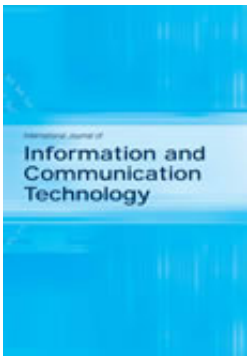
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Information and Communication Technology

ISSN online: 1741-8070 - ISSN print: 1466-6642
https://www.inderscience.com/ijict

Federated learning-enabled personalised delivery and student privacy protection in universities

Yunxian Li

DOI: 10.1504/IJICT.2026.10076059

Article History:

Received:	27 September 2025
Last revised:	19 October 2025
Accepted:	25 October 2025
Published online:	13 February 2026

Federated learning-enabled personalised delivery and student privacy protection in universities

Yunxian Li

School of Economics and Management,
Hebei North University,
Zhangjiakou, Hebei, 075000, China
Email: liyunyun5@163.com

Abstract: This study presents a federated learning framework enhanced with entropy-adaptive differential privacy, blockchain consensus, and knowledge distillation to safeguard student data while improving personalised education. Traditional federated learning preserves privacy by training collaboratively without sharing raw data, yet faces challenges of heterogeneity, efficiency, and resilience against malicious clients. Existing solutions like homomorphic encryption and secure multiparty computation often incur high computational costs and limited adaptability. To address these limitations, the proposed framework employs blockchain-based role incentives to ensure fairness and verifiability, while entropy-adaptive differential privacy dynamically balances privacy and utility. Knowledge distillation further improves robustness and mitigates non-IID data distribution issues. Experiments on a Python programming course dataset with 2,452 students demonstrate superior accuracy, fairness, and resilience compared to conventional FedAvg. The method achieves up to 97% prediction accuracy with enhanced stability under adversarial conditions, offering a scalable and secure solution for personalised, privacy-preserving education.

Keywords: federated learning; FL; personalised federated learning; PFL; student privacy protection; differential privacy; DP; homomorphic encryption; HE; blockchain-based federated learning; entropy-adaptive differential privacy; EADP; secure multi-party computation; SMPC.

Reference to this paper should be made as follows: Li, Y. (2026) 'Federated learning-enabled personalised delivery and student privacy protection in universities', *Int. J. Information and Communication Technology*, Vol. 27, No. 10, pp.42–62.

Biographical notes: Yunxian Li teaches at Hebei North University's School of Economics and Management in Zhangjiakou, Hebei, China. With a concentration on applied economic analysis, management science, and regional development studies, her academic work mostly focuses on economics and management-related fields. Her work focuses on applying empirical analysis and quantitative methodologies to real-world management and economic issues, supporting evidence-based policy assessment and decision-making. She has actively engaged in academic service, teaching, and research. She has also taken part in scholarly endeavours that promote interdisciplinary cooperation and the development of higher education.

1 Introduction

The utilisation of deep learning and the internet of things (IoT) has unquestionably contributed to the advancement of the smart industry. With deep learning's strong ability to classify and identify massive amounts of data, data-driven issues that crop up during the installation of Iota can be more easily addressed. The powerful capability, however, necessitates an enormous amount of data collection and sharing across clients and organisations. Data privacy is an issue due to the prevalence of shared data and the lack of control over its intended use, both of which could include sensitive information. A distributed deep learning model known as federated learning (FL) has arisen in response to data privacy legislation like the General Data Protection Regulation (GDPR) and the Personal Information Protection Law (PIPL). The vast possibilities of FL in fields like autonomous driving, smart health, and the IoT have piqued the imagination of many since Google's 2016 proposal (Fu and Zhang, 2022). The dispersed nature of FL, with its emphasis on communicating only gradients rather than directly sharing data, may provide some protection for client-sensitive information. Nevertheless, FL continues to face the issues listed below, according to recent studies.

One major issue that has come out in recent studies is the fact that privacy leaking from gradients is still a problem in FL. Even though consumers' data is stored locally, research has demonstrated that critical information may still be deduced from input gradients. There are now three main approaches to the privacy leaking problem that have been addressed in the literature. To begin, homomorphic encryption (HE) permits the transmission and processing of ciphertext, which contains sensitive information. This quality motivates the development of several HIM-based methods (Wang, 2020). Take PFMLP (Fang et al., 2023), a secure FL protocol that uses partial HE as an example.

We developed a secure protocol for vertical federated learning (FL) that leverages homomorphic encryption (HE) and randomisation techniques to protect data privacy during collaborative model training. Second, multiparty computing is an inevitable aspect of any SMC in this context refers to the well-known cryptographic paradigm of secure multi-party computation, also known as secure multiparty computing. Multiple participants can jointly calculate a function over their private inputs with SMC, which guarantees that no party learns more about another's personal information than can be deduced from the outcome. Because the training process naturally involves numerous clients and at least one coordinating server, SMC is a crucial part of federated learning (FL) settings. SMC guarantees the confidentiality of sensitive data, like gradients or local model changes, throughout transmission and aggregation by utilising strategies like secret sharing and safe aggregation. As a result, the paragraph's usage of the term 'secure multiparty computing' (SMC) is both technically and conceptually correct.

By employing double masking, Verify Net protects the local gradient's privacy while computing the aggregated gradient, guaranteeing that the gradient stays private. Differential privacy (DP) is an alternative to HIM and SMC that safeguards sensitive parameters by means of noise introduction. Differential privacy (DP) is an alternative to HIME and SMC that protects sensitive parameters by introducing random noise during gradient training, which may inevitably lead to a degradation in model performance. Although these methods have achieved privacy preservation to a certain degree, they do not consider heterogeneity. Furthermore, if the aggregation server becomes rogue, the majority of these protocols will not be able to identify or stop the harmful actions.

Personalised federated learning (PFL) (PPVP), which is both secure and easy to verify, is the solution this research suggests for the problems we have already identified. For the purpose of protecting both the global and private local gradients, HE is employed. To confirm the overall findings, Lagrange interpolation and commitment are employed. These safeguards have the added benefit of detecting and preventing server-client collusion attacks that attempt to bypass verification. Due to developments in AI, VR and the IoT, the consumer electronics (CE) industry has undergone a dramatic transformation.

Among the many predictions made by the 'Statista Research Department' for the years 2023–2028 is a rise of $125.5 billion in worldwide CE market revenue. Based on these facts, it is anticipated that a 'data lake' will be generated due to the CE devices' massive and steep expansion (Javed, 2021). The vast majority of CE now has internet connectivity, allowing it to provide users with an abundance of services. The problem of information overload, which hinders timely access to online resources of interest, is a result of the fact that the amount of information on the internet has far surpassed consumer requirements. This has led to an explosion in the need for recommender systems. Recommender systems filter through enormous amounts of dynamically generated content, find pertinent parts based on user interests, preferences, or observed behaviour, and then provide them to the user in an effort to reduce information overload. In addition, personalised recommendation systems (PRS), which use personalisation for product suggestions, are the subject of much research. But most PRS do their data processing and storage on centralised servers. In particular, the large amounts of processing power available on cloud servers allow for the analysis, visualisation and extraction of relevant data (Wang et al., 2021).

