla federated learning (federated

Table 5 Performance evaluation for soybean yield prediction (bushels/acre) using vanilla federated learning with three clients

Method	Model	Data distribution	Prediction year	Centralized evaluation metrics			Federated evaluation metrics		
				MSE	RMSE	R^2	MSE	RMSE	R^2
Vanilla FL	ResNet-16	IID	2018	34.8330	5.9019	0.70	44.5935	6.6767	0.60
			2017	36.1490	6.0124	0.60	43.6148	6.6033	0.49
			2016	34.8756	5.9055	0.62	44.8935	6.7002	0.61
		Non-IID	2018	32.7951	5.7267	0.72	52.7575	7.2633	0.52
			2017	38.2817	6.1872	0.55	51.2103	7.1491	0.39
			2016	34.5764	5.8802	0.45	43.3338	6.7540	0.31
	CNN-DNN	IID	2018	24.1129	4.9105	0.79	26.7496	5.1708	0.75
			2017	20.7649	4.5569	0.77	21.2668	4.6112	0.76
			2016	22.4563	4.7388	0.65	22.8957	4.7843	0.62
		Non-IID	2018	29.8550	5.4640	0.75	45.0015	6.6757	0.59
			2017	27.7618	5.2689	0.68	31.8250	5.6040	0.62
			2016	31.6951	5.6298	0.49	38.3297	6.1875	0.35
	CNN-LSTM	IID	2018	18.9095	4.3485	0.85	20.9910	4.5808	0.82
			2017	18.5655	4.3088	0.85	19.2252	4.3794	0.83
			2016	19.9222	4.4634	0.83	20.9784	4.5800	0.81
		Non-IID	2018	20.0456	4.4772	0.84	20.5739	4.5343	0.83
			2017	19.6099	4.4283	0.83	21.4132	4.6218	0.82
			2016	23.9318	4.8920	0.78	25.1418	5.0095	0.78

evaluation) with its centralized counterpart shows a marginal variation of −6 percent to +14 percent in RMSE value, following the same negative and positive percentages interpretation.

DP-Federated Learning with IID and Non-IID for Yield Prediction

The privacy-preserved federated training experiments for soybean yield prediction with IID and Non-IID data are performed using DP-federated learning methods outlined in Algorithms 2 and 3. For the best performing CNN-LSTM model, DP-federated learning is applied and evaluated for respective prediction years. At the client level, DP is employed by taking suitable values of clipping factor(C) and noise level (σ) to avoid any vulnerabilities of privacy leakage while sharing model updates. Generally, based on the privacy budget (ϵ) expected in a particular domain, the practitioner defines a privacy-utility trade-off by adjusting the values of C and σ. Specifically, in our training, the values

