Cluster Computing (2022) 25:2515–2526
https://doi.org/10.1007/s10586-021-03399-w

Privacy preserving personalized blockchain reliability prediction via federated learning in IoT environments

Jianlong Xu[1] · Jian Lin[1] · Wei Liang[2] · Kuan-Ching Li[3]

Received: 31 May 2021 / Revised: 20 July 2021 / Accepted: 17 August 2021 / Published online: 2 September 2021

Abstract
The integration of blockchain and the Internet of Things (IoT) is seen as having significant potential. In IoT Environments, Blockchain builds a trusted environment for IoT information sharing, where information is immutable and reliable. In particular, when edge devices are connected to a blockchain network, they need to be connected to reliable blockchain peers for synchronizing with valid data. Therefore, blockchain reliability prediction has gained attention owing to its ability to help users find highly reliable blockchain peers. Contextual information has been considered useful in many studies for generating highly personalized blockchain reliability predictions. However, these contextual factors are privacy-sensitive, and therefore disclosing them to third parties is risky. To address this challenge, we propose a privacy-preserving personalized blockchain reliability prediction model through federated learning neural collaborative filtering (FNCF) in IoT. Our model allows users to achieve user privacy protection without passing data to a third party and provides personalized predictions for users. We can also leverage the power of edge computing to enable a fast data processing capability and low latency required by IoT applications. Finally, our model was evaluated using a set of experiments based on real-world datasets. The experimental results show that the proposed model achieves high accuracy, efficiency, and scalability.

Keywords Privacy preservation · Federated learning · Blockchain · Reliability prediction · Internet of things · Edge computing

1 Introduction

The Internet of Things (IoT) has become widely pervasive in our daily lives [1]. There are large numbers of connected smart IoT devices, including cell phones, vehicles, home appliances, and sensors. At the same time, these smart devices generate large volumes of data. However, whether the data are being tampered with has become a difficult question to answer. Untrustworthy groups may modify the data for their own benefit, which makes the data they provide unreliable. Therefore, it is necessary to verify that the information has not been modified before using it. Blockchain is a point-to-point distributed ledger based on cryptography [2], which can enrich the IoT by providing a trusted sharing service, where information becomes reliable and traceable [3, 4]. Therefore, the integration of blockchain and IoT is considered to have significant potential. Notably, IoT devices can choose to offload their computationally intensive learning tasks to the edge, which satisfies the requirements for a fast processing power and low latency [5]. When edge devices join the blockchain network, they need to connect to highly reliable peers to obtain valid information. For example, in smart car areas, if users are unable to synchronize the latest road data from their blockchain peers, it can lead to a bad user experience

✉ Kuan-Ching Li
kuancli@pu.edu.tw

Jianlong Xu
xujianlong@stu.edu.cn

Jian Lin
20jlin3@stu.edu.cn

Wei Liang
weiliang99@hnu.edu.cn

[1] Shantou University, Shantou, China

[2] Hunan University, Changsha, China

[3] Providence University, Taichung, Taiwan

and possibly even traffic accidents. In addition, miners who need to solve preset proof-of-work puzzles of the blockchain system, if connected to unreliable blockchain peers, may synchronize to incorrect or old blocks and waste computational resources without being rewarded.

Therefore, there is an urgent need to choose a reliable blockchain peer. However, existing blockchain systems usually have many different peers. For example, with more than 6000 peers online at the same time in an Ethernet system, it is impossible for users to connect to all peers simultaneously to assess their reliability.

To solve this problem, a context-aware blockchain reliability prediction method is proposed. Studies have shown that it is beneficial to generate highly personalized blockchain reliability predictions for exploiting the context. Traditional approaches to context-aware block-chain reliability prediction collect decentralized user data in a central cloud for modeling purposes. However, collecting all such data for blockchain reliability prediction is often a challenging problem, as user data inevitably contain sensitive information and critical contextual information. (e.g., network settings and locations). This privacy threat limits the data collection from users to a central cloud, thus reducing the accuracy of blockchain reliability predictions. Moreover, with increasingly stringent data privacy protection legislation, such as the General Data Protection Regulation (GDPR) [6] and the Health Insurance Portability and Accountability Act (HIPAA) [7], the data flow is facing significant challenges.

To address this issue, we propose an effective privacy-preserving method for blockchain reliability prediction in IoT, FNCF, and a neural collaborative filtering method based on federated learning. Our contributions are as follows:

(1) We are the first to propose a privacy-preserving, multidimensional context-aware blockchain reliability prediction framework in IoT based on a federated learning framework.
(2) We propose a new shared parameter update mechanism to update the parameters shared by individual users.
(3) We choose a straightforward and practical approach that easily scales contextual information from one-dimensional to multidimensional.
(4) To evaluate the effectiveness of our approach, extensive experiments were conducted by employing real-world datasets, including two million test cases.

The rest of this paper is organized as follows. Section 2 introduces related work, and Sect. 3 describes the proposed methodology. In Sect. 4, we describes the experiments were conducted and an analysis of the results. Section 5 provides some concluding remarks and describes areas of future study.

2 Related work

In this section, we introduce studies related to blockchain reliability prediction, including traditional context-aware reliability prediction, and privacy preservation-based reliability prediction, as well as some existing research on blockchain peer reliability prediction.

2.1 Context-aware-based reliability prediction

Quality of service (QoS) is a set of non-functional [8, 9] attributes of a service, including reliability, throughput, response time, fault tolerance, and availability. Qos-based context-aware service prediction has attracted wide attention, providing users with a highly personalized web service prediction based on their previous experience or context from a large number of similar services. Recent studies [10–20] have found that contextual information about users and web services is also essential for generating more reliable predictions. Gao et al. [10] proposed a fuzzy c-mean combined with a neural collaborative filtering approach to prediction web services. In their approach, user and service context information is clustered using fuzzy c-mean, and the context information is then embedded into neural collaborative filtering to predict the QoS values using historical values. Wu et al. [12] presented a framework for implementing multi-attribute QoS prediction in a context using deep neural networks. In addition, Wu et al. [13] presented a context-sensitive matrix decomposition method (CSMF) based on the context. Moreover, Li et al. [14] presented a time-influence-aware collaborative filtering method. Yin et al. [15] also found that the accuracy of the prediction can be improved by mining a similar relationship between different users in the cyber-physical space of the network. By considering the invocation time and multiple spatial features, Zhou et al. [16] presented two generic spatio-temporal context-aware collaborative neural models for QoS prediction. In addition, Chowdhury et al. [17] presented a hybrid filtering combined with a hierarchical prediction mechanism to recommend web services by using user and service contextual information in combination, where the QoS values are accurately estimated by applying hierarchical neural regression. Xiong et al. [18] also presented a learning method to predict the QoS values of web services by using multidimensional contexts derived from past call histories. Moreover, Nagarajan et al. [19] presented a service-context-aware cloud agent. In their approach, the context information of the cloud service is considered to extract the

service details, and the service similarity is calculated according to the QoS value. In addition, Liu et al. [20] presented a context-aware multi-QoS prediction method for mobile edge computing.

