Training Node Screening in Decentralized Trusted Federated Learning

Hao Wang[1], Jiahua Yu[2], Shichang Xuan[1(✉)], and Xin Li[1]

[1] Harbin Engineering University, Harbin 150001, China
xuanshichang@hrbeu.edu.cn
[2] Heilongjiang Branch of CNCERT/CC, Harbin 150023, China

Abstract. The emergence of federated learning has to some extent solved the current problems of privacy protection of terminal data and the processing technology of massive data. However, its centralized architecture still has problems such as limited access and high establishment cost, so the trend of decentralization is inevitable. Although decentralized federated learning architecture circumvents the drawbacks of centralized structure, it also loses the convenience of third-party supervision. Therefore, to address the problem of missing supervision mechanisms for worker node training behavior in decentralized federated learning architecture, this paper proposes a backdoor-based supervision mechanism for arithmetic node training behavior. The mechanism can be applied to general classification tasks. Experiments revealed that this mechanism can accurately assess the training behavior of worker nodes while maintaining the accuracy of the original task. In addition, this paper proposes a rotation scheme for the watermarked datasets involved and gives a corresponding replacement prediction method, which further ensures that the training behavior of arithmetic nodes can be quantified completely by predicting and replacing the watermarked datasets, aiding the arithmetic party training operation of the behavior monitoring mechanism.

Keywords: Federated Learning · Digital Watermark · Training Behavior Supervision

1 Introduction

In recent years, with the rapid development of IoT [1, 2], Internet of Vehicles [3, 4], edge computing [5, 6], and unmanned aerial vehicles [6, 7], more and more intelligent terminal devices are connected to the Internet, which generates a huge amount of terminal data. This is certainly a valuable asset in the field of artificial intelligence [8]. However, while massive data provides a solid foundation for the development of artificial intelligence technology, it also makes the privacy protection and processing technology of smart terminals for massive data face more serious threats and challenges [9, 10]. Although federated learning provides a feasible solution to these problems through its unique advantage of "data does not move, model moves" [11–13]. However, most of the current federated learning applications are established by business or equipment

Z. Xiao et al. (Eds.): ICECI 2022, LNICST 478, pp. 236–261, 2023.
https://doi.org/10.1007/978-3-031-28990-3_17

owners, which have problems of limited access scale, difficulty in data expansion, and high system construction cost, resulting in a large number of data demanders and data owners not being able to effectively interface with each other and limiting the value of data. Therefore, a decentralized federated learning system structure has emerged. In this structure system, data demanders and data owners form a 1-to-N or N-to-N relationship, so that the docking between demanders and owners is no longer restricted by the platform.

As the executor of the training behavior in the decentralized federated learning architecture, the stable operation of the architecture must keep its behavior honest, and any bad behavior in the model training will limit the development of the architecture. Dishonest worker nodes may falsify training data and thus claim rewards without performing the actual training behavior, affecting the fairness of the system. Therefore, it is necessary to monitor the training behavior of worker nodes. Dishonest training behavior not only affects the stability of the entire architecture but also has a more direct impact on the training results of the model, i.e., the task publisher needs to pay more to get the expected results. Therefore, introducing data and training behavior trustworthy detection mechanism before model aggregation helps to screen out malicious nodes before model aggregation, prevent performance degradation or even failure of aggregated models, and improve model accuracy.

To address the above problems, this paper proposes a digital watermarking-based training behavior supervision mechanism for worker nodes to falsify training results and affect the overall accuracy of model training to obtain unreal gains in the decentralized federated learning architecture, which can quantify and visualize the training behavior of each worker node by expanding the application scenarios of digital watermarking, to reach the goal of promoting worker nodes to remain honest in their training behavior and provide credible arithmetic support for the decentralized federated learning system.

The main contributions of this paper are as follows.

- A digital watermarking-based training behavior supervision mechanism is proposed for the screening of trusted training nodes for the worker node training behavior supervision scenario in the decentralized federated learning scenario.
- Quantifying the training behavior of worker nodes and reducing the computational and communication redundancy in existing mechanisms by migrating the digital watermarking technique to a decentralized federated learning environment with the help of the digital watermarking construction process.

The remainder of this paper is organized as follows: the second part introduces the work related to training behavior supervision as well as digital watermarking; the third part describes the architecture of decentralized federated learning; the fourth part introduces the digital watermarking-based training behavior supervision mechanism; the fifth and sixth parts provide theoretical analysis and experimental validation of the feasibility of the mechanism, respectively; and finally, the conclusion.

2 Related Work

There is very little work on the authenticity of worker node training behavior alone as a supervised object in current research for federated learning, but it is possible to borrow from some other fields. The solution given by Nishant et al. [14] is to make multiple replications of a subset of the data and then compare the models on the replicated subset for consistency, where, to ensure the non-repudiation of the behavior of COs blockchain is introduced as a way to record the behavior of COs model release. Another solution for the supervision of the training behavior of worker nodes is the evaluation of their model quality. A method for evaluating model quality is proposed by Lu et al. [15]. It uses prediction accuracy to quantify the quality of the trained local model and completes the assessment of the model quality by measuring the accuracy through the mean absolute error (MAE) metric. The lower the MAE value of the model, the higher its accuracy. In addition to MAE, Kang et al. [16], used an ONLAD model to calculate the prediction error by establishing a representative subset of normal data as a validation standard set for each parameter model received.

The current supervision of the authenticity of the training behavior of worker nodes either has a large amount of redundant computation and communication or is assessed by the quality of the local model, but the quality of the local model is not only determined by the training behavior, so this approach is not comprehensive. Even if the model upload behavior of worker nodes is recorded through the blockchain, the parameters uploaded by worker nodes still have the possibility of forgery. Therefore, a more in-depth study on the mechanism of supervising the authenticity of the training behavior of training nodes is needed from a fresh perspective.

Digital watermarking as additional information embedded in the model [17–19], the completeness of the watermark embedding is gradually increased with the training process. Therefore, the technique is naturally suitable for evaluating the training behavior of workers. Since there is no prior work on this topic, the current state of research when digital watermarking works for the original model protection purpose is given below.