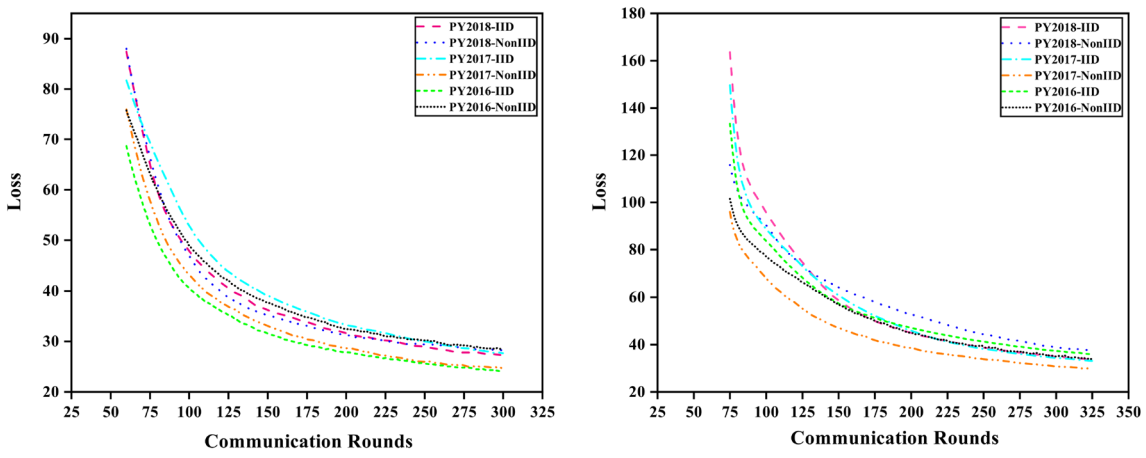


Fig. 10 Global loss versus rounds for CNN-LSTM with DP-federated learning

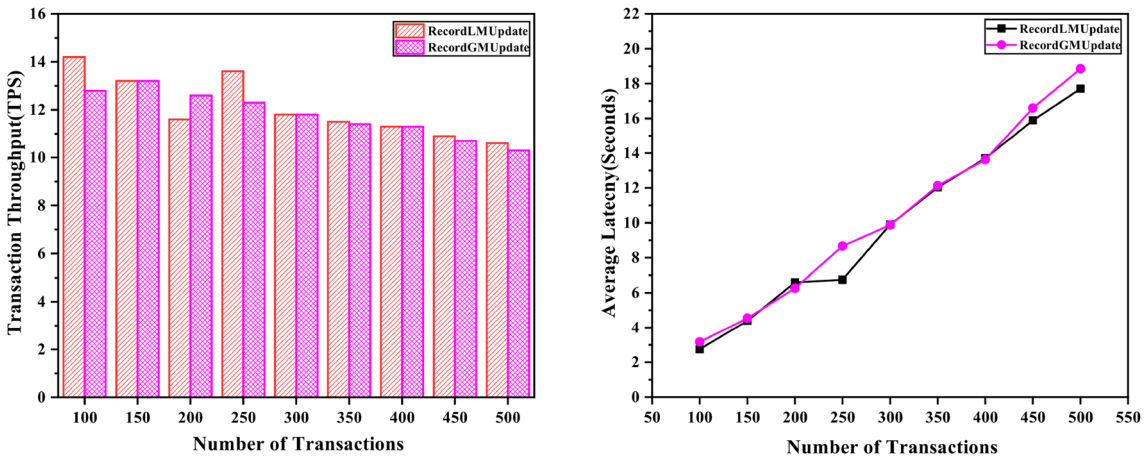
Table 6 Performance evaluation for soybean yield prediction (bushels/acre) using dp-federated learning with three clients

Method	Model	Data Distribution	Prediction Year	Centralized Evaluation Metrics			Federated Evaluation Metrics		
				MSE	RMSE	R^2	MSE	RMSE	R^2
DP-FL	CNN-LSTM	IID	2018	27.2545	5.2206	0.78	26.4957	5.1469	0.77
			2017	27.6655	5.2598	0.76	26.1798	5.1155	0.78
			2016	24.1048	4.9097	0.78	24.7176	4.9709	0.78
		Non-IID	2018	28.2238	5.3126	0.79	26.7842	5.1752	0.76
			2017	24.6852	4.9684	0.79	26.8742	5.1838	0.77
			2016	28.5599	5.3441	0.74	28.7135	5.3577	0.75

of C and σ are varied to obtain ϵ in the range of 10 to 14 for both IID and Non-IID data. For training with three clients, C=3, σ=1 is taken, while for five clients, C=1.5, σ=1.1 is considered. By considering the CNN-LSTM model, hyperparameter tuning is done for batch size (B), local epochs (E), and communication rounds (N) for varying combinations of the number of clients (K) and data distributions (IID/Non-IID). The results of hyperparameter tuning for three clients with IID and Non-IID is B:256, E:60, N:300 whereas, for five clients with IID and Non-IID, it is B:256, E:60, N:325. Figure 10a, b shows the convergence process of DP-federated learning for the CNN-LSTM model with three clients and five clients employing IID and Non-IID from various prediction years. The performance metrics for the CNN-LSTM model with DP-federated learning are detailed in Table 6 and Table 8, considering the different number of clients, data distributions, and prediction years. Figure 12a–d shows the plots for actual versus predicted yield for the prediction year 2018 with different values of clients and data distributions. For the CNN-LSTM model with DP-federated learning setting, there is approximately a 10 percent to 30 percent increase in RMSE value when tested with three and five clients, different prediction years, and data distributions, and compared with its centralized learning model. The modest decline in the CNN-LSTM model's performance is observed when applying DP can be attributed to the fact that privacy-preserving mechanisms inherently entail a trade-off between protecting privacy and optimal utility.