2.2 Privacy-preservation-based reliability prediction

However, the above techniques are concerned with accuracy, and with the development of IoT technology, user data are growing at an explosive rate. Internet-connected smart devices, including cell phones, wearables, home appliances, sensors, and vehicles, have become part of everyday life in many parts of the world [21]. These data inevitably contain sensitive user data. Encryption techniques are applied to protect user privacy, which is a more intuitive approach [22, 23]. Badsha et al. [22] proposed a location-aware web service prediction framework, where the QoS and location of the user are protected using encryption-based techniques. In [24, 25], the QoS data are transformed into a small set of item indexes with little privacy using the local sensitive hashing algorithm (LSH), and the indexes are then used to find similar services. Zhang et al. [26] presented a distributed edge service QoS prediction model with privacy protection, where the edge server collects user quality records from different regions. They use the Laplace mechanism to provide users with a measurable degree of differentiated privacy protection. In addition, Liu et al. [27] showed that users directly apply local differential privacy to camouflage the observed QoS data and then send the data to a central server. However, the above approaches only consider user privacy issues for QoS features of one specific dimension, such as time or location, and do not consider user privacy issues for QoS features of more dimensions (e.g., location and network settings). It can be seen from the above that owing to the lack of a unified framework, existing methods used to protect user-sensitive contextual information dimensions are difficult to extend from one dimension to another.

For blockchain peer reliability prediction, Zheng et al. [28] recently proposed a hybrid blockchain reliability prediction model that extracts blockchain-related factors (e.g., a block hash, block height, and request response) from the request history and then generates personalized predictions for each user. Xu et al. [29] proposed a blockchain-reliability prediction model based on the matrix decomposition. However, they did not consider the context or privacy preservation using blockchain.

Based on this, a privacy-preserving multidimensional context-aware blockchain reliability prediction method is proposed for IoT based on federated learning. This is a scalable method used to protect more dimensions of user-context-sensitive information and appropriate blockchain peers selected for users.

3 Methoedology

In this section, a privacy-preserving multi-dimensional context-aware blockchain reliability prediction model for IoT based on federated learning is proposed, including the architecture and detailed steps.

3.1 Architecture

The architecture of blockchain reliability prediction in IoT is shown in Fig. 1. This includes the following three main roles:

(1) User: User nodes request blockchain peer data, train models with other users, and predict the reliability of the blockchain system.
(2) Blockchain Peers: The peers running in a P2P network store the data of the entire blockchain system.
(3) Central Server: The central server coordinates the training tasks of individual users.

This architecture was inspired by federated learning. Federated learning allows multiple users, called clients, to collaborate on their collective data to train a shared global model without moving data from the local device, which will effectively protect user privacy. As shown in Fig. 1, the reliability prediction steps for personalized blockchain peers with multidimensional context-aware and privacy protection in IoT are as follows: First, users request blockchain data from blockchain peers and save the data locally, converting the raw data into metrics, which are then used by the central server and users to jointly train the models. Finally, users can use the jointly trained models to predict and select reliable peers.

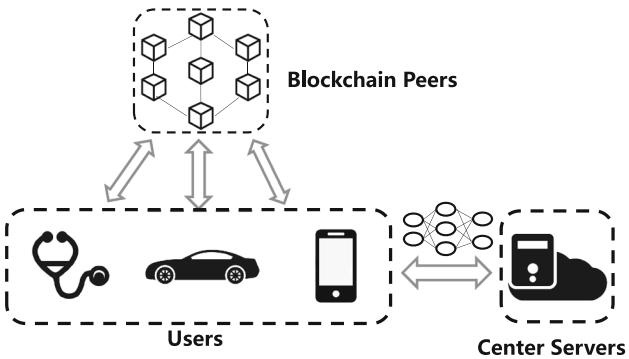

Fig. 1 Architecture of the blockchain reliability prediction

3.1.1 Data generation

Over a period of time, users (edge devices) at bulk randomly request the latest block data from blockchain peers, parse the data, and record the height and hash values for local storage. The complete record format contains seven data items: user IP, blockchain peer IP, request time, response time, bulk request time, block height, and block hash.

Users use the data collected to generate a success rate matrix. First, the central server uniformly sets two thresholds (MaxBlockBack, MaxRTT) and sends them to all users. MaxBlockBack represents the maximum tolerable block height of a blockchain peer that lags behind the block heights of other blockchain peers. MaxRTT represents the maximum round-trip time per peer. A user's request to a blockchain peer is successful if and only if (1) the block hash is correct in the corresponding block height on the main blockchain. (2) The difference between the block height and the highest block height in the batch was no greater than MaxBlockBack. (3) The round-trip time of the request does not exceed the MaxRTT. Otherwise, the request fails. The success rate of a user request to a blockchain peer is calculated by dividing the number of successful requests by the total number of requests. The total number of requests is equal to the number of failed requests plus the number of successful requests. The success rate formula for a request from user u to blockchain node i is thus as follows:

$$Success\,Rate_{ui} = \frac{Success\,Request_{ui}}{Success\,Request_{ui} + Fail\,Request_{ui}} \quad (1)$$

3.1.2 Model training and prediction

After generating the matrices, the user and the central server jointly train a global model for personalized blockchain reliability prediction. Model training is a key component of the overall architecture. We first introduce the general idea of the training process and then describe the process in detail in the following sections. We borrowed the idea of federation learning to protect sensitive user data. First, the central server chooses a global model to be trained. The central server then transmits the current model state to the participating users, and then trains the model locally based on its own data, which consists of the requested blockchain data records and the contextual information of the user. Finally, the central server aggregates the model results and updates to its global state. Based on the personalized reliability prediction results, the user can select a more reliable blockchain peer.