However, there are security and privacy concerns with transferring customer data to the cloud, since an attacker or a bad cloud service provider might potentially compromise or steal PRS data, leading to data breaches and identity theft. Another aspect that could compromise privacy is the possibility of selling the data to third-party organisations that intend to use it for product suggestion purposes. One of the new technological developments that allows machine-learning models to be executed in a distributed manner is FL. FL-based PRS addresses consumers' security and privacy concerns. However, there are still certain issues with FL-based PRSs' explainability, computation and communication costs, and their functionality in 5G and future networks. With the advent of digital tools in engineering education comes a deluge of student data, which presents great possibilities for improving teaching methods and paving the way for individualised education (Javeed et al., 2023). The data used to analyse student outcomes and enhance teaching quality comes from these databases, which include information on learning habits, academic performance, and interaction records. Python programming is an essential course for many undergraduate engineering programmes.

It teaches students to think computationally and code. The data-driven nature of educational research is greatly enhanced by the large-scale, structured datasets generated by its online distribution. However, with data protection rules around the world becoming stricter and educational ethics standards constantly changing, there are serious privacy concerns with students' personal information (Afrose, 2021). Our solution to this problem is EADP-Feedbag, which stands for entropy-adaptive differential privacy federated averaging and is specifically tailored to secure student privacy in Python programming courses while also improving prediction accuracy. An excellent option for educational applications that prioritise privacy is FL, since it allows several clients to train a model simultaneously without disclosing any raw data to the service.

Nevertheless, when traditional FL is applied alongside fixed-noise DP, performance is frequently severely hindered by noise overload. By altering the noise strength dynamically based on the model's average output entropy, EADP-Feedbag achieves a superior accuracy-privacy balance even when privacy is severely limited. Researchers at Baoji University of Arts and Sciences analysed test data from 2,452 of 493 electronic engineering majors (Chen and Qi, 2025). There are four levels of performance (fail, passed, good and excellent) and seventeen features (such as the number of clicks on courses, assignment scores, and study time) in the dataset. The trials are carried out in a controlled setting with a multilayer perceptron (MLP) model and ten federated clients.

The structure of this paper is organised as follows: Section 2 presents the related work on the FL-enabled personalised delivery. Section 3 outlines the methodology based on protection in universities. Section 4 discusses the results related to financial inclusion. Finally, Section 5 provides the conclusions.

1.1 Contribution of this study

A FL architecture that improves personalised learning results while protecting sensitive student data is introduced in this paper, which contributes to the field of privacy-preserving education technology. The suggested method assures security, equity, and transparency in collaborative model training by combining FL with entropy-adaptive differential privacy (EADP), blockchain mechanisms, and distillation defences, as opposed to traditional centralised models that expose raw student information to possible misuse. Overcoming the limits of fixed-noise techniques, which generally decrease model performance, the framework achieves a compromise between prediction accuracy and privacy protection by dynamically modifying privacy settings based on model entropy. Furthermore, this research enriches the educational data-mining ecosystem by providing an effective solution for analysing large-scale online learning data without compromising student confidentiality. Experimental results on Python programming course datasets demonstrate the method's superior convergence, robustness against malicious attacks, and resilience in heterogeneous environments. Beyond improving accuracy and safeguarding privacy, the integration of blockchain-based consensus and incentive mechanisms ensures verifiability, fairness, and sustained participation of distributed nodes. Thus, this work advances the practical adoption of FL in universities, setting the foundation for secure, trustworthy, and scalable AI-driven personalised education systems.

2 Related works

2.1 Foundations of FL

FL preserves the privacy and security of decentralised data while enabling collaborative training of different AI models using locally stored data. It is a revolutionary, decentralised machine-learning platform. The primary goal of this methodology is to protect the confidentiality of important data. Edge devices in a traditional FL architecture pool their model parameters using methods like standard gradient descent (SGD) and federated averaging (FedAvg) to train a global model. Clients can gain useful insights into overall model update patterns via communication-mitigated federated learning

(CMFL), which minimises duplicate data uploads. This method expedites the convergence of learning and enhances the efficiency of communication (Wen, 2022). An improved version of the classic dropout method for trimming models is the federated dropout (Fed Drop) strategy. It drastically cuts down on communication overhead by giving clients credit for contributions based on how well their local models perform. Problems with statistical heterogeneity and personalisation arise when using the same local model to train all customers in standard FL frameworks. These problems were addressed by the introduction of PFL, which generates unique models for individual or group customers.

In addition, each group of clients has a globally tailored model created for them using clustered federated learning (CFL) methods that group clients with comparable data distribution. By emphasising aggregated parameters and local adaptability, the Fed Per and Fiefdoms approaches seek to overcome the shortcomings of regional models. In most cases, all clients in traditional FL frameworks share the same global. PFL approaches reduce this disparity by tailoring models to meet the unique needs of clients or groups. A crucial step in this context is knowledge distillation, which allows smaller data models to benefit from bigger, more complicated learning models. To improve personalised learning while decreasing communication and computing overheads, approaches like federated model distillation (FEDD) have been created (Kaushal, 2025). A coordinating server oversees the process as edge devices build and improve personalised models by sharing model gradients, using transfer learning techniques from federated knowledge distillation (Feck). Concerns about privacy and security may arise, though, because this method could reveal confidential information.

2.2 AI-assisted learning, bias, and privacy challenges in education

Even malevolent users can recover datasets and high-resolution photos from shared gradients. Unfortunately, many of the current countermeasures for these kinds of assaults have limitations, such as inefficient operation and large computing costs (Shawkat et al., 2025). Tools for education have relied heavily on AI-assisted learning, which has translated into numerous AI/ML models of students' learning trajectories. One example is the ability of open learner models to provide accurate representations of students' progress, enhance metacognitive tasks, and encourage self-regulation of learning through monitoring one's own performance (Salim et al., 2021). Moreover, studies have shown that AI-assisted learning can personalise its approaches to training based on the specific requirements of each learner. This could significantly improve students' formal and informal academic performance. Despite AI and ML's potential for education, the conventional, centralised ML training methods encounter three major roadblocks when trying to improve educational systems with AI and ML models. Furthermore, even with huge student datasets, the issue of unavailable data remains, which raises the possibility of majority-group bias (Sengupta, 2024).

This has the ability to diminish the efficacy of AI/ML models, particularly when it comes to helping minority students, like women, who are underrepresented in STEM professions. Finally, there is the issue of data privacy, which becomes more pressing when many sources' data is combined to train ML models, potentially exposing sensitive student information. The use of FL and other distributed ML methods in the educational ecosystem has been the subject of some recent investigations into potential solutions to these problems. However, despite these growing contributions, FL is still relatively in its

early stages when it comes to education. Notably lacking is a future-focused study that synthesises FL's potential benefits for AI-assisted education, its potential drawbacks, and its implementation across all tiers of the education ecosystem (Hridi et al., 2024). This study aims to fill that informational need and suggests avenues for further research that might lead to FL's widespread adoption in the classroom. To address the issue of empirical risk minimisation, we can use the generalised FedAvg algorithm. Each round begins with the server relaying the global model's current state to the participating clients, who then perform various local optimisation processes and relay either the updated model or a differential update back to the server.

2.3 Personalisation and DP in FL

When using local datasets that are samples of non-congruent district buttons, this strategy fails to minimise both the regional and global objectives simultaneously, leading to underperformance. Therefore, other methods for addressing this problem surfaced, including the necessity of PFL. Authors provide three clustering models and data interpolation-based personalisation strategies in Andrew (2021). The study of hypothesis-based clustering provides further convergence guarantees of the population loss function. The approach also involves clustering the clients who are participating to create a personalised model. It then introduces a meta-algorithm to check if the clients are from non-congruent distributions, a method to cluster based on the cosine similarity of the updates, and a way to determine if the federated optimisation has reached minima of the clients' and servers' objectives. The works above assert privacy protection by ensuring that the clients' local raw data remains undisclosed during server-client connection cycles (Balle, 2019). To address this problem, several studies have concentrated on privatising the (federated) optimisation algorithm within the DP framework.