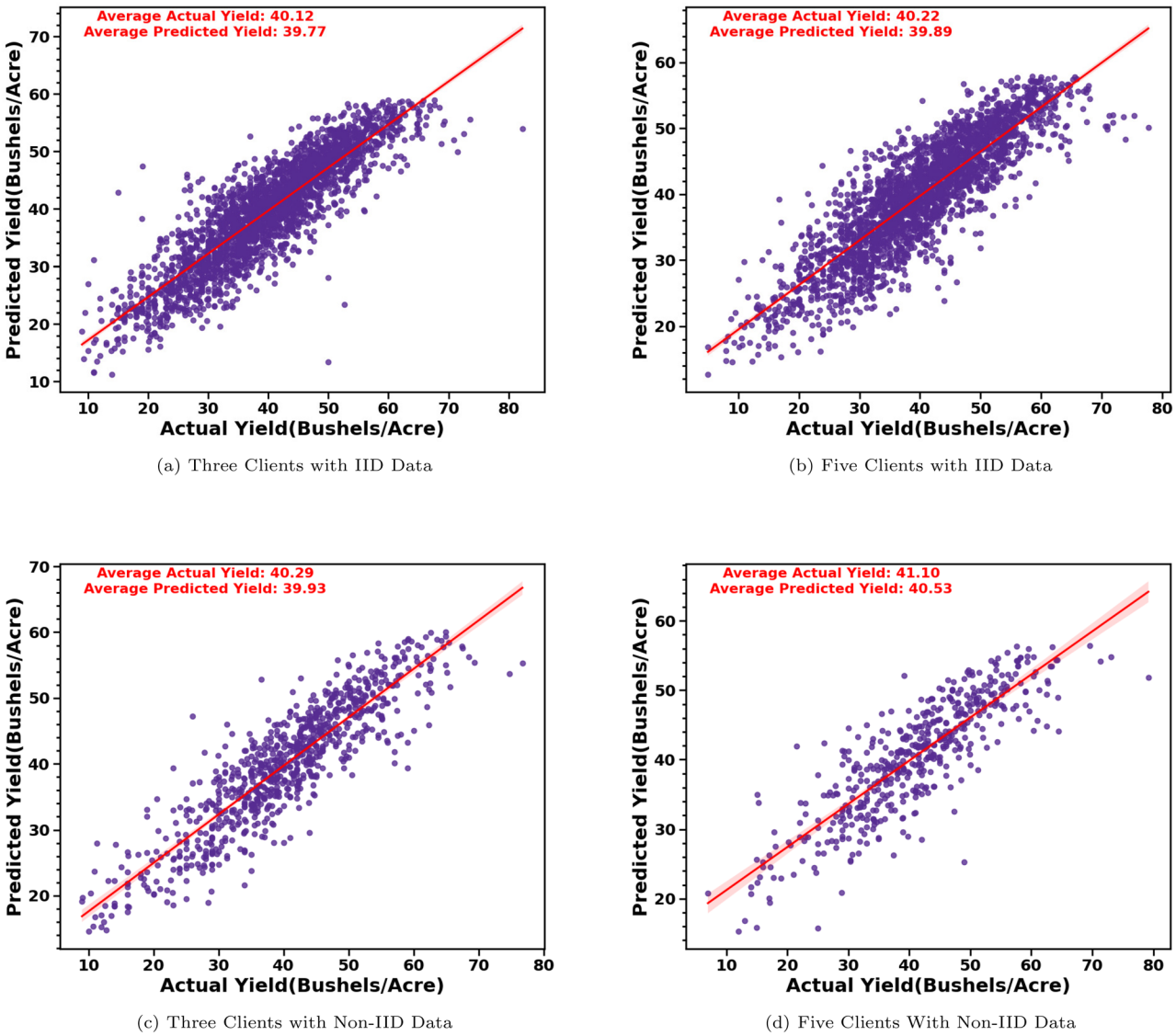
Smart Contracts for Model Updates Exchange in Federated Learning

A smart contract named **"ModelUpdate"** is designed with **recordLMUpdate()** and **recordGMUpdate()** functions to record the metadata associated with local and global model updates exchanged between clients and coordinator during federated learning into the blockchain. The **recordLMUpdate()** function invoked by clients in a federated round record **<ClientID,Timestamp,$LM_{updatesize}$,Hashvalue, Roundnumber>** as a transaction while **recordGMUpdate()** function invoked by coordinator in federated environment adds **<CoordinatorID, Timestamp,$GM_{updatesize}$, Hashvalue, Roundnumber>** transaction into the blockchain. The



(a) Throughput (b) Average Transaction Latency

Fig. 11 Throughput and average latency for "ModelUpdate" smart contract functions



(a) Three Clients with IID Data

(b) Five Clients with IID Data

(c) Three Clients with Non-IID Data

(d) Five Clients With Non-IID Data

Fig. 12 Actual vs predicted soybean yield using CNN-LSTM model with DP-federated learning for PY-2018

successful execution of **recordLMUpdate()** and **recordGMUpdate()** functions triggers an acknowledgement event to clients and coordinator in a particular communication round. The **"ModelUpdate"** smart contract is coded in Solidity and deployed on the Ethereum blockchain environment using Ganache. The performance of smart contracts functions is measured using metrics such as Transaction Throughput ($T_{Throughput}$) and Average Transaction Latency ($T_{AvgLatency}$) to determine its scalability in federated learning. $T_{Throughput}$ is the rate at which valid transactions are committed into the blockchain in a given time period and measured in terms of Transactions Per Second (TPS). $T_{AvgLatency}$ is a time period between the transaction being submitted, and confirmation received for the same. $T_{Throughput}$ and $T_{AvgLatency}$ for the **"ModelUpdate"** smart contract transactions are measured using a Hyperledger Caliper tool. Initially, smart functions **recordLMUpdate()** and **recordGMUpdate()** that append transactions to the blockchain are arranged in the form of the number of rounds for measuring the metrics. By increasing the number of transactions from 100 to 500 at a fixed rate and with a transaction send rate of 50 to the Ganache blockchain, evaluation is done for throughput and latency. Figure 11a and b illustrate $T_{Throughput}$ and $T_{AvgLatency}$ for **recordLMUpdate()** and **recordGMUpdate()** transactions. Based on the analysis of plots, we infer that both local and global model update functions in smart contract exhibit an optimal level of throughput and average latency, both of which fall well within the acceptable limits suitable for