3.2 Personalized blockchain reliability prediction model

In this paper, we propose FNCF, a novel context-aware blockchain reliability prediction model with privacy preservation in IoT, built to produce personalized context-based prediction in an efficient manner. Our model can be represented formally as a five-tuple (user, ucontext, peers, QoS, and privacy), of which ucontext represents the user context and privacy represents the need for the model to provide privacy protection. The main goal of our proposed model is to perform any computation without revealing the raw sensitive information about the user to a third party. This sensitive information includes the user context, QoS values, similarity between users, and QoS prediction results. In our model, we also do not need to know which blockchain peers the user is connected to, and we only pass the training model throughout the computation process.

CF models the interaction between user and blockchain peer features. Based on the learned patterns, the model can provide users with personalized predictions. In integrated blockchain and IoT systems, the numbers of users N and peers M can be scaled to several million. However, limited by the network conditions, each user usually interacts with only a few peers, making the user-peer interaction matrix $\mathbf{R} \in \mathbb{R}_{N\times M}$ highly sparse. Neural collaborative filtering methods [30] have been shown to handle sparsity well and are able to scale the retrieval to large datasets. The model, which uses a multi-layer perceptron to learn a nonlinear interaction function of the user items, is formulated as follows:

$$\hat{y}_{ui} = f(Q^T v_u^U, P^T v_i^I | Q, P, \theta_f) \quad (2)$$

where $Q \in \mathbb{R}_{m\times k}, P \in \mathbb{R}_{n\times k}$ denotes the potential factor matrices of users and peers, respectively; θ_f denotes the model parameters of the interaction function f; and v_u^U, v_i^I denote the feature vectors of user u and peer i, respectively, which are binarized sparse vectors with one-hot encoding. We denote by Q_j^T the j-th column vector of the matrix Q^T, and use a new vector $\mathbf{x}$ to denote vector v_u^U. Then, $Q^T v_u^U$ is denoted as follows:

$$Q^T v_u^U = Q_1^T x_1 + Q_1^T x_2 + \cdots + Q_1^T x_m \quad (3)$$

where x_i denotes the i-th element of the vector $\mathbf{x}$. Because $\mathbf{x}$ is a one-hot vector, without a loss of generality, we assume that the u-th element of the vector is 1. That is $x_u = 1$. Then, $Q^T v_u^U$ is denoted as follows:

$$Q^T v_u^U = Q_u^T \cdot 1 \quad (4)$$

Equation (2) can then be rewritten as

$$\hat{y}_{ui} = f(Q_u^T \cdot 1, P^T v_i^I | Q_u, P, \theta_f) \tag{5}$$

At this point, we generate a new prediction model for all users, which have their own private parameters Q_u and parameters shared among them, P, θ_f. In this way we transform the problem into a parameter sharing solution. We consider each user training model as a training task. Letting the parameters $\theta = (P, \theta_f, Q_1, Q_2, \cdots, Q_m)$, the parameters of the multi-task model are optimized during joint training with a loss [31]:

$$L(\theta) = \sum_{u=1}^{m} \lambda_u \sum_{i=1}^{n} L_{ui}(\hat{y}_{ui}, y_{ui}) \tag{6}$$

where $L_{ui}(\cdot)$ is the loss function for the i-th sample of task u, and λ_u is the weight of task u. In practice, λ_u is often considered as a hyperparameter to be tuned, but is assigned as 1 in this study.

We propose a parameter sharing training algorithm 1 (FNCF) with generality based on federal learning, which can effectively solve the above optimization problem. The key idea is to carry out model updates such that the user's private data and private interaction data are not uploaded to the central server. A central server coordinates users to train the model together, as is illustrated in Fig. 2. At the beginning of each round, the central server transmits the current model state to the participating users. The model parameters are divided into two parts: one for the shared parameters, P, θ_f, and one for the private parameters. $Q_u, u \in \{1, 2, ..., m\}$. Because the central server does not have access to the specific values of these private parameters, it can only randomly assign values to these private parameters before passing the model to the participating users. Each user then overwrites the randomized private parameters using the private parameters recorded locally, and updates the global model using the local data. Then, instead of directly uploading the updated model to the central server, the user first saves the private parameters of the updated global model locally. It then randomizes the private parameters of the global model, and then returns the updated model. Therefore, the uploaded global model parameters include both randomized private parameters and modified shared parameters. The purpose of this is to prevent third parties from inferring sensitive data about the user through private and shared parameters. Finally, the central server aggregates these updates from the participating users and updates the global model. The model iterates this process until convergence is achieved.

The computational effort of the entire model training is controlled by three key parameters: C, the proportion of users attending the training per round; E, the number of local training sessions per user per round on their local dataset; and B, the local batch size used for the user updates. For a user with n_u local samples, the number of updates for local training per round is as follows:

$$u_u = E\frac{n_u}{B} \tag{7}$$

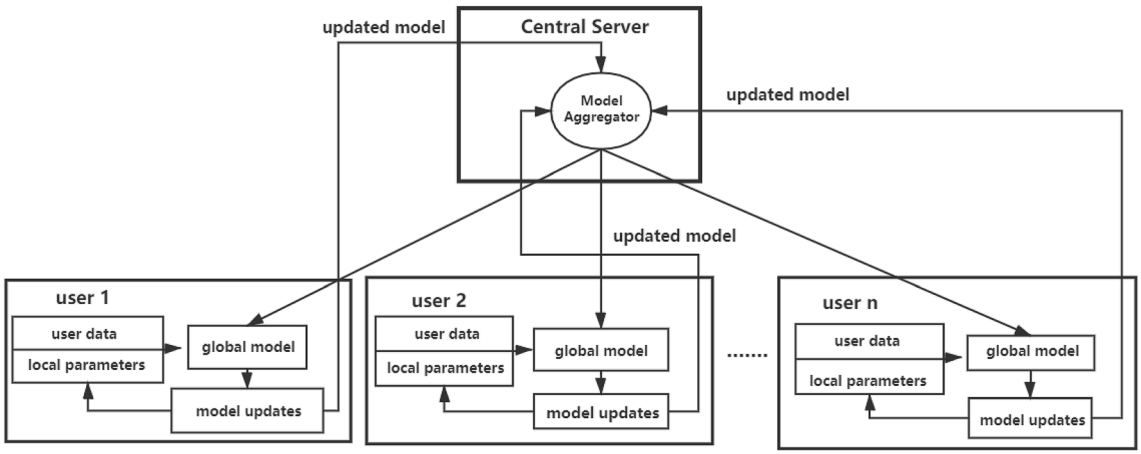

Fig. 2 Based on collaborative filtering in the federated learning paradigm. Each user updates the global model from the central server using locally sensitive data (including QoS, user context feature) and private model parameters, and then the central server aggregates the user's updated model