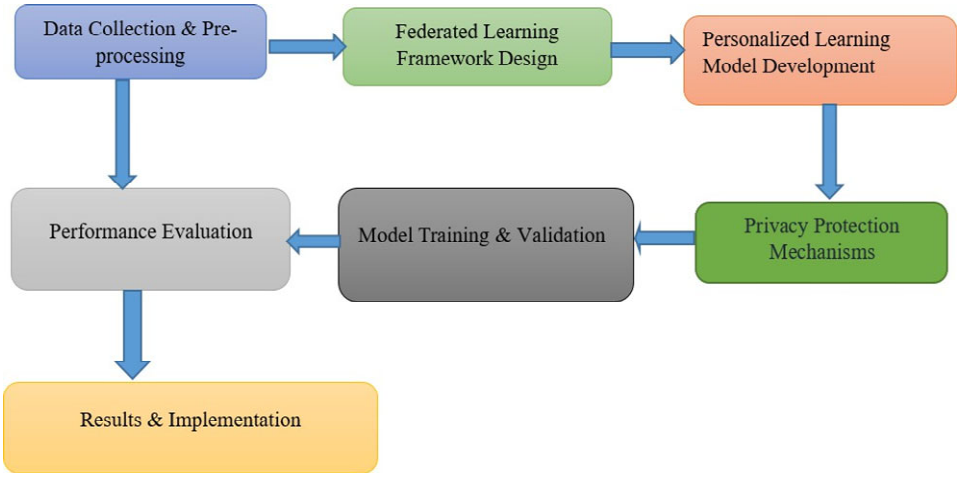
This guarantees formally that the learnt model will not rely too heavily on whether a specific user's record is present or not in the dataset used for the federated optimisation. This reduces the attacker model to that of a trustworthy, inquisitive foe whose access is limited to the trained model. Unfortunately, there is no safeguard in place to prevent the server or any other third party from intercepting client updates under this configuration. To increase the guarantee of the t-statistic in relation to central DP, the authors utilise shuffling, subsampling, and other approaches within the context of local DP. Importantly, a reliable aggregator is still necessary for these methods to work. The work delves into quantisation methods that boost communication efficiency and provide DP guarantees at the local level, protecting users from an unreliable or careless aggregator. Client updates belonging to the bounded domain of the diameter two game should be distinguishable up to a small multiplicative factor (Galli et al., 2022); thus, we point out how using local DP with non-trivial guarantees would be problematic with personalisation in relation to the works above. For the elementary scenario of inputs with convex, 1-Lipschitz cost functions, the writers tackle the issue of locally differentially private FL.

Notably, many statistical modelling techniques, including neural networks, do not rely on this assumption, and neither do most machine-learning models. On the other hand, we do not make these assumptions.

3 Methodology

Figure 1 illustrates the systematic workflow of a FL framework for personalised model development. After deciding what needs fixing, the next steps are to gather data and prepare it for analysis. A FL framework is then designed to support decentralised learning, while privacy protection mechanisms are integrated to ensure secure data handling. The personalised learning model is developed and undergoes model training and validation. Performance evaluation is conducted to assess effectiveness, and the results are implemented for practical application.

Figure 1 Workflow of FL framework for personalised model development (see online version for colours)



3.1 Data pre-processing

This study used four pre-processing approaches to clean, extract features from, normalise, and partition the 2,452 valid records. We updated the Python programming online test dataset to make it more compatible with the EADP-Feedbag MLP model and to ensure the results were good. These procedures dealt with irregularities in feature scale, outliers and missing values.

3.1.1 Data cleaning

There were 13 invalid entries eliminated from the original 2,465 raw records. Duplicate entries, records with unusual values, records without programming scores, and submissions with response times less than 5 minutes were all part of this category (Chen, 2025). To keep the data consistent, we also excluded error logs that had nothing to do with Python syntax. Following the cleaning process, 2,452 records were left to be analysed.

3.1.2 Feature extraction

After the dataset was cleaned, 17 features were extracted. Two demographic elements, a count of submissions and reaction times, and five scoring features – programming, multiple choice, fill-in-the-blank and true/false – made up the survey. Every demographic detail was considered. The MLP model requires tabular input; thus, we built each feature vector for each test attempt separately, without aggregating them over time.

3.1.3 Data normalisation

The data was normalised using z-scores because various attributes have different scales:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

The mean, μ, and standard deviation, σ, are included with the initial value, x, in equation (1). To make the model training process more stable, standardised attributes are changed to a mean of 0 and a standard deviation of 1. This stage did not include data on gender or categorical labels because they did not need to be normalised.

3.1.4 Data splitting

After student IDs and test identifiers were de-identified, 2,452 records, including demographic information, scores, and behaviours, made it into the final dataset. 1,961 training samples were obtained by dividing the dataset into an 8:2 training set and a 491-test set, following widely known approaches in educational data mining. Ten clients were given the training data, with around 196 records going to each client. Utilising independent and identically distributed (IID) sampling helped to streamline the training process. We saved the test set for our worldwide analysis.

3.2 FL framework design

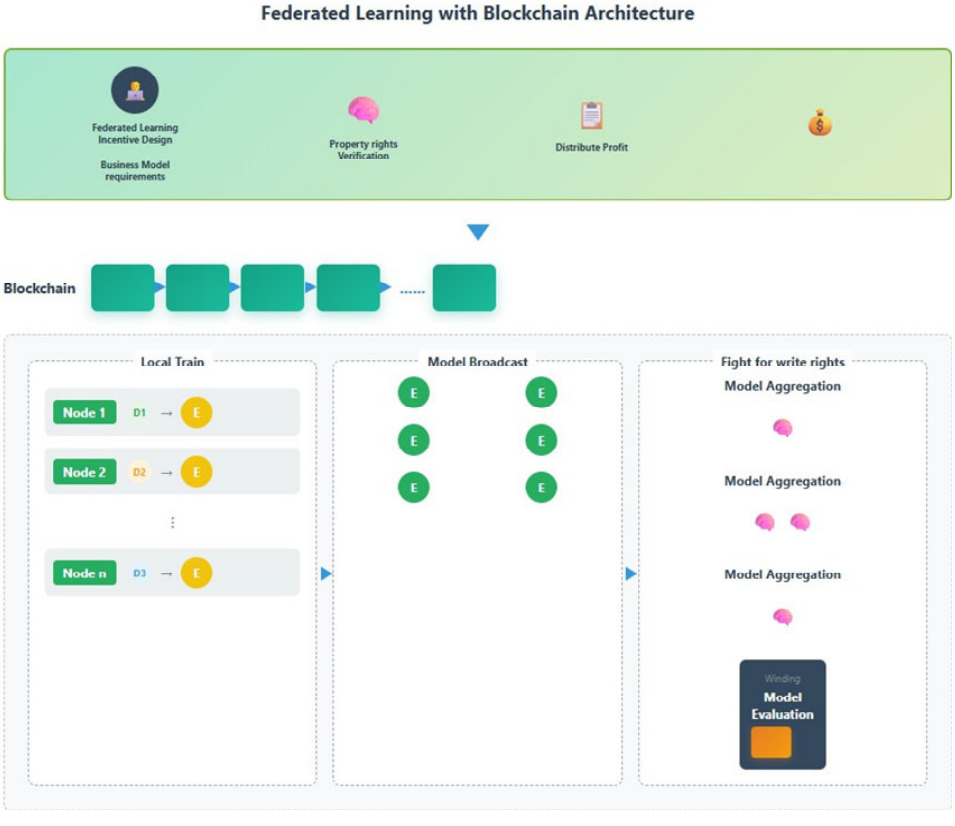
By combining FL with blockchain technology, our FedCFB architecture guarantees the entire federated exercise is fair and honest while also addressing the three issues raised in the section. We devised a rotating centre architecture that deviates from the conventional centralised FL model by making the task issuer the central server. Client, blockchain, and federation layers make up the FedCFB's overall architecture, as shown in Figure 2 (Zhu, 2024). All federation tasks are scheduled and coordinated by the federation layer, the client layer mostly performs client actions, and the blockchain layer stores information about federation tasks, client models, incentives, etc.