Table 7 Performance evaluation for soybean yield prediction (bushels/acre) using vanilla federated learning with five clients

Method	Model	Data distribution	Prediction Year	Centralized evaluation metrics			Federated evaluation metrics		
				MSE	RMSE	R^2	MSE	RMSE	R^2
Vanilla FL	ResNet-16	IID	2018	27.6058	5.2541	0.75	29.3595	5.4109	0.73
			2017	24.7434	4.9743	0.74	23.8681	4.8785	0.74
			2016	21.7849	4.6674	0.64	21.4934	4.6214	0.64
		Non-IID	2018	45.9736	6.7804	0.58	65.4265	8.0729	0.41
			2017	34.1688	5.8838	0.63	45.5578	6.7032	0.41
			2016	47.4614	6.8892	0.54	56.5692	7.5056	0.45
	CNN-DNN	IID	2018	20.6247	4.5414	0.8	22.0811	4.6968	0.80
			2017	22.9773	4.7935	0.73	24.057	4.9022	0.72
			2016	18.7804	4.3336	0.69	18.7736	4.3322	0.68
		Non-IID	2018	32.5571	5.7059	0.69	51.0578	7.0881	0.55
			2017	27.8761	5.2798	0.67	34.8948	5.8913	0.56
			2016	20.4653	4.5239	0.67	28.4307	5.3258	0.54
	CNN-LSTM	IID	2018	19.1176	4.3724	0.83	19.4328	4.4052	0.83
			2017	20.1389	4.4876	0.83	21.9268	4.6806	0.81
			2016	19.9576	4.4674	0.83	20.0072	4.4690	0.82
		Non-IID	2018	19.5387	4.4203	0.83	20.4432	4.5197	0.83
			2017	16.6057	4.0750	0.85	19.8384	4.4505	0.83
			2016	22.1097	4.7021	0.79	23.0201	4.7918	0.79

Table 8 Performance evaluation for soybean yield prediction(bushels/acre) using DP-federated learning with five clients

Method	Model	Data distribution	Prediction year	Centralized evaluation metrics			Federated evaluation metrics		
				MSE	RMSE	R^2	MSE	RMSE	R^2
DP-FL	CNN-LSTM	IID	2018	33.5456	5.7919	0.73	31.2055	5.5850	0.74
			2017	33.1047	5.7537	0.71	34.3837	5.8624	0.71
			2016	35.9308	5.9942	0.69	35.6098	5.9652	0.71
		Non-IID	2018	37.5365	6.1267	0.70	34.7898	5.8972	0.71
			2017	29.8363	5.4623	0.71	35.8675	5.9844	0.69
			2016	33.9497	5.8266	0.71	33.7462	5.8084	0.71

federated learning in the smart agricultural domain. In addition, it also demonstrates that transactions can be flexibly scaled to higher volumes (Fig. 12).

Discussion

This study has presented the AgriFLChain framework, which enables trusted federated learning in smart agriculture using blockchain. It has demonstrated the framework's application by implementing a soybean yield prediction task. The primary motivation for this study is to resolve the challenges encountered when harnessing agricultural data for machine learning tasks through the implementation of federated learning. At first, SSI-based DIDs and VCs ascertain the authentication and data provenance of all client organizations participating in a federated ecosystem. Implementing DID and VC-based techniques using blockchain authorizes and establishes trusted connections among federated learning participants. This technique offers several benefits over centralized authentication schemes in preventing attacks. Using VCs for data provenance prevents correlation attacks with blinded signatures in the credentials used for verification. Anchoring the link secret to the credential by the credential holder thwarts forgery attempts by malicious actors. Adopting DIDs for client organizations prevents malicious actors from creating fake identities and carrying out impersonation attacks.

Table 9 Comparison of RMSE (bushels/acre) values obtained for prediction year 2018 under different vanilla federated learning and DP-federated learning settings with centralized learning

Model	Learning Mode	RMSE
Khaki et al.,2020 [45] - CNN-RNN	Centralized Learning(Baseline)	4.9100
AgriFLChain(Our work) - CNN-LSTM	Centralized Learning	4.6975
	Vanilla Federated Learning	4.4046
	DP-Federated Learning	5.6129

The DID and VC-based authentication is immune to man-in-the-middle attacks and substitution attacks as communicating parties within the federated ecosystem are mutually verified using their digital signatures. It prevents model poisoning attacks by ensuring that only authenticated and verified client organizations participate in the federated training process. Further, during data provenance, the selective disclosure of VC attributes to verifiers minimizes the amount of personal data collected in the context of authentication.