A white-box solution to the DNN model intellectual property attribution problem was first proposed by Uchida et al. [20]. White-box watermarking requires prior knowledge of the model at the time of watermark embedding and embedding the watermark information into the static content of the model through the knowledge of the model information so that the tokens are also extracted from the verification model. Rouhani et al. [21] provide a way to embed watermark information into the dynamic content of a DNN model. Unlike white-box watermarking, black-box watermarking requires no prior knowledge of any model when embedding the watermark. Adi et al. [22] were the first to apply backdoor techniques to the work on watermark embedding in black-box scenarios. In this work, backdoor embedding occurs during the training of the model or fine-tuning, and is done by adding backdoor samples to the training set, which need to have a different distribution than the training samples. In addition to the above methods, other studies include adding the author's signature to the watermark [23], constructing watermarking models [24], expanding the representation of watermarking samples [25], introducing one-way hash functions to form a one-way chain pattern of trigger samples [26], and blind (blind) watermarking [27].

In terms of the protection of federated learning models, Buse et al. [28] proposed WAFFLE, a watermark embedding method for federated learning. by introducing a retraining process after model aggregation, the watermark can be embedded without accessing the training data and without affecting the accuracy of the normal task. Li et al. [29] proposed a watermarking protocol for protecting models in a fuzzy logic environment that satisfies the need to verify model ownership in FL scenarios by combining state-of-the-art watermarking schemes and cryptographic primitives.

In summary, the application of digital watermarking technology in federated learning is still in the early stage of development [30–32], and the existing schemes are not perfect either in theory or in the process of practical application, but digital watermarking can play a role in model protection, and its application in the scenario of training behavior authenticity supervision is feasible and has great research space.

3 Decentralized Federated Learning Architecture

The architecture of decentralized federated learning is shown in Fig. 1. The architecture contains two behavioral entities, the task publisher and the executor of training, i.e., the worker node. The main actions of the task publisher in different phases of the whole decentralized federated learning system are: in the initialization phase, model initialization, validation dataset (containing watermarked dataset and standard data validation set) preparation, and initial worker selection; in the training phase, model training using the validation set in parallel with the worker nodes; in the aggregation phase, data quality evaluation and training behavior evaluation, model aggregation and worker re-selection. The worker nodes, on the other hand, perform two main actions, i.e., data preparation and local training. The local data quality of a worker node and its training behavior jointly determine the quality of the local model trained by the worker node. On the one hand, a worker node may have several local datasets, which can correspond to different training tasks, but the quality of the datasets may be uneven, and having one high-quality dataset does not mean that all the datasets of the worker node reach the high-quality standard; in addition, different models have different functions and require different data characteristics, so the quality of the data is not constant but varies dynamically depending on the model requirements. On the other hand, even if a worker node has data that fits well with the training task of a certain model, but the worker node does not perform the actual training behavior or falsifies the training data due to factors such as malicious purposes or lazy behavior, then the contribution of the worker node to the task is insufficient or even negative. The above-mentioned problems of low data quality and falsified training behaviors will eventually affect the quality of the local model trained by the worker node and the stability and trustworthiness of the whole architecture.

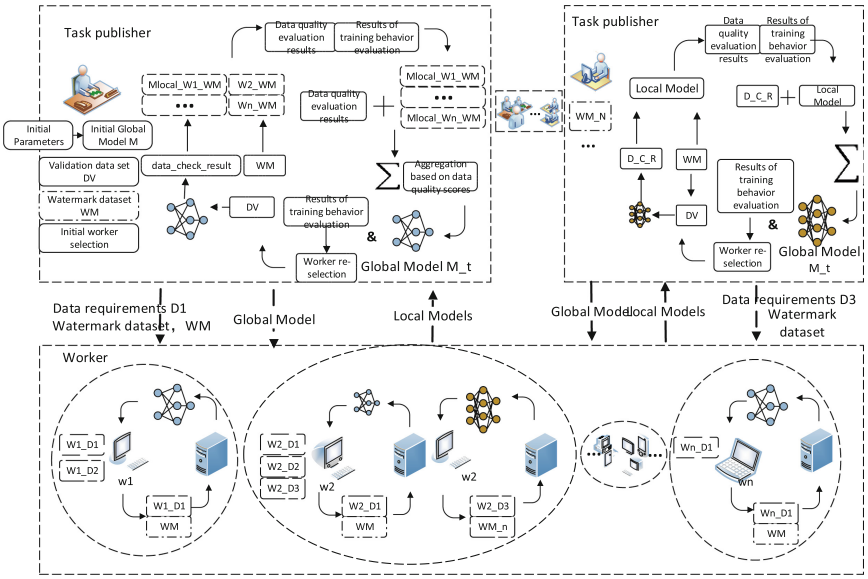

Fig. 1. Architecture of the decentralized federated learning system

The description of the entities in this architecture and the interaction process between the entities are as follows.

1. System initialization process: this process mainly consists of the initial preparation of both parties for the respective upcoming federated learning process.

   a. Task publisher. The task publisher, as the party that proposes the training requirements, first needs to initialize the required model to obtain the initial model $M^{\theta}_{G(0)}$, and construct the corresponding standard validation dataset D_v, watermark dataset D_{wm}, where the standard validation dataset D_v is mainly used to assist in evaluating the data quality of worker nodes, and the watermark dataset D_{wm} is mainly used for worker authenticity evaluation of node training behavior; after that, the task publisher will select the appropriate set of worker nodes S for training. And sends the initial model $M^{\theta}_{G(0)}$, data requirements, and the watermark dataset D_{wm} together to all selected worker nodes.
   b. Worker nodes: After receiving the requirements from the task publisher, the selected worker nodes first need to prepare their own local datasets D_k, $\mathrm{k} \in \mathrm{S}$ according to the data requirements of the task publisher, and after processing their own local datasets, they will be trained locally according to the training requirements of the task publisher.

2. Training phase: the training phase is jointly participated by both parties.

a. Task publisher. The initial model will be trained locally using the standard validation dataset D_v and the obtained results will be used for the evaluation of data quality.
b. Worker Node. Start the local training process and send the resulting local model $M^{\theta}_{W_k(t)}$ back to the task publisher after the training is completed, waiting for it to aggregate the local models of all worker nodes and to downlink the new global model. If no new global model is sent down, the training task will be ended.

3. Aggregation phase. Done by the task publisher. In this phase, the task publisher needs to inspect the local model sent by the worker node to complete data quality evaluation and authenticity detection of the worker's training behavior, and then finish the aggregation process of the local model according to the corresponding aggregation algorithm to get the new global model $M^{\theta}_{G(t+1)}$, and finally check its performance, and send it to the worker node if it does not meet the requirements The training continues and the worker nodes continue the training phase; if the model meets the requirements then the current training task is ended.