This paper introduces the federated blockchain structure, depicted in Figure 2, to address the issues of FL security in a non-trust environment, significant resource wastage due to the consensus process, and declining node participation. Here, the most important data points recorded on the chain are the hash of the prior block, the aggregated model's parameters, the local gradient set that was used to build the aggregated model, property rewards according to evaluation criteria, and optimisation goals for the next training round. Nodes taking part in the protocol will train a gradient model with watermarks locally after acquiring the publicly published starting model and training target from the blockchain. Once it has enough gradient information, it will try to retrieve the aggregate

model using the aggregation technique and then disseminate the gradient model using the gossip protocol. Atlas, every node will be supplied with the combined model for evaluation. Simultaneously, the new block will be filled with the optimal model that was developed through voting and the optimisation targets for the next round of protocols. The distributed storage, hash chain, longest chain, and data tampering problems will all be considered during the blockchain's development to ensure its durability.

By keeping track of the block-level gradient model for each node as it is built, this method ensures that the aggregate model is accurate.

Figure 2 Blockchain system's structure (see online version for colours)



3.3 Privacy protection mechanisms

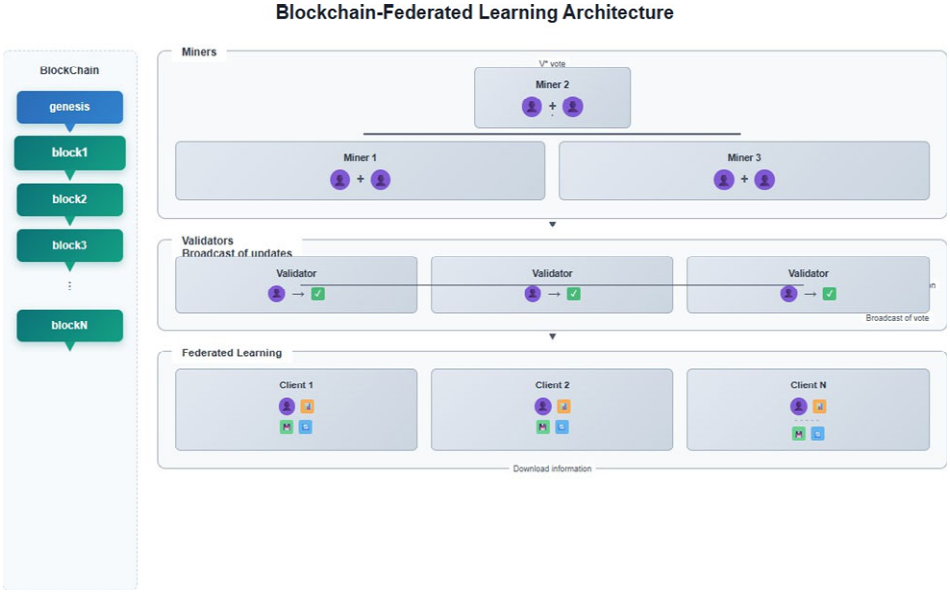
The system design of this research incorporates a blockchain that is collaboratively maintained by all clients, as seen in Figure 3. Everyone in the client pool takes on one of three responsibilities during a training round: miners, workers or validators. The operational customers employ distillation defence mechanisms and FL for data training to secure their local data. The system uses validating clients to assess the performance of active clients and make sure their models are safe and of good quality. Since the validators in this study do not possess direct access to the local data of other customers, they rely on proxy comparison methods to verify the data (Wan, 2024). Lastly, through a

few small responsibilities, the system logs the execution and result of every training round on the shared blockchain.

Role rotation and incentive mechanisms based on roles are used to make sure this study is fair and effective. The system's implemented approaches motivate each member to maintain system operations and submit data to the FL training. Furthermore, the role-based incentive mechanism decides which crucial part miners play inside the system. The mean of all the regional models *Mi* is the conventional FL model's name for the global model. The aggregation process can be illustrated in this way:

$$G = \sum_{t=1}^{n} w_t . M_t \tag{2}$$

Figure 3 Flowchart of a decentralised FL system design with distillation protection (see online version for colours)



The variables utilised in equation (2) are the local model's weight (*wi*) and the number of FL clients (*n*). The distillation local model was trained in this study using the prediction outputs of both the global and regional models. All of the regional models *Mi* have their loss functions provided by equation (3).

$$L_t = w_t . \left(L_{CE}\left(M_t, D_t\right) + \frac{\alpha}{T_t} . L_{KD}\left(M_t, G\right) \right) \tag{3}$$

The cross-entropy loss function, where *ai* is the loss function on the i^{th} device, is used to measure the performance of the local model on local data. How to account for information loss during distillation. One way to measure the disparity between the global model *G* and the regional model *Mi* is to use *Oi*, OKD. To compare the predictions of the international and regional models, a hyper-parameter $\alpha \in \{0, 1\}$ is utilised. A larger α signifies that the global model is more influential. The temperature parameter *Ti* from knowledge distillation and the local weight *wi* during aggregation are combined in this

study to control the amount of parameter softening. A smaller *Ti* places greater emphasis on the model's own predictions, whereas a larger one leads to smoother model forecasts. During the trial-and-error phase, local clients can tweak this temperature parameter to influence the degree of knowledge distillation. Every local model *Mi* keeps its distilled information throughout the distillation defence process. Equation (4) illustrates the procedure followed to incorporate this information into the study's global model *G*:

$$G = \sum_{t=1}^{n} w_t.\left(\beta.M_t + (1-\beta).G\right) \tag{4}$$

In this context, $\beta \in \{0, 1\}$ is a hyperparameter that regulates the impact of distilled knowledge on the weights of the global model. A smaller $\boldsymbol{\nu}$ value increases the likelihood that the model's own weights will be retained, but a larger $\boldsymbol{\nu}$ value amplifies the effect on the global model.