Algorithm 1 FNCF Algorithm. m: total number of users; u: index of users; C: proportion of total number of users; B: local batch size; E: number of local epochs; P, θ_f: shared parameters; Q_u: u-th user pravate parameters; $local\_Q_u$: u-th user private parameters saved locally; η: learning rate; P_u: local train dataset. n_u : the number of samples for user u; n: the sum of the number of samples for all users

Server executes:
initialize $P, \theta_f, Q_u, u \in \{1, 2, ..., m\}$
$w_t \leftarrow (Q_1, P, \theta_f)$ //the values of vector $Q_u, u \in \{1, 2, ..., m\}$ are randomized so that Q_1 can be used instead of the other vectors
$m \leftarrow \max(C * m, 1)$
for each round t = 1, 2, . . . **do**
  $S_t \leftarrow$random set of m users
  for each user u $\in S_t$ in parallel **do**
    $w_{t+1}^u \leftarrow$ **UserUpdate**(u, w_t)
  end for
  $w_{t+1} \leftarrow \sum_{u=1}^{m} \frac{n_u}{n} w_{t+1}^u$
end for

UserUpdate(u, w): // run on user u
$(Q_u, P, \theta_f) \leftarrow w$
$Q_u \leftarrow local\_Q_u$
$\beta \leftarrow$ (split P_u into batches of size B)
for each local epoch i from 1 to E **do**
  for batch b $\in \beta$ **do** //gradient descent
    $L_u = \frac{\sum_{i=1}^{B} L_{ui}(f(Q_u^T \cdot 1, P^T v_i^I, c_k | Q_u, P, \theta_f), y_{ui})}{B}$
    $Q_u \leftarrow Q_u - \eta \frac{dL_u}{dQ_u}$
    $P \leftarrow P - \eta \frac{dL_u}{dP}$
    $\theta_f \leftarrow \theta_f - \eta \frac{dL_u}{d\theta_f}$
  end for
end for
$\text{local\_}Q_u \leftarrow Q_u$
$Q_u \leftarrow$ random value
$w \leftarrow (Q_u, P, \theta_f)$
return w to server

The training strategy in [31] is a special case of our proposed approach when the number of batch tasks is C = 1 and u_u = 1.

To provide more personalized recommendations, the user's context is taken into account. We believe that users in similar environments have similar preferences. We take advantage of the scalability of the input layer of neural collaborative filtering to easily add more contextual features as inputs to the model, which is constructed as follows:

$$\hat{y}_{ui} = f(Q_u^T \cdot 1, P^T v_i^I, c_u | Q_u, P, \theta_f) \tag{8}$$

where c_u is the contextual feature vector of the u-th user. These contexts include, but are not limited to, the user's country, city, and network service providers. The local training process is shown in Fig. 3. Before obtaining these dense features, we generate raw data into model training data. We divide the raw data into two categories and generate them separately: one for the user ID and

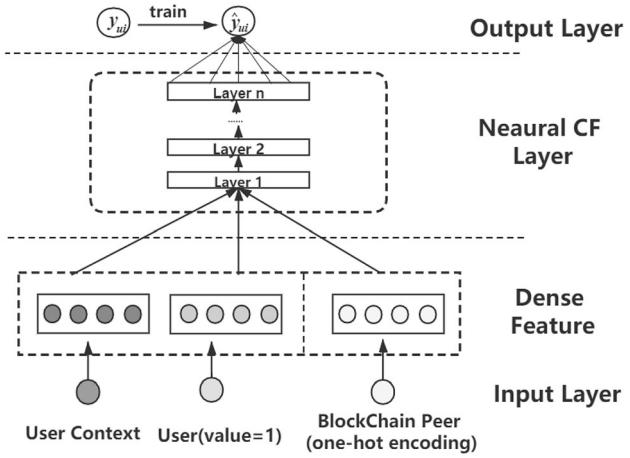

Fig. 3 Multi-dimensional context-aware neural collaborative filtering model

blockchain peer ID with unique identifiers, and the other for potentially duplicated user contexts, for example, two users belong to the same country.

(1) User ID & Blockchain peer ID: Each blockchain peer has a corresponding unique ID, which we use as a feature. We use the IDs as indexes in the embedding, which allows us to learn the vector representation of each list and encode their respective unique properties. For user ID, according to Eq. (8), we uniformly assign a value of 1 to the user ID.

(2) User Context: We first pass the user context through a hash function that converts the string into an ID, e.g., hash ("Shanghai") → 79524613, to build our classification properties, use this ID as a seed for a random number, and then use this random seed to randomly generate a dense vector that is input into the neural collaborative filtering model. This has two benefits: As the first benefit, different users can share the input layer vector without the need to communicate. As the second advantage, when there are too many types of contextual information, less space is needed for storage compared to a one-hot representation.

Once we have these dense features, we stitch these dense features into a long vector input into a neural filtering layer consisting of a multilayer perceptron, and then output the predicted success rate $\hat{y}$. If we want to observe the impact of different contexts on the performance of the model, we can simply stitch together a dense vector of the corresponding contexts.

4 Experiment

In this section, we describe relevant experiments conducted to answer the following questions:

Question 1: Can a personalized federated learning model improve the prediction performance of the QoS?

Question 2: Does selecting more users for training per round help reduce the number of training communication rounds?

Question 3: Does increasing the number of local computations of the users per round help reduce the number of training communication rounds?

Question 4: How does the configuration of the deep neural model affect the prediction performance?

Question 5: Do more contextual factors help improve the QoS prediction performance?