In this architecture, the data quality evaluation of the task publisher and the training behavior check correspond to the quality detection of the local dataset and the supervision of the training behavior of the worker node, respectively. These two supervision mechanisms are important to ensure that worker nodes upload high quality local models. The results of data quality evaluation will be applied on top of the federated aggregation algorithm.

4 Supervisory Mechanism for Training Behavior of Worker Nodes Based on Digital Watermarking

4.1 Overall Architecture

In decentralized federated learning scenarios, worker nodes may have malicious or lazy behaviors, thus not performing training or faking the training process. Therefore, a digital watermarking-based mechanism for monitoring the authenticity of training behavior (WM-TBM) is proposed to ensure that worker nodes perform the training behavior honestly. Firstly, the overall framework of the mechanism is introduced, and the role roles of each participant and the overall interaction process are introduced, followed by a detailed description of the mechanism, specifically, the watermarking dataset construction process, the digital watermark embedding process, the worker node training behavior checking process, and the watermarking dataset replacement process.

Federated learning is an iterative process, so WM-TBM will run through the whole interaction process between task publisher and worker nodes. WM-TBM consists of three parts, where the execution of the task, i.e., the training process of the federated learning task, is done by the worker nodes and the rest is done by the task publisher. Its overall framework is shown in Fig. 2.

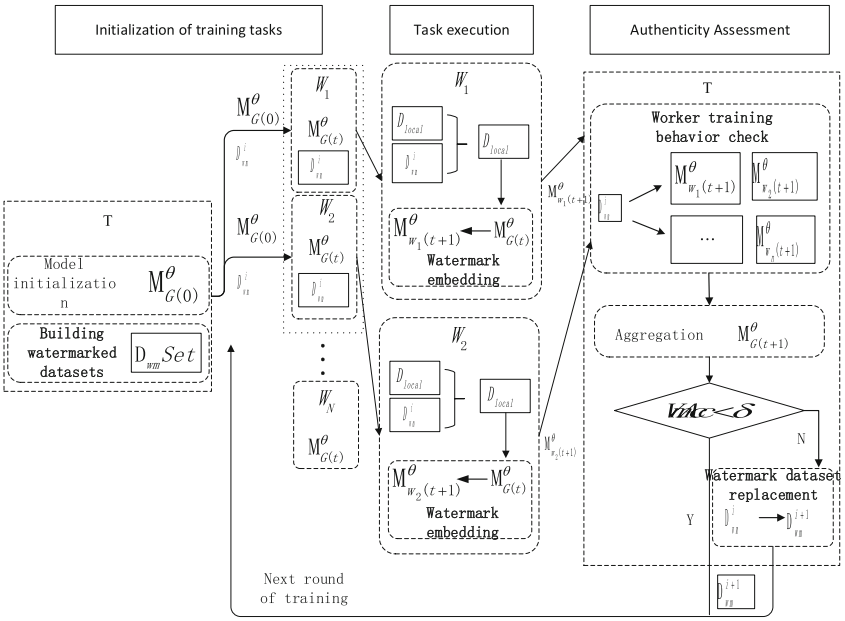

Fig. 2. Overall framework of the WM-TBM mechanism

Initialization of the Training Task. In the initialization phase, the task publisher needs to initialize the local model to get the global model $M_{G(0)}$ and prepare n watermarked datasets $D_{wm}Set\{D^1_{wm}, D^2_{wm}, \cdots D^n_{wm}\}$ for detecting the training behavior of worker nodes. The initial global model $M_{G(0)}$ and one of the watermarked datasets D^i_{wm} are sent to the selected worker nodes after the above preparatory work is prepared.

Task Execution. In this phase, the worker node uses the global model issued by the task publisher instead of the local model and receives the watermark dataset D^i_{wm}. After merging the local dataset D_{local} with D^i_{wm} into the new local dataset D_{local}, the worker node starts the local training process and gives the training results back to the task publisher in a timely manner.

Specifically, at a specific round t, the task publisher will send the current global model $M_{G(t)}$ to N workers, and the selected worker s $\in$ S will perform the training task $t_i(w)$ locally using its own local dataset D_{local} and learning rate l_r, and after the local E rounds of training, get a new local model $M^{wm}_{s(t)}$, and returns that model to the task publisher.

This training process can be summarized as finite-times aggregation optimization by.

$$min_{w \in R^d}\left[T(w) := \frac{1}{N}\sum\nolimits_{i=1}^{N} t_i(w)\right] \tag{1}$$

where N worker nodes use the local privacy dataset $D_{local} = \{x_j^i, y_j^i\}$ for the local training task $t_i(w)$, and $\{x_j^i, y_j^i\}$ represents the data samples and the labels of the samples; the training task performed locally by the workers can be described as $t_i(\mathrm{w}_i) = l(\{x_j^i, y_j^i\}_{j \in D_i}, w_i)$, l being the loss resulting from the prediction using the local weights. The worker nodes will complete the embedding process of the watermark while performing the local task.

Authenticity Assessment. This phase will involve three main assessments, namely, worker training behavior authenticity assessment, model accuracy assessment on the primary task, and watermarking task assessment.

The worker training behavior authenticity assessment will use the digital watermark dataset D_{wm}^i issued by the task publisher to check all local models and test the accuracy of the local model on D_{wm}^i, specifically two dimensions will be checked, and the results of this phase will be used as the main basis for the worker training behavior evaluation, which will be used by the task publisher to make a decision on whether to continue to select the worker for subsequent training tasks.

The accuracy evaluation of the watermarking task will be performed using the watermarked dataset on the aggregated new global model, which is mainly used to determine whether watermarked model replacement is required. Performing watermarked dataset replacement when necessary ensures that the training behavior of worker nodes is quantified completely and consistently.

The accuracy evaluation of the model on the primary task occurs after the training behavior of the workers is evaluated. The task publisher aggregates the training results returned by the workers according to Eq. (4–2) to obtain a new global model $\mathrm{M}_{G(t+1)}$.

$$\mathrm{M}_{G(t+1)} = \mathrm{M}_{G(t)} + \frac{\eta}{n}\sum\nolimits_{s=1}^{s}\left(M_{s(t)}^{wm} - \mathrm{M}_{G(t)}\right) \tag{2}$$

After completing the aggregation process of the global model, the task publisher needs to validate the performance of the new global model using the validation set of the main task, determine whether it meets the requirements for use, and make a decision on whether to continue with the next round of the training process of federated learning.