3.4 Performance evaluation

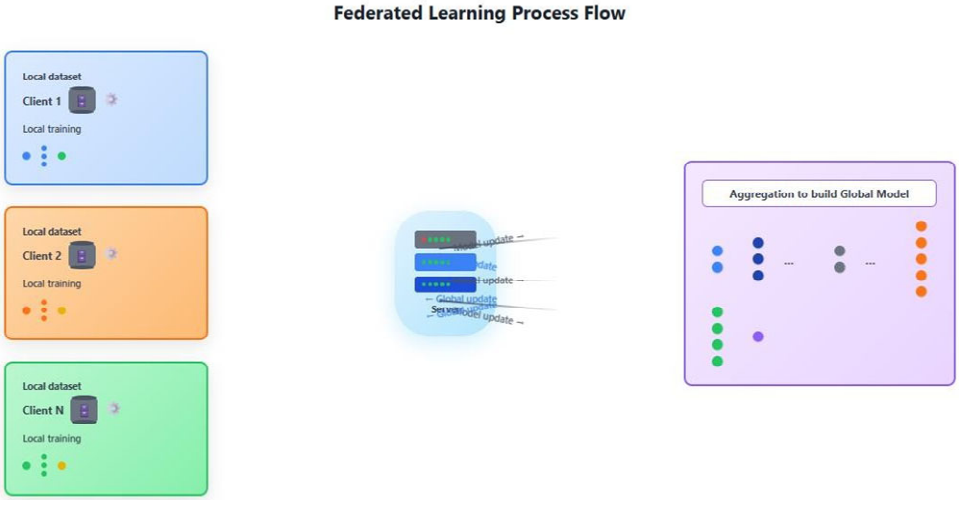
In our CFL-based system, the server sends the initial model parameters to each client, who uses their local datasets. This is followed by them sending the server an updated version of their models. After every client updates their model, the server node compiles the updated data and updates the global model. When dealing with aggregate, feedbag is the way to go. Every client who is a part of the programme receives the most recent update to the global model. Thus, by the process's conclusion, every customer has both a local model and a global model that are uniquely theirs. Algorithm 2 lays out the server-side algorithms, whereas Algorithm 1 lays out the client-side algorithms. In the algorithms, represents a client and represents the server. Algorithm 1 presents the steps of the client-side process. The clients are connected with the server and get model parameters. Each client splits its local dataset into batches and performs local model training (Mukherjee, 2025). The client updates the model parameters on the server after training is complete. The second algorithm specifies the steps to be taken by the server. Clients can access the server. After all clients have sent their model revisions to the server, it will save them.

The server gets model updates from all clients at the end of each round, applies an aggregation-using function, and then sends the updated model parameters to the clients. For as many rounds as the FL process takes into account, this procedure is repeated. In CFL, as seen in Figure 4, clients and the server collaborate during training to build the global model by combining the client-received model updates.

In CFL, the server acts as an aggregator and shares model updates with clients. Customers have access to both the global model update and their own customised models. Each of the clients performs data analysis locally through a collaborative training process without sharing the data. Hence, privacy is protected, and through collaborative training, prediction accuracy is enhanced. Model initialisation, local model training, update interchange, and aggregation all contribute to the total temporal complexity of the CFL process. The aggregation time complexity is expressed. Though there are several benefits, the CFL has some limitations. As the server performs as the aggregator, good network connectivity with the server is highly desirable. However, many applications do not have the provision of seamless connectivity with the server. Further, the overhead on the server is very high because the aggregation takes place inside the server. Further,

sharing model updates by all the clients with the server may raise a concern regarding security. To address these limitations, DFL has come.

Figure 4 The CFL process (see online version for colours)



Algorithm 1 Client-side algorithm

Input: Data, No Ne
Output: m
1: Function update (Data, N, N):
2: P Data pre-process(Data)
3: while connected with S, do
4: Train(m get model parameters())
5: end while
6: Save Parameters(m)
7: Function Train(m):
8: N Split(P Data, B) D Split data into N_1 batches
9: for e = 0 to N – 1 do
10: for b = 1 to N do
11: $m^2 + 1m^2$ nm^2 D Vm represents the gradient
12: end for
13: end for
N 14: mm
15: Send model update(m) D Send model update to S

Algorithm 2 Server-side algorithm

Input: Ne, fe, N
Output: mfinal
1: Function Collect(N. N,):
2: Connected Clients -

```
3: while (length(Connected Clients) ≠ N) do
4: listen()
5: acceptconnection()
6: end while
7: Fed Aug()
8: Release clients
9: Function Fed Avg():
10: m Init Model()                          initial model is generated
11: for 1 to N, do
12: M, Subset(max(f, N, 1),}}randome)
13: MU
14: force M, do
15: m getmodelupdate(c)                     Get model update from client cat round r
16: MU.append(m)                            Append model update of client c
17: end for
18: m² + 1 – M 5 M, c = 1
19: sendtoclients(m + 1)
20: end for
21: mfinalm N, +1
```

4 Results

To simulate an FL environment with numerous collaborating clients, experiments were carried out on a high-performance computing infrastructure. Thanks to a multi-core CPU and enough RAM for seamless parallel processing, the hardware design improved training efficiency and model convergence speed. The training process was made more stable and efficient with the help of the Pytorch framework, which made use of methods like gradient clipping, data loading acceleration, and the Adam optimiser.

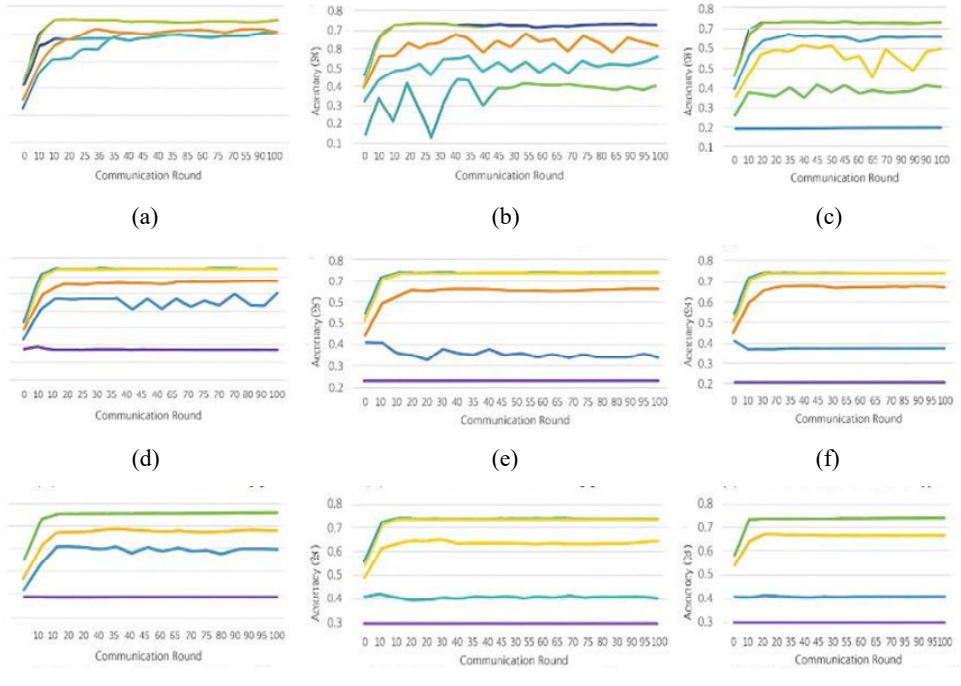
Table 1 The experimental setup

Category	*Details*
Hardware	
Processor (CPU)	Intel Core i7-9850H, up to 4.6 GHz
Memory (RAM)	64 GB
Graphics card (GPU)	NVIDIA GeForce RTX 2080
Software	
Operating system	Windows 10, 64-bit edition
Virtualisation platform	VMware Workstation version 16
Python release	Python 3.7
Machine learning library	PyTorch 1.9.1

This setup comes with 64 GB of RAM, a 6-core, 12-thread Intel Core i7-9850H CPU with a turbo speed of 4.6 GHz, and a graphics-processing unit from NVIDIA called an RTX 2080. This architecture allows for high-performance concurrent training of several

clients and accelerates model convergence. The software environment is configured to run on 64-bit Windows 10, and it uses Python 3.7 and PyTorch 1.9.1 to implement the EADP-FedAvg method and MLP model. To model distributed FL with isolated clients, VMware Workstation 16 is utilised. Popular Python libraries like SciPy, NumPy, and Scikit-learn are available for use in feature engineering, model evaluation and statistical analysis. With PyTorch's efficient tensor operations and automated differentiation, you can optimise and train MLP models quickly. Assessing the efficiency of different algorithms with federated tasks: we ran a battery of comparison experiments on the CIFAR-10 dataset to get a better feel for how each algorithm fared in FL scenarios. In Figure 5, you can observe the experimental outcomes.