Analysis of deep learning models based on RMSE values in centralized and vanilla federated learning reaffirms the fact that CNN-LSTM is suitable for modeling soybean yield data considered in the study. Notably, the best performing CNN-LSTM model in vanilla federated learning outperforms the centralized baseline, as detailed in Table 9. The evaluation of DP-federated learning demonstrates that the CNN-LSTM model performs on par with centralized models. However, performance is slightly decreased due to balancing privacy protection with model utility. The type of data distribution does not considerably impact the CNN-LSTM model's performance as it can be observed that RMSE value slightly increases with Non-IID data experimented with five clients. Adopting blockchain in federated learning brings traceability and transparency to the model updates exchanged among participants. The throughput and average latency measurements of smart contract functions elicit the confidence that blockchain transactions are scalable during federated learning. Therefore, we conclude that privacy-preserved federated learning schemes deployed on the blockchain platform perform at a comparable level to centralized learning models.

Conclusion

Trusted agricultural data is needed to leverage the quality of prediction tasks in smart agriculture used for decision-making. In this study, we introduced the AgriFLChain framework, which relies on blockchain to enable trusted learning in smart agriculture. The framework utilizes federated learning, which is exhibited by the crop yield prediction task. We described and implemented the detailed architecture of four deep learning models that include ResNet-16, ResNet-28, CNN-DNN and CNN-LSTM for yield prediction. Initially, these models are trained in a conventional way in a centralized fashion. Subsequently, the method to achieve authentication and data provenance for client organizations within a federated environment is described with SSI-based DIDs and VCs. Then, vanilla and DP-federated learning is used to train deep learning models using different combinations of client organizations and data distributions for multiple prediction years. Detailed algorithms are described to perform a DP-federated learning at the client side and coordinator end, respectively. Smart contract based functions are designed and implemented to work in synergy with federated learning to create an auditable record of metadata associated with model updates. A detailed performance evaluation of centralized and federated learning models conducted in varied setups indicates the effectiveness of the CNN-LSTM model for yield prediction. Furthermore, a smart contract's throughput and latency evaluation exhibits its suitability for federated learning. Thus, the framework implemented in this study attempts to demonstrate the suitability of blockchain-based federated learning for smart agriculture tasks like yield prediction. This study has considered only the horizontal data partitioning with few client organizations. As a future work, we want to scale this study with more client organizations for experimentation with vertical data partitioning scheme. The study has utilized federated averaging for model update aggregation with differential privacy. However, this can be extended to more efficient federated aggregation techniques and other privacy-preserving mechanisms to overcome the privacy-utility trade-off. This framework can be enhanced to address additional security and privacy concerns using threat modeling.

Author Contributions All authors contributed to the study conception, investigation, validation, and writing-review editing. All authors have read and approved the final manuscript.

Funding Open access funding provided by Manipal Academy of Higher Education, Manipal.

Data availability The dataset used in this study is publicly available at [63], [64] and [65]. Researchers interested in accessing the dataset can obtain it from the provided source.

Declarations

Conflict of interest The authors declare no competing interests.

Ethical approval Not applicable.