The specific process is shown in Algorithm 4.1:

Algorithm 4.1: WM-TBM

Input: D_{local}: Training set; D_{test}: Test set; $M_{G(0)}$: Initial Model;
S: Selected set of workers; T: Global Training Round;
$D_{wm}Set\{D_{wm}^1, D_{wm}^2, \cdots D_{wm}^n\}$: Collection of watermarked data sets;
η: Global model learning rate; η_s: Local learning rate of worker s;
E: Local training rounds

Onput: $M_{G(T)}$ Global Model

// **Task publisher execution:**

TaskInit():

$M_{G(0)} \leftarrow M_{G(0)} - \eta\nabla l(M_{G(0)})$

$W_{G(0)} \leftarrow (parameters\ of\ M_{G(0)})$

TaskEvaluation():

for each epoch t from 1 to T **do**

WorkerEval()

$W_{G(t+1)} \leftarrow \sum_{s=1}^{S} \frac{1}{m} W_{s(t)}$

$M_{G(t+1)} \leftarrow W_{G(t+1)}$

$Acc_{test}^{global} \leftarrow M_{G(t+1)} + D_{test}$

end for

return $W_{G(t+1)}$

WorkerEval():

for each s ∈ S **do**

$VmAcc_{test}^{s} \leftarrow M_{s(t)}^{wm} + D_{wm}$

if $VmAcc_{test}^{s}$ *is not available*

exchange D_{wm}^{i} *to* D_{wm}^{i+1}

send D_{wm}^{i+1} to each s ∈ S

end if

end for

// **Worker execution:**

TaskExec(**W**):

receive $W_{s(t)}$ *and* , D_{wm}^{i} *from task publisher*

$M_{s(t)} \leftarrow (replace\ the\ parameters\ of\ M_{G(t-1)}\ with\ W_{s(t)})$

$\boldsymbol{D_{local} \leftarrow dataset\_patch(D_{local}, D_{wm})}$

for each epoch t from 1 to E **do**

for batch b ∈ D_{local} ***do***

$M_{s(t+1)}^{wm} \leftarrow M_{s(t)}^{wm} - \eta\nabla l(M_{s(t)}^{wm}, b)$

$W_{s(t+1)} \leftarrow (parameters\ of\ M_{s(t+1)}^{wm})$

end for

end for

end for

return $W_{s(t+1)}$

4.2 Watermarking Dataset Construction

One of the necessary requirements for watermarking data is that it should not interfere with the execution of the main task in federated learning, so the watermarked data should be independent of the training data, and the watermarked data chosen to be independent of the training data outperforms the pre-specified Gaussian noise patterns in terms of performance [14]. Therefore, a new idea is provided for the construction of the watermarked dataset based on the previous studies, which is to embed a fixed pattern in the images that are independent of the training set. The pattern will act as a trigger for the watermarked backdoor and is used to detect the completion of this backdoor, which identifies whether the worker has performed the training process honestly or not.

Specifically, the images that are not related to the main task are first selected as watermarks, and then corresponding patterns are embedded for each image, which are different in terms of color, category, position and orientation, and are given a random label from the actual task category. Thus, for the specified pattern, the model with an accompanying watermark backdoor will output its assigned label.

The advantages of using this approach to construct the watermarked dataset are that (1) using a watermarked dataset that is independent of the main task and adding different patterns to it ensures that the watermarked set is independent of the other training sets, greatly improving the generalizability of the watermarked dataset so that it can act on multiple federated learning tasks; (2) the randomly generated embedding patterns for each category of the main task further ensures that each sample noise is unique; (3) since each image can correspond to a different pattern, the same set of watermarked data can be reused, reducing the difficulty of data preparation. Figure 3 gives the style of a partial watermarking dataset, and the text in the figure indicates the assigned label to which the image is assigned, and the source of the label is the Cifar10 dataset.

Fig. 3. Partial watermarking dataset

4.3 Watermark Embedding

After generating the watermark data, the next step is to embed the watermark into the target DNN, which can be done with the help of deep neural network due to its strong intrinsic learning ability. The embedding algorithm SF-WE is shown as follows.

Algorithm 4.2: SF-WE

Input: $D^i_{non-wm} = \{X_i, Y_i\}_{c=1}^{C}$: Prepared watermark dataset;
$\sigma = \{Y_o, Y_n\}(o \neq n)$: Mapping of the original label Y_o to the new label Y_n
Output: $M^{wm}_{s(t)}$ Partial model with watermark; D^i_{wm}: Watermark dataset
// **Task publisher execution:**
watermarkingEmbedding():
  $D^i_{wm} \leftarrow \emptyset$
  $D^i_{temp} \leftarrow sample(D^i_{non-wm}, Y_o, percentage)$
  for each $d \in D^i_{temp}$ **do**
    $x_{wm} = \text{embedding\_pattern}(d[x], \text{pattern}),\ y_{wm} = y_n$
    $D^i_{wm} = D^i_{wm} \cup \{x_{wm}, y_{wm}\}$
  end for
// **Worker node execution:**
$M^{wm}_{s(t)} = \textbf{TaskExec}(M_{s(t)}, D^i_{wm})$
return $M^{wm}_{s(t)}$

The algorithm SF-WE takes as input the original watermark dataset D^i_{non-wm} and the label mapping relation $\sigma = \{Y_o, Y_n\}(o \neq n)$ and outputs the watermark dataset D^i_{wm}, which in turn outputs the local model $M^{wm}_{s(t)}$ with the watermark after the worker nodes are trained. The label mapping relationship will be defined by the task publisher indicating how the watermark will be labeled. Y_o is the true label of the original data and Y_n is the pre-defined watermark label that will include the fingerprint used for training behavior verification. Next, the algorithm's watermarkingEmbedding() function will draw all the labels labeled with Y_o from the trained dataset, on which the corresponding patterns are generated and re-labeled with Y_n, which will generate both the patterns and the carefully prepared labels. After receiving the complete watermarked dataset D^i_{wm}, the worker node will use this D^i_{wm} dataset and the local dataset D_{local} for local training, during which the DNN will automatically learn the patterns of these watermarked data, so that the watermark specified by the task publisher is embedded in the local model of this worker node. The completion of the digital watermark will gradually increase with the number of local training rounds of the worker node. The completion of the digital watermark will be used as one of the main metrics for subsequent checks of the training behavior of the worker nodes.

4.4 Worker Training Behavior Check

When the task publisher receives the local model M_(s(t))^wm from the worker node, the complete degree of the watermark will be checked, and this result will further identify whether the worker node has performed the training process truthfully. To ensure the objectivity of the evaluation, the training honesty of the worker node will be checked from two perspectives separately, and the final score of the authenticity of the training behavior of this worker will be obtained. The specific checking process is shown in Algorithm 4.3.