Figure 5 Algorithm robustness against several forms of assault, (a) three malicious client of type 1 (b) five malicious client of type 1 (c) seven malicious client of type 1 (d) four malicious client of type 2 (e) five malicious client of type 2 (f) seven malicious client of type 2 (g) malicious client of type 2 (h) five malicious client of type 2 (i) seven malicious client of type 2 (see online version for colours)

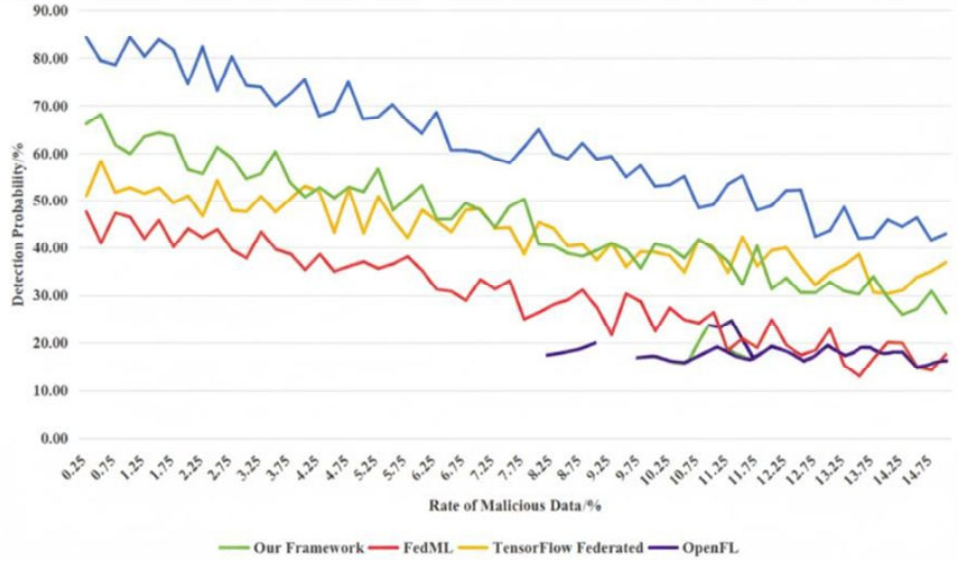


Notes: We examine the three top-performing algorithms to see how well they handle each communication round. At five malicious clients, the trimmed mean method becomes quite inaccurate in attack type 1, whereas the MutilKrum algorithm becomes extremely inaccurate at the same level of malicious clients (seven clients). The trimmed mean technique is rendered useless in assault type 2.

With a rise in the number of malevolent clients, the accuracy of all algorithms varies under attack type 1. In the presence of seven malicious nodes, these methods completely fall short when it comes to accomplishing global model convergence. The algorithms known as Krum, median, and trimmed-mean find type 2 assaults extremely difficult. On

the other hand, our suggested algorithms outperform multi-Krum in terms of stability and accuracy, even when subjected to hostile interference. By comparing our method to others on the Cifar-10 dataset, we can see that our algorithm is more stable and accurate, and that it is also more resilient and resistant when faced with hostile scenarios. Fatal defects, including huge variances, mistakes, and noise, are common in malevolent data. The data trainer could be misled if, even though the distribution of the data as a whole matches the sample, there is a large outlier in some data segments. Just as illustrated in Figure 6.

Figure 6 Consistent and detrimental clusters in the dataset sample (see online version for colours)



This study's technique, which is based on the distillation defence, was first compared to the standard FedAvg algorithm in an experiment that did not involve any dangerous devices. To make things easier to compare, let us say that no device introduced Gaussian noise during aggregation and that the FL process was unaffected by an external adversary using adversarial sample attacks. In this study, the term 'accuracy' refers to a model's performance measure in relation to a certain job, often the percentage of samples that are properly identified in a classification task. Figure 7 shows the experimental results, which demonstrate that the proposed model in the study had a slower convergence rate than the FedAvg algorithm after 30 rounds. The reason for this difference is that the model aggregation technique used in this work necessitates periodic distillation and information transmission. Because there is not enough variety in the training data, and client coverage is not complete before 30 rounds, the model performs poorly in its early stages. This value, however, is less than or equal to 0.05. The study's suggested model reaches a stable state and outperforms the classic FedAvg method as the number of training cycles increases. This proves that our technique is better at training models over the long-term and that the distillation defence we implemented while aggregating the models worked.

Figure 7 The proposed systems in this study were compared to the typical FedAvg algorithm in terms of accuracy, assuming no malicious device interaction (see online version for colours)

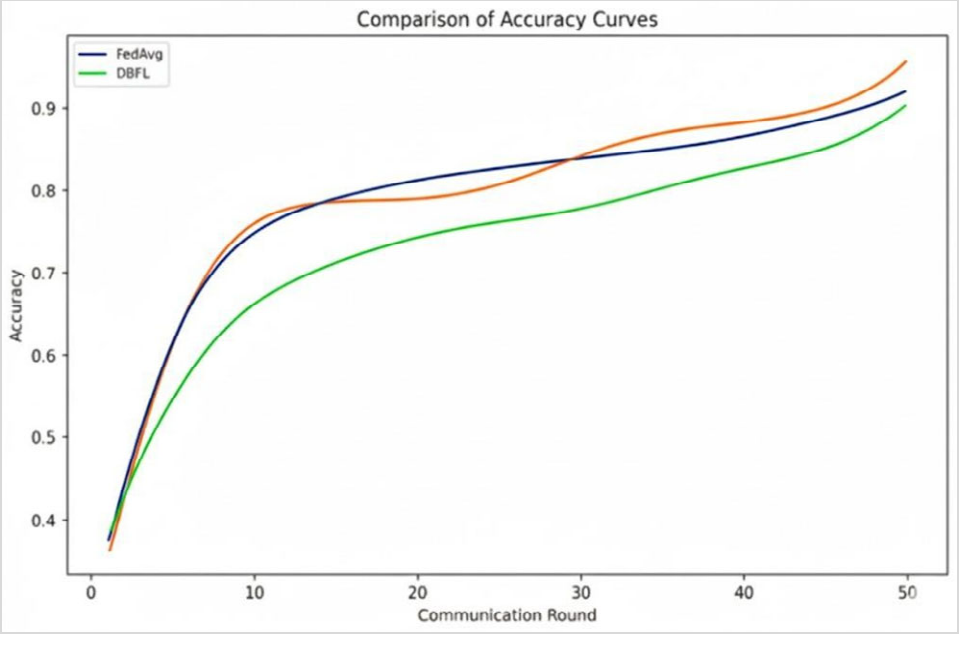


Figure 8 Variation of training and validation losses of client models across communication rounds (see online version for colours)