References

1. United Nations Organization.: UN Sustainable Development Goals. (accessed:05.03.2023). Available from: https://www.un.org/sustainabledevelopment/hunger/.
2. World Bank.: Food System Jobs. (accessed:05.03.2023). Available from: https://www.worldbank.org/en/topic/food-system-jobs.
3. Fukase E, Martin W. Economic growth, convergence, and world food demand and supply. World Dev. 2020;132: 104954. https://doi.org/10.1016/j.worlddev.2020.104954.
4. Lyubchich V, Newlands NK, Ghahari A, Mahdi T, Gel YR. Insurance risk assessment in the face of climate change: Integrating data science and statistics. Wiley Interdisciplinary Reviews: Computational Statistics. 2019 apr;11(4):e1462https://doi.org/10.1002/wics.1462.
5. ICRISAT.: ICRISAT Uses Satellite Data To Assess Crop Cutting Experiments For Crop Yield Estimations. (accessed:05.03.2023). Available from: https://www.icrisat.org/icrisat-uses-satellite-data-to-assess-crop-cutting-experiments-for-crop-yield-estimations/.
6. Kosmowski F, Chamberlin J, Ayalew H, Sida T, Abay K, Craufurd P. How accurate are yield estimates from crop cuts? Evidence from smallholder maize farms in Ethiopia. Food Policy. 2021;102:102122. https://doi.org/10.1016/j.foodpol.2021.102122.
7. Benos L, Tagarakis AC, Dolias G, Berruto R, Kateris D, Bochtis D. Machine Learning in Agriculture: A Comprehensive Updated Review. Sensors. 2021;21(11). https://doi.org/10.3390/s21113758.
8. Linsner S, Kuntke F, Steinbrink E, Franken J, Reuter C. The Role of Privacy in Digitalization - Analyzing Perspectives of German Farmers. Proceedings on Privacy Enhancing Technologies. 2021;2021:334–50.
9. Wiseman L, Sanderson J, Zhang A, Jakku E. Farmers and their data: An examination of farmers' reluctance to share their data through the lens of the laws impacting smart farming. NJAS - Wageningen Journal of Life Sciences. 2019;90-91:100301. https://doi.org/10.1016/j.njas.2019.04.007.
10. Qu Y, Uddin MP, Gan C, Xiang Y, Gao L, Yearwood J. Blockchain-Enabled Federated Learning: A Survey. ACM Comput Surv. 2022;55(4). https://doi.org/10.1145/3524104.
11. Ali M, Karimipour H, Tariq M. Integration of blockchain and federated learning for Internet of Things: Recent advances and future challenges. Computers & Security. 2021;108:102355. https://doi.org/10.1016/j.cose.2021.102355.
12. Brendan McMahan H, Moore E, Ramage D, Hampson S. Agüera y Arcas B. PMLR: Communication-efficient learning of deep networks from decentralized data; 2017.
13. Yang Q, Liu Y, Chen T, Tong Y. Federated Machine Learning: Concept and Applications. arXiv: Artificial Intelligence. 2019;.
14. Nakamoto S.: Bitcoin: A peer-to-peer electronic cash system. (accessed:06.03.2023). Available from: http://bitcoin.org/bitcoin.pdf.
15. Bodkhe U, Tanwar S, Parekh K, Khanpara P, Tyagi S, Kumar N, et al. Blockchain for Industry 4.0: A Comprehensive Review. IEEE Access. 2020;8:79764–79800. https://doi.org/10.1109/ACCESS.2020.2988579.
16. Javed AR, Hassan MA, Shahzad F, Ahmed W, Singh S, Baker T, et al. Integration of Blockchain Technology and Federated Learning in Vehicular (IoT) Networks: A Comprehensive Survey. Sensors. 2022;22(12). https://doi.org/10.3390/s22124394.
17. Mühle A, Grüner A, Gayvoronskaya T, Meinel C. A survey on essential components of a self-sovereign identity. Computer Science Review. 2018;30:80–6. https://doi.org/10.1016/j.cosrev.2018.10.002.
18. Allen C.: The Path to Self-Sovereign Identity. (accessed:06.03.2023). Available from: http://www.lifewithalacrity.com/previous/.
19. Cucko S, Kersic V, Turkanović M. Towards a Catalogue of Self-Sovereign Identity Design Patterns. Applied Sciences. 2023;13(9). https://doi.org/10.3390/app13095395.
20. Hughes A, Sporny M, Reed D.: A Primer for Decentralized Identifiers-An introduction to self-administered identifiers for curious people.(W3C Credentials Community Group). (accessed:05.03.2023). Available from: https://w3c-ccg.github.io/did-primer/.
21. Sporny M, Longley D, Chadwick D.: Verifiable Credentials Data Model 1.0: Expressing Verifiable Information on the Web. Accessed:06.03.2023. Available from: https://www.w3.org/TR/vc-data-model/#what-is-a-verifiable-credential.
22. Behera MR, Upadhyay S, Shetty S. Federated Learning using Smart Contracts on Blockchains, based on Reward Driven Approach. ArXiv. 2021;abs/2107.10243.
23. Ma C, Li J, Shi L, Ding M, Wang T, Han Z, et al. When Federated Learning Meets Blockchain: A New Distributed Learning Paradigm. IEEE Comput Intell Mag. 2022;17(3):26–33. https://doi.org/10.1109/MCI.2022.3180932.
24. Zhang Q, Palacharla P, Sekiya M, Suga J, Katagiri T. Blockchain-based Secure Aggregation for Federated Learning with a Traffic Prediction Use Case. In: 2021 IEEE 7th International Conference on Network Softwarization (NetSoft); 2021. p. 372–374.
25. Dwork C, Kenthapadi K, McSherry F, Mironov I, Naor M. Our data, ourselves: Privacy via distributed noise generation. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2006;4004 LNCS:486–503. https://doi.org/10.1007/11761679_29.
26. Dwork C, Roth A. The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science. 2013;9(3–4):211–487. https://doi.org/10.1561/0400000042.
27. Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, et al. Deep Learning with Differential Privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. Association for Computing Machinery; 2016. p. 308-318.
28. McMahan HB, Ramage D, Talwar K, Zhang L. Learning differentially private recurrent language models. International Conference on Learning Representations, ICLR; 2018. .
29. Murthy CS, Poddar MK, Pandey V, Biswal A, Choudhary KK.: Replacing CCE-yield estimates with modeled-yield estimates for crop insurance. (accessed:05.03.2023). Available from: https://pmfby.gov.in/compendium/Technology/2021%20-%20Replacing%20CCE-yield%20with%20modelled%20yield-Technical%20Report.pdf.
30. Ahmed OAA, Serra T. Economic analysis of the introduction of agricultural revenue insurance contracts in Spain using statistical copulas. Agric Econ. 2015;46:69–79.
31. Goodwin BK, Hungerford A. Copula-based models of systemic risk in U.S. Agriculture: Implications for crop insurance and reinsurance contracts. American Journal of Agricultural Economics. 2014;97(3):879–896. https://doi.org/10.1093/ajae/aau086.
32. Bzdok D, Altman N, Krzywinski M. Points of Significance: Statistics versus machine learning. Nat Methods. 2018;15:233–4.