4.1 Data set

To evaluate the blockchain reliability prediction performance in IoT environments, we used the real dataset presented in [28], which consists of 100 users and 200 blockchain peers. The users in this dataset are distributed globally in 15 countries around the world, and the blockchain peers are distributed across 21 countries. The dataset includes over two million test cases. The dataset also provides contextual information, including the region, autonomous system, geographical coordinates, and network service provider of the users, among other factors.

4.2 Evaluation metrics

The root mean square error (RMSE) metric is the standard deviation of the prediction error and is used to measure the prediction performance of different methods. A smaller RMSE indicates a better model performance. The formula for the RMSE is as follows:

$$RMSE = \sqrt{\frac{\sum_{u,i}(Success\,Rate_{ui} - \widehat{Success\,Rate}_{ui})^2}{N}} \quad (9)$$

where $\widehat{SuccessRate}_{ui}$ is the success rate of the prediction of user u's request to blockchain peer i, $SuccessRate_{ui}$ is the real success rate of user u's requests to blockchain peer i, and N is the number of all test samples.

4.3 Data processing

Adapt to different scenarios by modifying the MaxRTT and MaxBlockBack values. We set MaxRTT = 1000 ms and MaxBlockBack = 12 to evaluate the accuracy for scenarios with timeliness to confirm the blockchain data. For descriptive purposes, we named the scenario "blockchain system 1." We then set MaxRTT to 5000 ms and MaxBlockBack to 100 to evaluate the accuracy under scenarios that tolerate little real-time confirmation block lag. For descriptive purposes, we labeled the scenario "blockchain system 2." Of particular note, we selected the service provider, autonomous system, country, city, latitude, longitude, and time zone as contextual information for the user.

4.4 Parameter settings

To verify the effectiveness of the model under different dataset densities, we randomly selected a certain percentage of QoS values from each user's dataset as the training set, namely, 30%, 50%, 65%, 80%, and 95%. For example, a density of 30% indicates that 30% of the QoS values from each user's dataset of requested blockchain peer results are randomly selected as the training set, and the remaining 70% of the QoS dataset are used as the test set. The dimensionality of the dense feature vector is set to 64. Our model uses the AdamW optimizer; in addition, the learning rate is set to 0.01, and the exponential decay rate is set to 0.001. Finally, L1 Loss is used as the loss function.

4.5 Performance comparison (Question 1)

This section focuses on the first problem. To validate the effectiveness of our proposed FNCF method, we compared it with other blockchain reliability prediction methods.

(1) UPCC [32]. This method finds similar users to achieve a reliability prediction.
(2) IPCC [33]. This method finds similar items to achieve a reliability prediction.
(3) UIPCC [34]. This method is a combination of UPCC and IPCC.
(4) HBRP [28]. The method predicts reliability by modelling the relationship between reliability-related factors and success rates through linear regression.
(5) BSRPF [29]. The method conducts blockchain reliability prediction through matrix factorization.

Tables 1 and 2 show the prediction errors of the various prediction methods under Blockchain System 1 and Blockchain System 2, respectively. From Table 1, we can see that FNCF has significantly smaller prediction errors at different training set densities compared to UPCC, IPCC, UIPCC, HBRP, and BSRPF. From Table 2, we can see that we achieved a better accuracy than all other methods at 30% and 95% of the density matrix. In addition, the accuracy is slightly lower at 50%, 65%, and 80% of the density matrix compared to HBRP, but better than the other methods. This effect is due to the fact that HBRP uses a traditional centralized training method and does not

Table 1 Performance comparison of different approaches in blockchain system 1

Method	Density = 30%	Density = 50%	Density = 65%	Density = 80%	Density = 95%
UPCC	0.3627	0.3591	0.3566	0.3552	0.3559
IPCC	0.1009	0.0949	0.0919	0.0908	0.0920
UIPCC	0.1053	0.0994	0.0963	0.0951	0.0961
HBRP	0.1042	0.0925	0.0894	0.0879	0.0824
BSRPF	0.0946	0.0867	0.0810	0.0746	0.0680
FNCF	**0.0803**	**0.0700**	**0.0651**	**0.0601**	**0.0501**

The value with the smallest prediction error is marked in bold

Table 2 Performance comparison of different approaches in blockchain system 2

Method	Density = 30%	Density = 50%	Density = 65%	Density = 80%	Density = 95%
UPCC	0.4595	0.4585	0.4569	0.4564	0.4574
IPCC	0.0921	0.0887	0.0842	0.0806	0.0774
UIPCC	0.1022	0.0993	0.0954	0.0923	0.0895
HBRP	0.0648	**0.0562**	**0.0524**	**0.0494**	0.0433
BSRPF	0.0890	0.0851	0.0809	0.0795	0.0694
FNCF	**0.0643**	0.0574	0.0527	0.0533	**0.0425**

The value with the smallest prediction error is marked in bold

provide privacy protection; however, FNCF uses personalized federation learning to provide privacy protection. From the above analysis, we can conclude that the accuracy of the proposed FNCF is significantly better than that of the other methods in the case of the high requirements in confirming the blockchain data (e.g., cryptocurrency wallets and cryptocurrency exchanges) as compared to QoS prediction methods without privacy protection, and is better than most methods under daily use with a high tolerance for block lag and latency (e.g., regular blockchain users); however, in a density matrix, it is slightly lower than the HBRP within the range of 50% to 80%.

4.6 Increasing parallelism (Question 2)

Considering the limitations of IoT device communication, and to avoid affecting the performance of the IoT devices, only some users participate in each round of training. Thus, we experiment with the ratio C of the devices in Blockchain System 1, which indicates the number of devices trained in parallel during each round of communication. Table 3 shows the effect of different C values on the performance of the FNCF model under Blockchain System 1. We use the number of communications required to achieve the specified test accuracy as an indicator to evaluate the model performance. For B = ∞ (all samples are selected for each local update), we can see that the acceleration becomes increasingly pronounced as C increases. By contrast, using a smaller batch size of B = 10, the acceleration becomes slower as C becomes larger in terms of using C $\geq$ 0.1. Considering the computational efficiency and stability, we set C = 0.1 in the subsequent experiments.