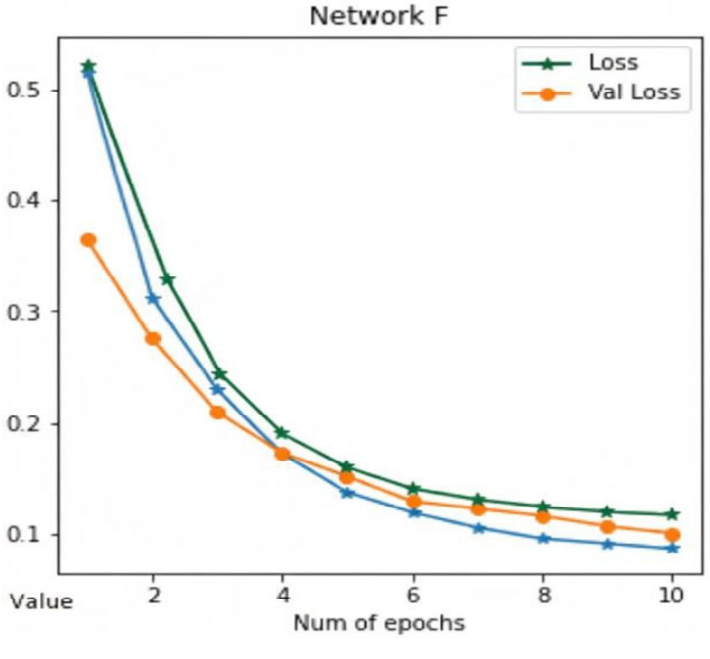
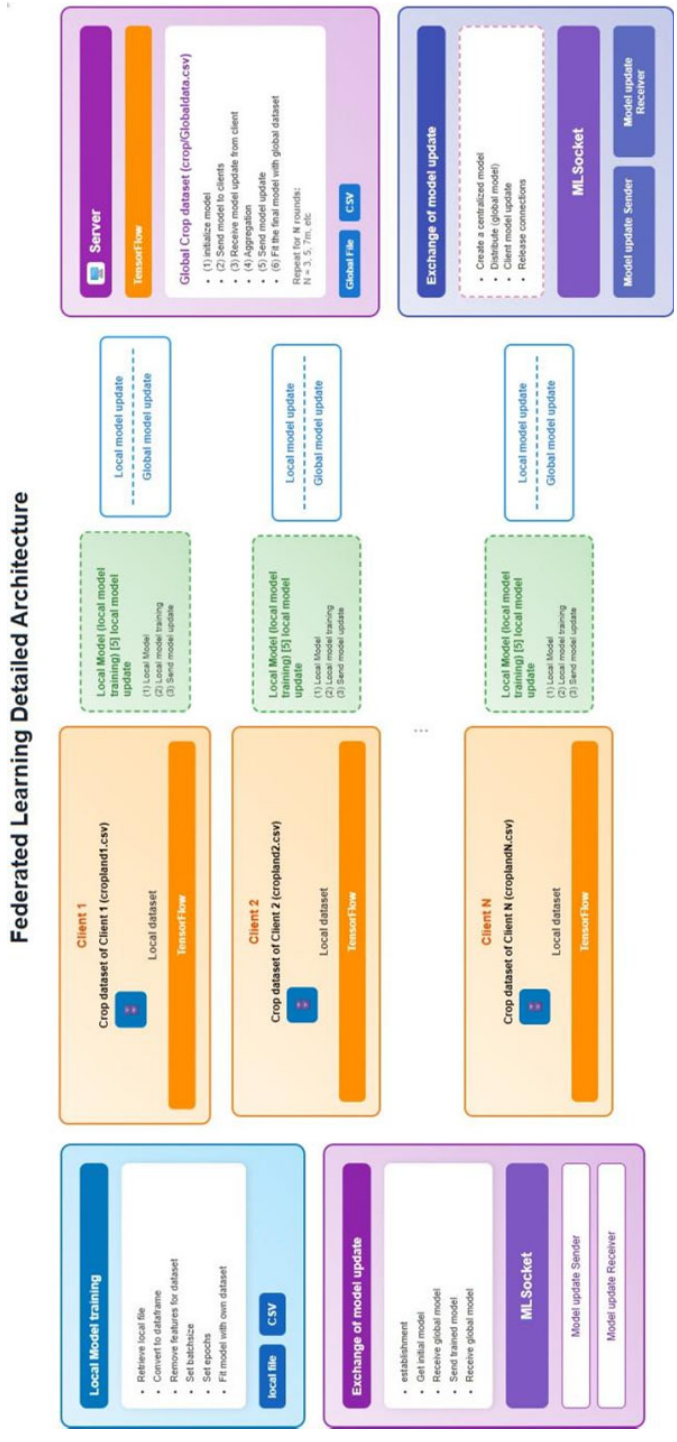


Figure 9 CFL-based crop yield prediction experimental implementation schematic (see online version for colours)



The experiment then used the same experimental setup to determine how well the distillation defence technique fares against adversarial sample attacks directed at the client. Incorporating the distillation defensive process into the local client training model did not impact its stability or convergence in this investigation. Figure 8 shows that the model converged with increasing rounds of training, which means that there are more rounds of training for clients.

We have shown the DFL design in Figures 9 and 10, which use ring and mesh topologies, respectively. It supports LSTM and GRU, and it also makes use of TensorFlow. Socket programming is utilised to construct the client-server concept and facilitate communication over the network. We have used ML Socket for the transfer of model updates. Secure Shell is the protocol of choice for encrypted communication. The dataset under consideration is partitioned and given to the server as the global dataset and to the clients as the local datasets. The LSTM-based architecture makes use of ReLU as the activation function for both the first and second dense layers. Adam is the optimiser in GRU-based designs and LSTM-based ones. At the same time, categorical cross-entropy is the loss function and softback is the activation function for the output layer in the former.

Figure 10 The experimental architecture for predicting crop yields using DFL and ring topology (see online version for colours)

Figure 11 displays the global model's LSTM prediction accuracy, recall, precision, and F1-score for the three situations that were considered. In scenario 3, the LSTM global model attained a prediction accuracy of 0.97, as can be seen in the image. The LSTM global model has a 0.95 accuracy in scenario 1 and a 0.96 accuracy in scenario 2. In Figure 11, we can see the worldwide loss of the third scenario's model that used LSTM. The LSTM-based CFL framework achieves a global loss of less than 0.0007 after ten cycles.

Figure 11 A global model's accuracy, precision, recall, and F1-score in LSTM-based CFL (see online version for colours)

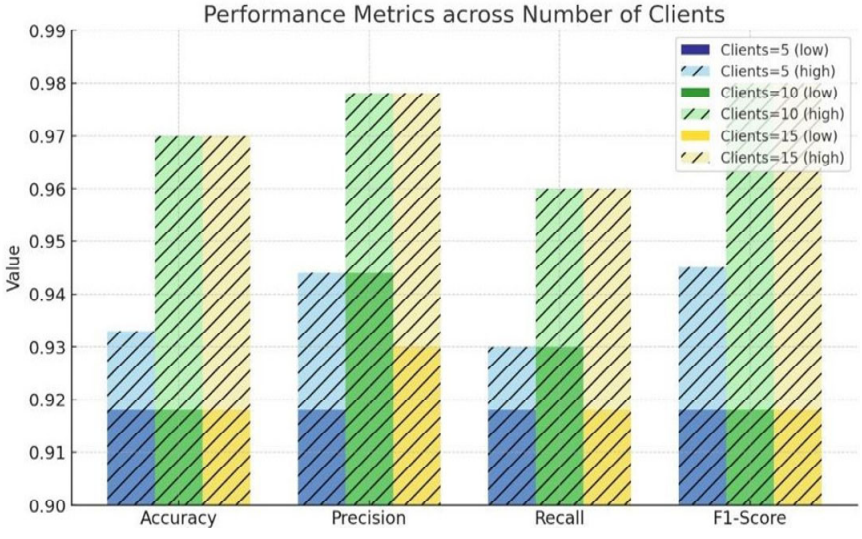
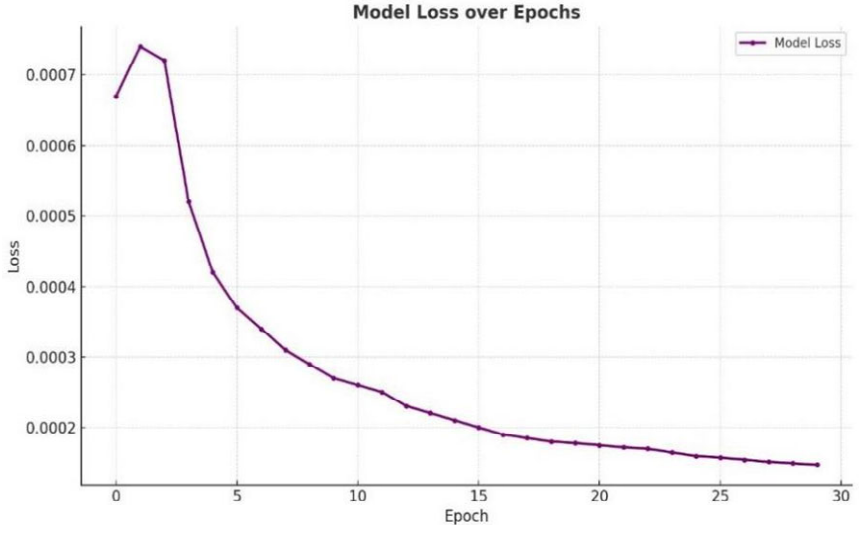