33. Ranjan AK, Parida BR. Paddy acreage mapping and yield prediction using sentinel-based optical and SAR data in Sahibganj district, Jharkhand (India). Spat Inf Res. 2019;27:399–410. https://doi.org/10.1007/s41324-019-00246-4.
34. Wei MCF, Molin JP. Soybean Yield Estimation and Its Components: A Linear Regression Approach. Agriculture. 2020;10(8). https://doi.org/10.3390/agriculture10080348.
35. Dalhaus T, Finger R. Can gridded precipitation data and phenological observations reduce basis risk of weather index-based insurance? Weather, climate and society. 2016;8(4):409–19. https://doi.org/10.1175/WCAS-D-16-0020.1.
36. Amaratunga V, Wickramasinghe L, Perera A, Jayasinghe J, Rathnayake U, Zhou JG. Artificial Neural Network to Estimate the Paddy Yield Prediction Using Climatic Data. Mathematical Problems in Engineering. 2020;2020. https://doi.org/10.1155/2020/8627824.
37. Mwaura JI, Kenduiywo BK. County level maize yield estimation using artificial neural network. Modeling Earth Systems and Environment. 2020;7:1417–24. https://doi.org/10.1007/s40808-020-00943-2.
38. Shahhosseini M, Hu G, Archontoulis SV. Forecasting Corn Yield With Machine Learning Ensembles. Frontiers in Plant Science. 2020;11. https://doi.org/10.3389/fpls.2020.01120.
39. Filippi P, Jones E, Wimalathunge NS, Somarathna PDSN, Pozza LE, Ugbaje SU, et al. An approach to forecast grain crop yield using multi-layered, multi-farm data sets and machine learning. Precision Agriculture. 2019;p. 1–15. https://doi.org/10.1007/s11119-018-09628-4.
40. Son NT, Chen CF, Chen CR, Guo HY, Cheng YS, Chen SL, et al. Machine learning approaches for rice crop yield predictions using time-series satellite data in Taiwan. Int J Remote Sens. 2020;41(20):7868–88. https://doi.org/10.1080/01431161.2020.1766148.
41. Gandhi N, Armstrong LJ, Petkar O, Tripathy AK. Rice crop yield prediction in India using support vector machines. In: 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE); 2016. p. 1–5.
42. Khaki S, Wang L. Crop Yield Prediction Using Deep Neural Networks. Frontiers in Plant Science. 2019;10. https://doi.org/10.3389/fpls.2019.00621.
43. Kim N, Na SI, Park CW, Huh M, Oh J, Ha KJ, et al. An Artificial Intelligence Approach to Prediction of Corn Yields under Extreme Weather Conditions Using Satellite and Meteorological Data. Applied Sciences. 2020;10(11). https://doi.org/10.3390/app10113785.
44. Srivastava AK, Safaei N, Khaki S, Lopez G, Zeng W, Ewert F, et al. Winter wheat yield prediction using convolutional neural networks from environmental and phenological data. Scientific Reports. 2022;12(1). https://doi.org/10.1038/s41598-022-06249-w.
45. Khaki S, Wang L, Archontoulis SV. A CNN-RNN Framework for Crop Yield Prediction. Frontiers in Plant Science. 2020;10. https://doi.org/10.3389/fpls.2019.01750.
46. Jiang H, Hu H, Zhong R, Xu J, Xu J, Huang J, et al. A deep learning approach to conflating heterogeneous geospatial data for corn yield estimation: A case study of the US Corn Belt at the county level. Glob Change Biol. 2020;26(3):1754–66. https://doi.org/10.1111/gcb.14885.
47. Elavarasan D, Vincent PMDR. A reinforced random forest model for enhanced crop yield prediction by integrating agrarian parameters. J Ambient Intell Humaniz Comput. 2021;12(11):10009–22. https://doi.org/10.1007/s12652-020-02752-y.
48. Milham MP, Craddock RC, Son JJ, Fleischmann M, Clucas J, Xu H, et al. Assessment of the impact of shared brain imaging data on the scientific literature. Nature Communications. 2018;9(1). https://doi.org/10.1038/s41467-018-04976-1.
49. Durrant A, Markovic M, Matthews D, May D, Enright J, Leontidis G. The role of cross-silo federated learning in facilitating data sharing in the agri-food sector. Computers and Electronics in Agriculture. 2022;193. https://doi.org/10.1016/j.compag.2021.106648.
50. Vimalajeewa D, Kulatunga C, Berry D, Balasubramaniam S. A Service-based Joint Model Used for Distributed Learning: Application for Smart Agriculture. IEEE Trans Emerg Top Comput. 2021. https://doi.org/10.1109/TETC.2020.3048671.