Algorithm 4.3 Checking Process

Input: $D_{wm}Set\{\mathrm{D}^1_{wm}, \mathrm{D}^2_{wm}, \cdots \mathrm{D}^n_{wm}\}$: Watermark dataset collection;
$M^{wm}_{s(t)}(\mathrm{s} \in \mathrm{S})$: Local model of worker nodes
Onput: $\{\xi^t_{s_1}, \xi^t_{s_2} \cdots \xi^t_{s_m}\}$: Training authenticity score set for workers selected in that round
// **Task publisher execution:**
WorkerEval():
  $\mathrm{D}^i_{wm} \leftarrow D_{wm}Set\{\mathrm{D}^1_{wm}, \mathrm{D}^2_{wm}, \cdots \mathrm{D}^n_{wm}\}$
  for each $M^{wm}_{s(t)}$ **do**
    $acc^t_{s_i} \leftarrow \mathrm{Acc}(M^{wm}_{s(t)}, \mathrm{D}^i_{wm})$
  end for
  $acc^t_{avg} \leftarrow \frac{1}{m}\sum_{i=1}^{m} acc^t_{s_i}$
  if $acc^t_{s_i} > \mathrm{Acc}(M^{wm}_{s(t-1)}, \mathrm{D}^i_{wm})$ and $acc^t_{s_i} \geq acc^t_{avg}$
    $\xi^t_{s_i} \leftarrow acc^t_{s_i}$
  end if
  return $\{\xi^t_{s_1}, \xi^t_{s_2} \cdots \xi^t_{s_m}\}$

After receiving the local models $M^{wm}_{s(t)}$ for all worker nodes, the task publisher first needs to determine the watermark dataset D_{wm}^i used for the current training in the watermark dataset set $D_{wm}Set\{\mathrm{D}^1_{wm}, \mathrm{D}^2_{wm}, \cdots \mathrm{D}^n_{wm}\}$, and after that use this dataset D^i_{wm} to detect all local models and the The accuracy of all worker nodes is averaged as one of the comparison metrics. After the above steps, the following checks are performed on the local models of each worker node.

- Compare horizontally the accuracy of all selected workers and the average accuracy to ensure that they did not undergo falsification training.
- Check the test accuracy of the local model submitted by the workers for D^i_{wm} to ensure that it is higher than the accuracy of the global model in the previous round, and ensure that the complicity can be detected in the case that the vast majority of workers perform falsification training that makes the average accuracy rate decrease and leads to the failure of this evaluation metric.

After the checks have been performed, the accuracy $acc^t_{s_i}$ of the local model of the worker node for the current round on the watermarked dataset is recorded as the training

score $\xi_{s_i}^t$ of that node, which is used as the basis for the selection of worker nodes performing subsequent training tasks.

4.5 Watermark Dataset Replacement

Throughout the training process of federated learning, the task publisher and worker nodes will communicate several times, and to ensure the observability of the watermark completion, the watermark dataset needs to be rotated when it reaches the unavailable state. Specifically, the rotation of the watermark dataset to enable another watermark task and the natural extinction of the current watermark task ensures that the watermark task can continuously and completely quantify the training behavior of the worker nodes. The watermark dataset replacement process is shown in Algorithm 4.4.

Algorithm 4.4 Watermarked Dataset Replacement

Input: $D_{wm}Set\{\mathrm{D}_{wm}^1, \mathrm{D}_{wm}^2, \cdots \mathrm{D}_{wm}^n\}$: Watermark dataset collection;
$M_{s(t)}^{wm}(\mathrm{s} \in \mathrm{S})$: Local model of worker nodes
// **Task publisher execution:**
WMDataSetExchange():
  for each t ∈ T **do**
    $M_{G(t+1)}^{wm} = \mathrm{M}_{G(t)} + \frac{\eta}{n}\sum_{s=1}^{S}(M_{s(t)}^{wm} - \mathrm{M}_{G(t)})$
    $VmAcc_{s(t)}^{test} \leftarrow M_{G(t+1)}^{wm} + D_{wm}^i$
    if $(VmAcc_{s(t)}^{test} - VmAcc_{s(t-1)}^{test} > \Delta\delta)\ and (Cnt_{useless} > \vartheta)$
      exchange D_{wm}^i *to* D_{wm}^{i+1}
      send D_{wm}^{i+1} to each s ∈ S
    end if
  end for

Define the observable coefficient as δ. When the difference between the accuracy $VmAcc_{s(t)}^{test}$ of the global model $M_{G(t+1)}^{wm}$ on the watermarked dataset D_{wm}^i and the accuracy $VmAcc_{s(t-1)}^{test}$ of the previous round t is less than the observable coefficient, it means that the watermarked dataset has reached the unavailable state ΔF. To make this criterion have some fault tolerance, the tolerance factor $Cnt_{useless}$ is defined as the number of times that the unavailable state can be tolerated, and the watermarked dataset is considered to be always in the available state until this number is satisfied. When the performance of the watermarked dataset on the global model satisfies the above conditions at the same time, the replacement of the watermarked dataset $\{\mathrm{D}_{wm}^i \rightarrow \mathrm{D}_{wm}^{i+1}\}$ is performed, the reinforcement of the watermarking task T_x corresponding to the watermarked dataset D_{wm}^i is canceled, and the watermarking task T_y corresponding to the watermarked dataset D_{wm}^{i+1} is enabled.

As the training rounds proceed, the performance of task T_x on the main task T will gradually decline and the watermarking task will gradually die out, and after the performance of T_x on the main task T declines to a certain degree, T_x will return from the unavailable state to the available state and can participate in the next watermarking

rotation, i.e., the above watermarking rotation process can be expressed as follows.

$$\begin{aligned} &\forall \mathrm{i} < \mathrm{E}, \text{ if } S_i^{T_x} \xrightarrow{e_f} S_{false}, \text{ exchange } T_x \text{ to } T_y \\ &\text{where } S_{false} \text{ is } \left(Acc_{e_f} - Acc_{e_f-1}\right) < \Delta\delta \end{aligned} \tag{3}$$

where e_f is the number of rounds elapsed when the task reaches the unavailable state, and $S_i^{T_x}$ denotes the state exhibited by task T_x at the ith round.