Figure 12 Model loss in LSTM-based CFL: a global perspective (see online version for colours)



The provided plot, titled 'model loss', illustrates the training progress of a machine learning model over 30 epochs. The curve shows a steep, rapid decrease in the loss value during the initial epochs (0 to ≈ 5), dropping from approximately 0.00075 to 0.00035, indicating that the model quickly learned the major patterns in the data. After this initial phase, the decrease in loss becomes gradual and continuous, showing the model fine-tuning its parameters until it converges to a very low final loss value of below 0.0002 by the 30th epoch. This decreasing, smooth curve is characteristic of a successful training run where the model's performance continuously improves without significant instability.

5 Conclusions

The importance of FL in protecting student privacy and providing individualised education at universities is emphasised in this study. The suggested frameworks strike a balance between data security, model accuracy, and fairness by incorporating advanced techniques like distillation defences, blockchain-based consensus and EADP. The experimental evaluation demonstrates that these approaches not only safeguard sensitive educational data but also enhance predictive performance and resilience against adversarial attacks. Overall, the findings indicate that FL provides a viable pathway toward secure, collaborative, and personalised educational ecosystems in the digital age. To build on these accomplishments, future research can investigate ways to integrate with new technologies like 6G and edge computing to enhance privacy protection and learning outcomes, as well as scalability in big heterogeneous networks.

Declarations

All authors declare that they have no conflicts of interest.

Funding: Hebei Provincial Higher Education Project: Research on Legal Safeguards for the Construction of Industry Colleges in Hebei's Undergraduate Universities (No. ZC2025341).

References

Afrose, S. (2021) 'Frequent itemsets mining with a guaranteed local differential privacy in small datasets', in *Proc. 33rd Int. Conf. Sci. Stat. Database Manag*.

Andrew, G. (2021) 'Differentially private learning with adaptive clipping', *Advances in Neural Information Processing Systems*, 6 December, Vol. 34, pp.17455–17466.

Balle, B. (2019) 'The privacy blanket of the shuffle model', in *Annu. Int. Cryptol. Conf.*, Springer, pp.638–667.

Chen, S. and Qi, X. (2025) 'Entropy-adaptive differential privacy federated learning for student performance prediction and privacy protection: a case study in Python programming', *Front. Artif. Intell.*, Vol. 8, p.1653437, DOI: 10.3389/frai.2025.1653437.

Chen, S.W. (2025) 'Entropy-adaptive differential privacy federated learning for student performance prediction and privacy protection: a case study in Python programming', *Front. Artif. Intell.*, 8 September, Vol. 8, p.1653437.

Chicaiza, J., Cabrera-Loayza, M.C., Elizalde, R. and Piedra, N. (2020) 'Application of data anonymization in learning analytics', in *Proceedings of the 3rd International Conference on Applications of Intelligent Systems*, 7 January, pp.1–6.

Fu, X. and Zhang, B. (2022) 'Federated graph machine learning: a survey of concepts, techniques, and applications', *SIGKDD Explor.*, 5 December, Vol. 24, No. 2, pp.32–47.

Galli, F. et al. (2022) *Group Privacy for Personalized Federated Learning*, arXiv preprint arXiv: 2206.03396.

Hridi, A.P., Sahay, R., Hosseinalipour, S. and Akram, B. (2024) 'Revolutionizing AI-assisted education with federated learning: A pathway to distributed, privacy-preserving, and debiased learning ecosystems', in *Proceedings of the AAAI Symposium Series*, 20 May, Vol. 3, No. 1, pp.297–303.

Javed, U. (2021) 'A review of content-based and context-based recommendation systems', *Int. J. Emerg. Technol. Learn.*, Vol. 16, No. 3, pp.274–306.

Javeed, D., Saeed, M.S., Kumar, P., Jolfaei, A., Islam, S. and Islam, A.N. (2023) 'Federated learning-based personalized recommendation systems: an overview on security and privacy challenges', *IEEE Transactions on Consumer Electronics*, 25 September, Vol. 70, No. 1, pp.2618–2627.

Kaushal, V. (2025) 'Weighted FedCOM: a communication-efficient approach to federated learning', *Evolv. Syst.*, Vol. 16, No. 1, p.27.

Mukherjee, A. (2025) 'Federated learning architectures: a performance evaluation with crop yield prediction application', *Software: Practice and Experience*, Vol. 55, No. 7, pp.1165–1184.

Salim, S., Turnbull, B. and Moustafa, N. (2021) 'A blockchain-enabled explainable federated learning for securing internet-of-things-based social media 3.0 networks', *IEEE Trans. Comput. Soc. Syst*., 28 December, Vol. 11, No. 4, pp.4681–4697.

Sengupta, D. (2024) 'FedEL: federated education learning for generating correlations between course outcomes and program outcomes for internet of education things', *Internet of Things* [online] https://doi.org/10.1016/j.iot.2024.101056.

Shawkat, M. et al. (2025) 'A robust and personalized privacy-preserving approach for adaptive clustered federated distillation', *Scientific Reports*, Vol. 15, No. 1, p.14069.

Wan, C. (2024) 'Research on privacy protection in federated learning combining distillation defence and blockchain', *Electronics*, 6 February, Vol. 13, No. 4, p.679.

Wang, F., Zhu, H., Srivastava, G., Li, S., Khosravi, M.R. and Qi, L. (2021) 'Robust collaborative filtering recommendation with user- item-trust records', *IEEE Trans. Comput. Soc. Syst.*, Vol. 9, No. 4, pp.986–996.

Wang, M. (2020) 'Security and privacy in 6G networks: new areas and new challenges', *Digit. Commun. Net.*, 1 August, Vol. 6, No. 3, pp.281–291.

Wen, D. (2022) 'Federated dropout – a simple approach for enabling federated learning on resource-constrained devices', *IEEE Wireless Commun. Lett.*, Vol. 11, No. 5, pp.923–927.

Zhu, F. (2024) 'A secure and fair federated learning framework based on consensus incentive mechanism', *Mathematics*, 30 September, Vol. 12, No. 19, p.3068.