Table 3 Effect of user ratio C with E = 5

C	B = ∞	B = 10
0.0	1060	1331
0.1	355(2.8X)	279(4.7X)
0.2	333(3.1X)	384(3.4X)
0.5	249(4.2X)	400(3.3X)
1.0	219(4.8X)	511(2.6X)

Note that C = 0.0 indicates that only one device is selected per round; since the blockchain system 1 has a total of 100 users, for example, 0.1 corresponds to 10 users here. Each table entry represents the number of communications required to achieve the specified prediction error (RMSE $<$0.090, RMSE: root mean square error), and the acceleration relative to the C = 0.0 baseline

4.7 Increasing the computation for each user (Question 3)

In this section, we set C = 0.1 and experiment using a different u. In addition, u is used to measure the local user computations each round. According to u = E (n/B), E or decrease B to increase the number of local computation. Table 4 shows the impact of different values of E and B on the model in Blockchain System 1. We sort each row in Table 4 based on the size of u. We can see that the overall trend of the number of communications decreases and then increases as u increases. To observe the changes more

Table 4 Number of communication rounds to reach the specified error at different u (RMSE <0.095, RMSE: root mean square error)

E	B	u	Communication rounds
1	∞	1	1383
1	30	2	1105(1.3X)
5	∞	5	221(6.3X)
1	10	6	1154(1.2X)
5	30	10	196(7.1X)
20	∞	20	125(12.0X)
5	10	30	219(6.3X)
20	30	40	207(6.7X)
20	10	120	>2000

The u column gives u = E (n/B), the number of user updates per communication round

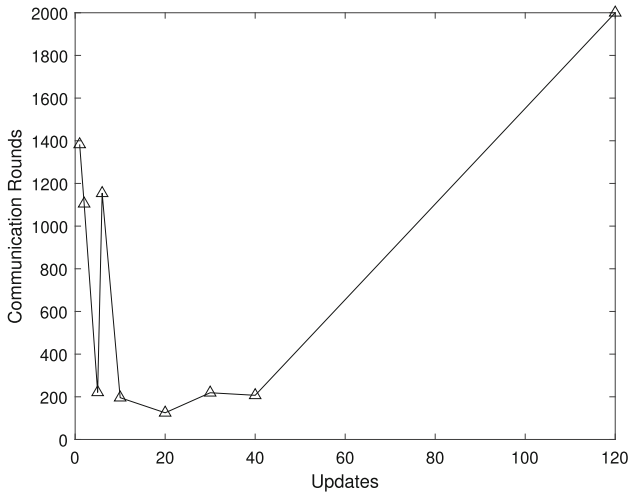

Fig. 4 Communication rounds with different updates

easily, we visualize Table 4 in Fig. 4. For Blockchain System 1, the speedup (6.3-12.0X) is more significant when u is increased from 10 to 40.

4.8 Parameter analysis(Question 4)

To answer question 4, we describe a series of conducted experiments in this section.

4.8.1 Impact of dimensionality and density

Dense vectors are used to represent the features of users, blockchain peers, and the user context. The number of dimensions refers to the length of the dense vector. We set the density to $\{30\%, 50\%, 65\%, 80\%, 95\%\}$ and set the number of dimensions to $\{8, 16, 32, 64\}$. Figs. 5 and 6

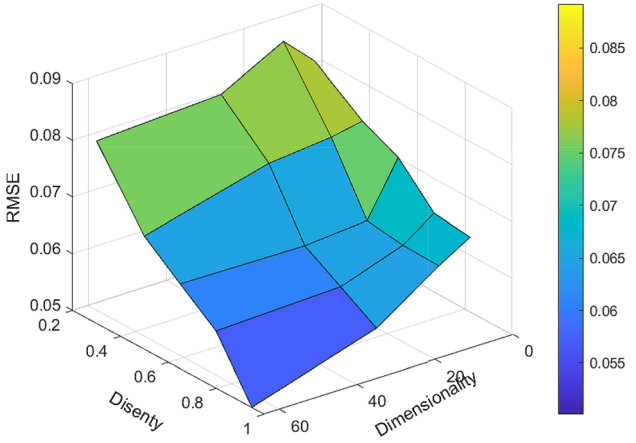

Fig. 5 Impact of dimensionality and matrix density in blockchain system 1

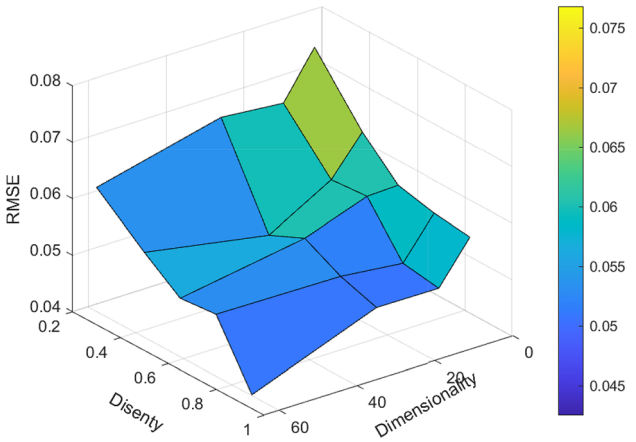

Fig. 6 Impact of dimensionality and matrix density in blockchain system 2

show the prediction performance of FNCF in different blockchain systems with different dimensions and matrix densities, where RMSE is the evaluation metric. In Fig. 5, under the condition of a fixed dimensionality of the dense features, the prediction error decreases with an increase in the density of the dataset. By contrast, under the condition of a fixed dataset density, overall, the prediction error decreases with an increase in the dimensionality of the dense features. These observations are generally consistent with our expectation that the prediction error decreases as the dimensionality and density increase. In Fig. 6, under the condition of a fixed dimensionality of the dense features, the prediction error decreases with an increase in the dataset density. Increasing the number of dimensions improves the prediction accuracy when the matrix density is approximately 30%, 65%, and 95%, respectively. Based on the above results, a larger number of dimensions usually leads to a better performance.