Since the speed of watermark extinction is smaller than the speed of watermark creation, two watermark tasks T_1 and T_2 corresponding to two watermark datasets of the same size are taken as an example, assuming that task T_1 rotates with task T_2 after reaching the unavailable state, and task T_2 needs to rotate when it reaches the unavailable state, task T_1, with which it should rotate, does not completely extinguish, thus will continue to watermark tasks from the partially extinguished state, and this time the number of turns required to reach the unavailable state will be lower than the number of turns required to reach the unavailable state in the previous time, that is, after k times of the above repeated exchanges, the initial state of both tasks will become unavailable and the subsequent tasks cannot be completed. The number of rounds required for both tasks to reach the unavailable state is

$$e_f^{T_1} + e_f^{T_2} + \sum_{i=1}^{k} (e_{dis_i}^{T_1} + e_{dis_i}^{T_2}) \tag{4}$$

where $e_{dis_i}^{T_1}$ denotes the number of rounds elapsed when task T_1 reaches the unavailable state again at the i-th extinction. Therefore, with the same size of watermarking dataset, the number of datasets required to complete the entire federated learning training can be determined by determining the creation rate and extinction rate of watermarking tasks.

To ensure observability of watermark completion, i.e., to be able to completely quantify the training behavior of worker nodes, the watermark data needs to have different patterns. In theory, when the given watermarking dataset is large enough, the state of that watermarking task may not reach the unavailable state during the whole execution of the training task. This reduces the complexity of watermark rotation to a certain extent, but also lengthens the time required for the entire training task, and may even appear that the time spent on the watermarking task far exceeds the time spent on the main task, which is undoubtedly more than worth the loss. Instead, consider another scenario: prepare a number of small-scale watermarking data for constant rotation. This approach reduces the time taken by the watermarking task, but it injects a large number of backdoors into the model, which has a significant impact on the security of the model. In addition, there is a potential pitfall of small-scale datasets, namely, the accuracy of backdoor watermarking can no longer accurately quantify the training behavior of workers before all rounds of training tasks are completed, and frequent replacement of watermarked datasets will increase the communication overhead of worker nodes in terms of data acquisition.

Communication bottleneck has been one of the important problems facing federated learning. During the whole training process of federated learning, the communication between task publisher and worker nodes is focused on exchanging model data and

replacing watermark datasets. Since the volume of model data is significantly smaller than that of watermarked datasets, the main communication overhead mainly comes from the transfer of watermarked datasets. Although the addition of watermarked datasets increases the communication overhead of federated learning, this part of consumption can be minimized by a reasonable dataset size design.

5 Feasibility Analysis of Training Behavior Monitoring Mechanism

The feasibility analysis of the training behavior supervision mechanism includes the analysis of the inherent characteristics of digital watermarking, the analysis of the data independence of worker nodes, and the analysis of the additional computational cost introduced. The training behavior supervision in federated learning uses digital watermarking technology, and although it is different from the traditional model protection application scenario, the inherent characteristics of digital watermarking technology should be retained intact, i.e., the original functionality of the model and the security of the watermark should not be destroyed; secondly, the inclusion of the watermarked dataset should not have an impact on the independence of the local data of the worker nodes; finally, for the additional watermarking verification process introduced, a comparison with DDSM for a comparative analysis on computational and communication redundancy.

- The original functionality of the model is not destroyed

That is, the watermarking task cannot have an impact on the main task, both tasks need to be performed independently and both perform well on the relevant targeted tests. Therefore, the model obtained by noting the participation in the watermarking task is M′, and M′ should satisfy the following conditions.

$$\begin{aligned} &\underset{x \in D/T}{Pr}[f(d) \neq Classify(M', d)] \leq \varepsilon \\ &and \underset{x \in T}{Pr}[T_L(d) \neq Classify(M', d)] \leq \varepsilon \end{aligned} \tag{5}$$

The total error rate of M′ is at most $\varepsilon' = \varepsilon + n/|D|$, since D is much larger than n, ε' will be infinitely close to ε. In summary, the watermarking task can be performed simultaneously with the main task while ensuring that the main task is not disturbed.

- Security

Security refers to the fact that the watermark itself is not easy to forge. The guarantee of watermark security mainly comes from two aspects: first, in the construction of the watermark dataset, the choice of using the insertion of specific patterns on irrelevant data, the size and color of the patterns are specified by the task publisher, which creates difficulties for the forgery of the dataset; second, forging the watermark dataset does not bring expected or additional benefits to the worker nodes, and the bad behavior they show

in the training behavior will probably lead to their disqualification from participating in the next round of training.

For the watermarked dataset $\mathrm{T} = \left\{t^{(1)}, \ldots, t^{(1)}\right\}$ and the watermarked dataset label $T_L = \{T_L^{(1)}, \ldots, T_L^{(n)}\}$, the task publisher has absolute control as well as the right to know. Even if a worker forges the backdoor watermark dataset T_{false} and submits the model $M_{local}^{T_{false}}$, it will still be checked out at the WorkerEval() step as follows.

$$VmAcc_{test}^{c} \leftarrow M_{local}^{T_{false}} + T < VmAcc_{test}^{c} \leftarrow \mathrm{M}_{local}^{c} + T \tag{6}$$

- Data independence of worker nodes

The task publisher does not need to know any prior knowledge of the training data during the whole federated learning process, and the task publisher does not need the training data of worker nodes to generate its own watermark sequence, and the data of worker nodes will be saved locally independently to meet the requirements of federated learning.

- Computational cost analysis

The computational cost includes computational resources and communication volume. In the work of Somy et al. [33] (DDSM), DOs need to sign offline agreements with a certain number of COs and securely send subsets of data to selected COs, and to ensure data privacy, DOs need to partition the entire dataset such that each CO has a range of all classes and values for which the subset does not contain data, and the subset cannot contain a single class from most of the data. The solution to the problem of false training by COs is to perform replication of data subsets among multiple COs, as shown in Fig. 4. During federated learning training, model training is performed by all m*n COs holding subsets of the dataset, so that when a dishonest CO reports false information about training behavior, it can be compared with other COs of the same replicated subset, as long as the dishonest CO node is less than one-half of the total number of COs, the dishonest behavior of that CO can be identified.

Assume that there are m copies of data involved in training in each round, i.e., m DOs in DDSM, m × n COs in training, and m worker nodes in WM-TBM in training. Let the GPU computational resources required for training a data of size S be c.