51. Kumar P, Gupta GP, Tripathi R. PEFL: Deep Privacy-Encoding based Federated Learning Framework for Smart Agriculture. IEEE Micro. 2021. https://doi.org/10.1109/MM.2021.3112476.
52. Friha O, Ferrag MA, Shu L, Maglaras L, Choo KKR, Nafaa M. FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things. Journal of Parallel and Distributed Computing. 2022;165:17–31. https://doi.org/10.1016/j.jpdc.2022.03.003.
53. Patros P, Ooi M, Huang V, Mayo M, Anderson C, Burroughs S, et al. Rural AI: Serverless-Powered Federated Learning for Remote Applications. IEEE Internet Computing. 2022;p. 1–5. https://doi.org/10.1109/MIC.2022.3202764.
54. Zhang Q, Zhao X, Han Y, Yang F, Pan S, Liu Z, et al. Maize yield prediction using federated random forest. Computers and Electronics in Agriculture. 2023;210. https://doi.org/10.1016/j.compag.2023.107930.
55. Li A, Markovic M, Edwards P, Leontidis G. Model pruning enables localized and efficient federated learning for yield forecasting and data sharing. Expert Syst Appl. 2024;242: 122847. https://doi.org/10.1016/j.eswa.2023.122847.
56. Idoje G, Dagiuklas T, Iqbal M. Federated Learning: Crop classification in a smart farm decentralised network. Smart Agricultural Technology. 2023;5. https://doi.org/10.1016/j.atech.2023.100277.
57. Aggarwal M, Khullar V, Goyal N, Prola TA. Resource-efficient federated learning over IoAT for rice leaf disease classification. Computers and Electronics in Agriculture. 2024;221:109001. https://doi.org/10.1016/j.compag.2024.109001.
58. Devaraj H, Sohail S, Ooi M, Li B, Hudson N, Baughman M, et al. RuralAI in Tomato Farming: Integrated Sensor System, Distributed Computing, and Hierarchical Federated Learning for Crop Health Monitoring. IEEE Sensors Letters. 2024;8(5):1–4. https://doi.org/10.1109/LSENS.2024.3384935.
59. Qammar A, Karim A, Ning H, Ding J. Securing federated learning with blockchain: a systematic literature review. Artif Intell Rev. 2022. https://doi.org/10.1007/s10462-022-10271-9.
60. Arachchige PCM, Bertok P, Khalil I, Liu D, Camtepe S, Atiquzzaman M. A Trustworthy Privacy Preserving Framework for Machine Learning in Industrial IoT Systems. IEEE Trans Industr Inf. 2020;16(9):6092–102. https://doi.org/10.1109/TII.2020.2974555.
61. Shen R, Zhang H, Chai B, Wang W, Wang G, Yan B, et al. BAFL-SVM: A blockchain-assisted federated learning-driven SVM framework for smart agriculture. High-Confidence Computing. 2024;p. 100243. https://doi.org/10.1016/j.hcc.2024.100243.
62. Nie J, Jiang J, Li Y, Li J, Qiao Y, Ercisli S. UAVEC-FLchain: Distributed multi-regional jujube orchard joint yield estimation for secure agricultural-IoT applications. Internet of Things. 2024;25:101143. https://doi.org/10.1016/j.iot.2024.101143.
63. Thornton PE, Thornton M, Mayer B, Wei Y, Devarakonda R, Vose RS, et al. Daymet: Daily Surface Weather Data on a 1-km Grid for North America, Version 3; 2016. .
64. Hengl T, Mendes de Jesus J, Heuvelink GBM, Ruiperez Gonzalez M, Kilibarda M, Blagotić A, et al. SoilGrids250m: Global gridded soil information based on machine learning. PLOS ONE. 2017 02;12:1–40. https://doi.org/10.1371/journal.pone.0169748.
65. USDA-NASS.: USDA - national agricultural statistics service. (accessed:19.03.2023). Available from: https://www.nass.usda.gov/.

66. LeCun Y, Bengio Y, Hinton G. Deep learning nature. 2015;521(7553):436–44.
67. Goodfellow I, Bengio Y, Courville A. Deep learning. MIT press; 2016.
68. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80.
69. He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016:770–778.
70. Chen D, Hu F, Nian G, Yang T. Deep Residual Learning for Nonlinear Regression. Entropy. 2020;22(2). https://doi.org/10.3390/e22020193.
71. Manoj T, Makkithaya K, NV G. A trusted IoT data sharing and secure oracle based access for agricultural production risk management. Comput Electron Agric. 2023;204:107544. https://doi.org/10.1016/j.compag.2022.107544.