Table 5 Impact of hidden layer configuration

HL	RMSE
{32}	0.0818
{64}	0.0803
{96}	0.0784
{128}	0.0774
{256}	0.0782
{32, 16}	0.0811
{64, 32}	0.0930
{96, 48}	0.0945
{128, 64}	0.0913
{256, 128}	0.0914

4.8.2 Impact of hidden layers

Because few studies have been conducted on learning user-blockchain peer interaction functions using neural networks under a federation setting, we wanted to know whether using deep network structures would be beneficial for blockchain reliability prediction tasks. The proposed FNCF model was based on the MLP model. The classical MLP model consists of an input layer, hidden layers, and an output layer, and we can change the predictive performance of the model by adjusting the number of hidden layers and the width of the hidden layer. To this end, we further investigated the MLP with different numbers of hidden layers and each hidden layer (HL) containing a different number of neurons under the federation setting. For example, the structure 32, 16 indicates that the multilayer perceptron model contains two hidden layers, counting from small to large, with the first hidden layer consisting of 32 neurons, and the second hidden layer consisting of 16 neurons. In addition, we set the training set to Blockchain System 1 and the density of the training set to 30%. The experimental results are summarized in Table 5. The experimental results show that adding hidden layers increases the prediction error. This may be because hidden layers that are too deep can introduce a model overfitting. For MLP models containing only one hidden layer, the inclusion of more neurons means more accurate predictions. However, more neurons also mean a longer training time.

4.9 Increasing the number of contextual factors (Question 5)

In this section, we describe experiments conducted to investigate the effect of the number of contextual factors on QoS predictions. Here, we focus on seven factors: ISP (user's ISP), AS (user's autonomous system), TZ (user's time zone), R (user's region or country), C (user's city), LAT (latitude of user's geographic location), and LON (longitude of user's geographic location). We add contextual factors to build the prediction sub-models. Our prediction sub-models are (1) a sub-model without any context factor, labeled $\{-\}$; (2) a sub-model containing one context factor for the country, labeled $\{R\}$; (3) a sub-model containing three context factors for the country, labeled $\{R, AS, ISP\}$; and (4) a sub-model containing all context factors for the country, labeled $\{ALL\}$. We then compared the results of these sub-models. In addition, we set the training set to Blockchain System 1 and the density of the training set to 30%. The experimental results are shown in Table 6, where we observe that, as the number of contextual factors increases, the predictive performance of the model improves accordingly.

Table 6 Prediction performances (RMSE) of FNCF with different number of contextual factors

Sub-model	RMSE
$\{-\}$	0.0861
$\{R\}$	0.0849
$\{R, AS, ISP\}$	0.0821
$\{ALL\}$	0.0803

5 Summary and future work

In this paper, we propose FNCF, a personalized federated learning framework for blockchain reliability prediction with data privacy protection in IoT environments. FNCF can be learned for all users without access to the original IoT device data. Unlike existing studies in the field of blockchain reliability prediction in IoT environments, we conducted an experimental analysis of multidimensional user contexts with data privacy protection. The experimental results demonstrate the accuracy, efficiency, and scalability of our approach. In the future, we intend to introduce differential privacy and multiparty computing to provide stronger privacy guarantees.

Acknowledgements This research was financially supported by the National Natural Science Foundation of China (No.61702318), 2020 Li Ka Shing Foundation Cross-Disciplinary Research Grant (No.2020LKSFG08D), the Shantou University Scientific Research Start-up Fund Project (No.NTF18024), and in part by 2019 Guangdong province special fund for science and technology ("major special projects + task list") project (No. 2019ST043).

References

1. Liang, W., Ji, N.: Privacy challenges of IoT-based blockchain: a systematic review. Clust. Comput. (2021). https://doi.org/10.1007/s10586-021-03260-0