The computational resources required for training and the computational resources required for validation in the DDSM are

$$C_{Train} = m \times n \times c \tag{7}$$

$$C_{Verify} = m \times n \times (m \times n - 1) \tag{8}$$

Therefore, the total computational resource consumption of DDSM is

$$C_{Total} = C_{Train} + C_{Verify} = m \times n \times c + m \times n \times (m \times n - 1) \tag{9}$$

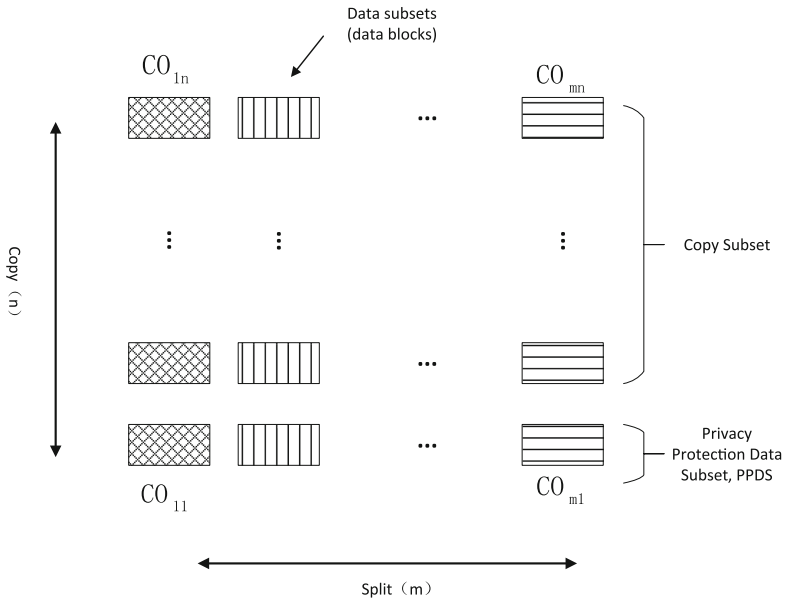

Fig. 4. DDSM Solutions

In WM-TBM, the size of the watermarked dataset received by the worker nodes is W, so the computational resources required for each round of training and the computational resources required for validation are:

$$C_{W-Train} = m \times c \tag{10}$$

$$C_{W-Verify} = m \times c \times \frac{W}{S} \tag{11}$$

Therefore, the total computational resource consumption of WM-TBM is

$$C_{W-Total} = C_{W-Train} + C_{W-Verify} = m \times c + m \times c \times \frac{W}{S} \tag{12}$$

Define the savings index C_{Save} as the ratio of the computational resources required by DDSM to those required by WM-TBM, so that when the value of C_{Save} is greater than 1, it indicates that WM-TBM performs better.

$$C_{Save} = \frac{C_{Total}}{C_{W-Total}} = \frac{S}{S+W}(n + m \times n^2 \times \frac{1}{c} - \frac{n}{c}) \tag{13}$$

The analysis of the values of C_{Save} using modeling is distinguished from centralized training in that the number of worker nodes required for federated learning is at least 2, i.e., $m \geq 2$; in addition, DDSM requires segmentation and replication of the dataset, i.e., $n \geq 2$. Consider two configurations of watermarked dataset sizes, one for the normal training case where the watermarked dataset is much smaller than the training dataset,

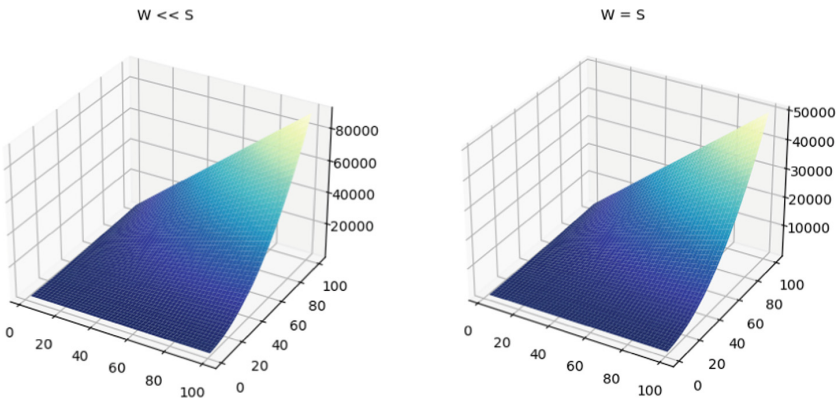

Fig. 5. Comparison results of computing resources

and the second for the extreme case where the watermarked dataset is the same size as the training set, and the results obtained are shown in Fig. 5.

The minimum values obtained for the two configuration cases are 2.834 as well as 5.152, which are in accordance with the expected logic. In summary, WM-TBM can save more computational resources than DDSM when the federated learning condition is satisfied, and the improvement of this saving index is more obvious as the scale of federated learning increases.

In terms of communication volume, assuming that the communication volume required to transmit a data set of size S is t_1 and the communication volume required to transmit model parameters is t_2, the transmission of model parameters is mainly divided into two transmission processes, namely, transmission from CO to blockchain and downloading parameters from blockchain by MO, so the communication volume required by DDSM is

$$T_{Total} = T_{dataset} + T_{model} = m \times t_1 + m \times n \times t_2 \times 2 \tag{14}$$

The amount of communication required by the WM-TBM is:

$$T_{W-Total} = T_{W-dataset} + T_{W-model} = m \times t_1 \times \frac{W}{S} + m \times t_2 \tag{15}$$

Define T_{Save} as the difference between the amount of communication required by the DDSM and the amount of communication required by the WM-TBM. Similarly, a value of T_{Save} greater than 0 indicates that the WM-TBM performs better:

$$T_{Save} = T_{Total} - T_{W-Total} = m \times t_1 \times (1 - \frac{W}{S}) + m \times t_2(2 \times n - 1) \tag{16}$$

The size of the validation dataset W is much smaller than the size of the training dataset S. Therefore, $T_{Save} > 0$ holds for both DDSM and WM-TBM configurations, so the amount of communication required by WM-TBM is always smaller than that of DDSM.

In summary, WM-TBM has a very obvious advantage over DDSM in terms of computation and communication volume, especially when the scale of federated learning is large.

6 Experimental Design and Analysis of Results

6.1 Experimental Design

In order to verify the feasibility of WM-TBM, a total of five sets of experiments are designed for watermarking task impact validation, watermarking dataset usability validation, worker node training behavior authenticity assessment validation, watermarking task periodicity validation, and watermarking dataset rotation validation from the perspective of watermarking task itself and watermarking dataset replacement.

The experiments use the Cifar10 dataset, and the watermark dataset is a small-scale dataset constructed according to the above watermark dataset construction method.