2. Liang, W., Xiao, L., Zhang, K., Tang, M., He, L., Li, K.: Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems. IEEE Internet Things J. (2021). https://doi.org/10.1109/JIOT.2021.3053842
3. Reyna, A., Martín, C., Chen, J., Soler, E., Díaz, M.: On blockchain and its integration with IoT. Challenges and opportunities. Future Gener. Comput. Syst. **88**, 173–190 (2018). https://doi.org/10.1016/j.future.2018.05.046
4. Liang, W., Zhang, D., Lei, X., Tang, M., Li, K., Zomaya, A.: Circuit copyright blockchain: blockchain-based homomorphic encryption for IP circuit protection. IEEE Trans. Emerg. Top. Comput. (2020). https://doi.org/10.1109/TETC.2020.2993032
5. Wu, Q., He, K., Chen, X.: Personalized federated learning for intelligent IoT applications: a cloud-edge based framework. IEEE Open J. Comput. Soc. **1**, 35–44 (2020). https://doi.org/10.1109/OJCS.2020.2993259
6. Voigt, P., Bussche, A.: The eu general data protection regulation (gdpr). A practical guide, 1st edn. Springer, Cham (2017)
7. Annas, G., et al.: Hipaa regulations-a new era of medical-record privacy? N. Engl. J. Med. **348**(15), 1486–1490 (2003)
8. Ran, S.: A model for web services discovery with QoS. ACM Sigecom Exch. **4**(1), 1–10 (2003). https://doi.org/10.1145/844357.844360
9. Liang, W., Li, Y., Xu, J., Qin, Z., Li, K.-C.: QoS prediction and adversarial attack protection for distributed services under DLaaS. IEEE Trans. Comput. (2021). https://doi.org/10.1109/TC.2021.3077738
10. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang, X.: Context-aware QoS prediction with neural collaborative filtering for internet-of-things services. IEEE Internet Things J. **7**(5), 4532–4542 (2020). https://doi.org/10.1109/JIOT.2019.2956827
11. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
12. Wu, H., Zhang, Z., Luo, J., Yue, K., Hsu, C.: Multiple Attributes QoS prediction via deep neural model with contexts. IEEE Trans. Serv. Comput. (2018). https://doi.org/10.1109/TSC.2018.2859986
13. Wu, H., Yue, K., Li, B., Zhang, B., Hsu, C.: Collaborative QoS prediction with context-sensitive matrix factorization. Future Gener. Comput. Syst. (2018). https://doi.org/10.1016/j.future.2017.06.020
14. Li, J., Wang, J., Sun, Q., Zhou, A.: Temporal influences-aware collaborative filtering for QoS-based service recommendation. In: 2017 IEEE International Conference onServices Computing (SCC), pp. 471–474. IEEE (2017). https://doi.org/10.1109/SCC.2017.67
15. Yin, Y., Yu, F., Xu, Y., Yu, L., Mu, J.: Network location-aware service recommendation with random walk in cyber-physical systems. Sensors **17**(9), 2059 (2017). https://doi.org/10.3390/s17092059
16. Zhou, Q., Wu, H., Yue, K., Hsu, C.: Spatio-temporal context-aware collaborative QoS prediction. Future Gener. Comput. Syst. (2019). https://doi.org/10.1016/j.future.2019.05.024
17. Chowdhury, R.R., Chattopadhyay, S., Adak, C.: CAHPHF: context-aware hierarchical QoS prediction with hybrid filtering. IEEE Trans. Serv. Comput. (2020). https://doi.org/10.1109/TSC.2020.3041626
18. Xiong, W., Wu, Z., Li, B., Gu, Q.: A learning approach to QoS prediction via multi-dimensional context. In: 2017IEEE international conference on web services (ICWS),pp. 164–171. IEEE (2017). https://doi.org/10.1109/ICWS.2017.29
19. Nagarajan, R., Thirunavukarasu, R.: A service context-aware QoS prediction and recommendation of cloud infrastructure services. Arab. J. Sci. Eng. **45**, 2929–2943 (2020). https://doi.org/10.1007/s13369-019-04218-6
20. Liu, Z., Sheng, Q., Zhang, W., Chu, D., Xu, : X.: Context-aware multi-QoS prediction for services in mobile edge computing. In: 2019 IEEE international conference on services computing (SCC), pp. 72–79. IEEE (2019). https://doi.org/10.1109/SCC.2019.00024
21. Nguyên, T.T, Xiao, X., Yang, Y., Hui, S., Shin, H., Shin, J.: Collecting and analyzing data from smart device users with local differential privacy. Preprint at arXiv:1606.05053 (2016)
22. Badsha, S., et al.: Privacy preserving location-aware personalized web service recommendations. IEEE Trans. Serv. Comput. (2018). https://doi.org/10.1109/TSC.2018.2839587
23. Badsha, S., Yi, X., Khalil, I., Liu, D., Nepal, S., Lam, K.: Privacy preserving user based web service recommendations. IEEEIEEE Access **6**, 56647–56657 (2018). https://doi.org/10.1109/ACCESS.2018.2871447
24. Qi, L., et al.: Privacy-aware data fusion and prediction with spatial-temporal context for smart city industrial environment. IEEE Trans Ind. Inform. **17**(6), 4159–4167 (2021). https://doi.org/10.1109/TII.2020.3012157
25. Chi, X., Yan, C., Wang, H., Rafique, W., Qi, L.: Amplified LSH-based recommender systems with privacy protection. Pract. Exp. Concurr. Comput. (2020). https://doi.org/10.1002/CPE.5681
26. Zhang, Y., Pan, J., Qi, L., He, Q.: Privacy-preserving quality prediction for edge-based IoT services. Future Gener. Comput. Syst. (2021). https://doi.org/10.1016/j.future.2020.08.014
27. Liu, S., et al.: Privacy-preserving collaborative web services QoS prediction via differential privacy. In: Chen, L., Jensen, C., Shahabi, C., Yang, X., Lian, X. (eds.) Web and big data. APWeb-WAIM 2017. Lecture notes in computer science, vol. 10366. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63579-8_16
28. Zheng, P., Zheng, Z., Chen, L.: Selecting Reliable Blockchain Peers via Hybrid Blockchain Reliability Prediction, arXiv preprint arXiv:1910.14614 (2019)
29. Xu, J., Zhuang, Z., Wang, K., Liang, W.: High-accuracy reliability prediction approach for blockchain services under BaaS. In: Zheng, Z., Dai, H.N., Fu, X., Chen, B. (eds.) Blockchain and trustworthy systems. BlockSys 2020. Communications in computer and information science, vol. 1267. Springer, Singapore (2020)
30. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.: Neural Collaborative Filtering. In: Proceedings of the 26th international conference on world wide web (WWW '17). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, pp. 173–182. https://doi.org/10.1145/3038912.3052569
31. Sun, T., Shao, Y., Li, X., Liu, P., Yan, H., Qiu, X., & Huang, X.: (2020). Learning sparse sharing architectures for multiple tasks. Proc AAAI Conf Artif Intell. **34**(05), 8936–8943 (2020). https://doi.org/10.1609/aaai.v34i05.6424
32. Shao, L., Zhang, J., Wei, Y., Zhao, J., Xie, B., Mei, H.: Personalized QoS prediction for web services via collaborative filtering. In: Proceedings of the IEEE international conferenceon web services (ICWS), pp. 439–446. IEEE (2007)
33. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Item-based-collaborative filtering recommendation algorithms. In: Proceedings of the10th international world wide web conference (WWW), pp. 285–295 (2001)
34. Zheng, Z., Ma, H., Lyu, M.R., King, I.: QoS-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4**(2), 140–152 (2011)

Jianlong Xu is currently a lecturer in the College of Engineering, Shantou University, China. He received his Ph.D. degree from the South China University of Technology in 2013 and then studied as a postdoctoral fellow at Shenzhen Research Institute, the Chinese University of Hong Kong. His research interests include service computing, Internet of Things, Blockchain, and information security.

Jian Lin is a graduate student at Shantou University, China. His research interests include service computing and Blockchain.

Wei Liang is currently an Associate Professor at the College of Computer Science and Electronic Engineering, Hunan University, China. He received his Ph.D. degree at Hunan University in 2013 and was a postdoctoral scholar at Lehigh University, USA, during 2014–2016. He served as Application Track Chair of IEEE Trustcom 2015, a Workshop Chair of IEEE Trustcom WSN 2015, and IEEE Trustcom WSN 2016. He has published more than 110 journal/conference papers such as IEEE Transactions on Industrial Informatics, IEEE Transactions on Emerging Topics in Computing, IEEE Transactions on Computational Biology and Bioinformatics, and IEEE Internet of Things Journal. His research interests include Blockchain security technology, Networks Security Protection, embedded system and Hardware IP protection, Fog computing, and Security management in WSN.

Kuan-Ching Li is a Distinguished Professor at Providence University, Taiwan. He is a recipient of distinguished and chair professorships from universities in several countries and awards and funding support from a number of agencies and high–tech companies. Besides publishing numerous journal articles, book chapters, and refereed conference papers, he is co-author/co-editor of more than 25 technical professional books published by CRC Press/Taylor & Francis, Springer, McGraw-Hill, and IGI Global. His research interests include parallel and distributed computing, Big Data, and emerging technologies. He is a senior member of the IEEE and a fellow of the IET.