6.2 Analysis of Experimental Results

Experiment 1: Impact Validation of the Watermarking Task. The Cifar10 dataset is used for the classification task in a federated environment, the aggregation algorithm uses the federated average algorithm, and the dataset is subjected to non-IID processing. The results obtained by running 1000 rounds each without and with the watermarking task, where the size of the watermarked data for the watermarking task is 1000, the number of locally set workers is 4, and the number of local rounds for training each worker locally is 5, are shown in Fig. 6.

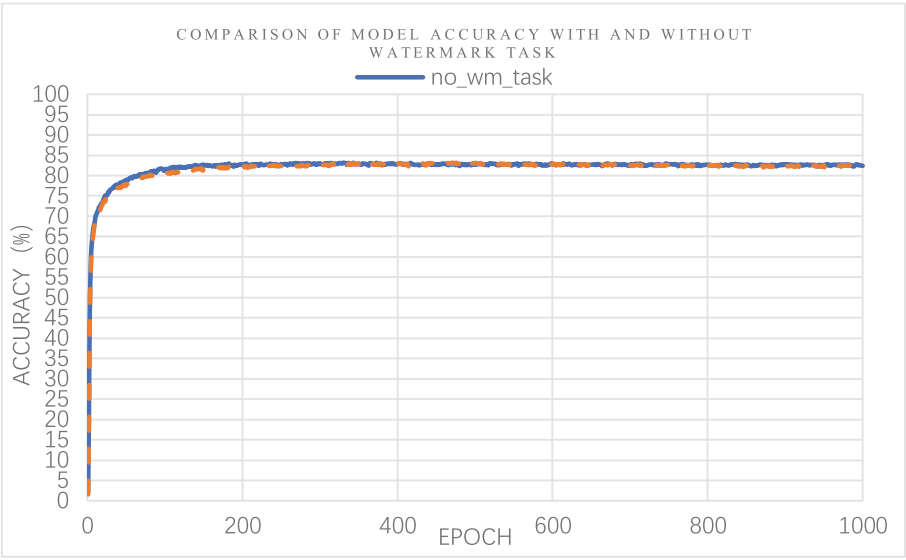

Fig. 6. Comparison of model accuracy with and without watermarking task

From the experimental results, it can be seen that the convergence trend of the accuracy of the resulting model on the main task is smooth with or without the watermarking task involved, and eventually converges to about 82%, which shows that the watermarking task does not have an impact on the main task.

Experiment 2: Watermarking Dataset Availability Validation. The size of the watermarked dataset was experimented from 100–1000 in units of 100 to obtain the number of rounds required for its creation and extinction, respectively, and the experimental results are shown in Fig. 7.

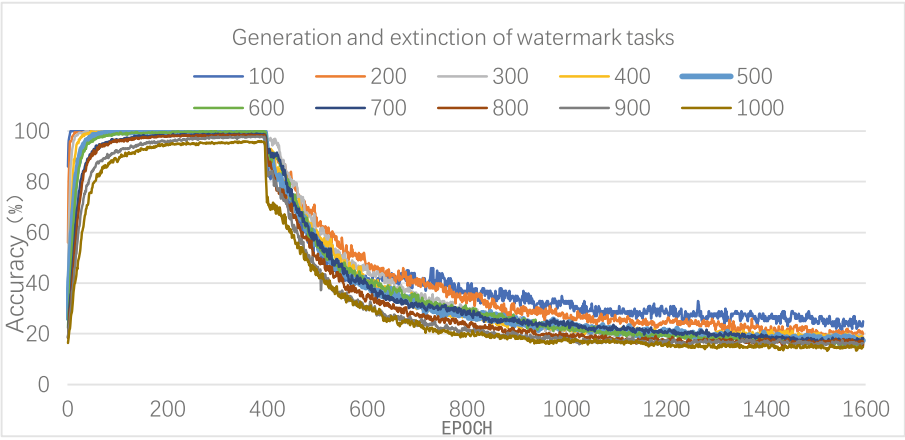

Fig. 7. The generation and extinction cycle of watermarking

As can be seen from the experimental results, the extinction of the watermark task will remain at a uniform level for a long time after maintaining it for a period of time, so it is unwise to wait for its accuracy rate to extinguish to zero before rotation, and the rotation operation can be performed when the extinction rate of the watermark task is reduced to a point where the floating change is not significant.

In order to determine the unavailability status of the watermark, the data of the first 100 rounds of the task are organized separately as follows (Fig. 8).

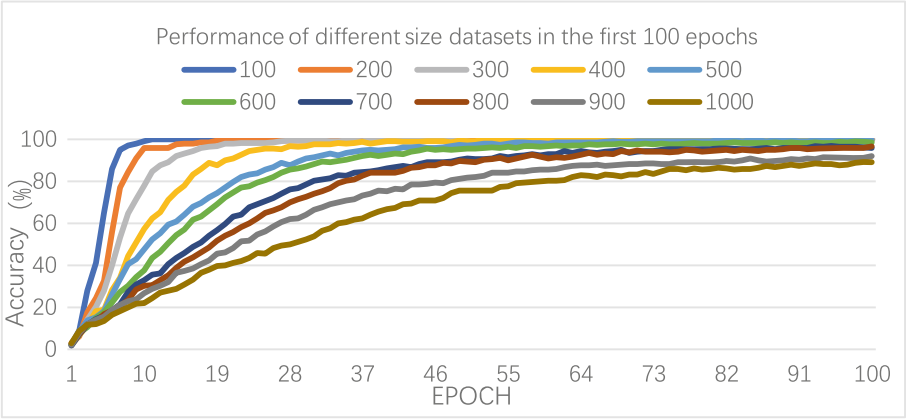

Fig. 8. Performance of different dataset sizes in the first 100 rounds

From the above experimental results, it is clear that watermarked datasets of different sizes have roughly the same trend in convergence; however, the dataset with a data size of 1000, for example, has a relatively flat convergence rate, and the difference in accuracy from round to round is not very helpful in determining the training behavior. Therefore, in subsequent experiments, a dataset of size 800 that can be maintained for as many rounds as possible while observing significant changes was chosen.

Experiment 3: Authenticity Assessment of Workers Training Behavior Validation. In the following experiments, one worker among all four workers will be randomly selected for spurious training, and the local model submitted by the worker will be examined for the watermarking task before aggregation, and it will be determined whether the worker has been trained or not. The results of the experiments are shown in Fig. 9.

From the experimental results, it is clear that the inspection before aggregation can accurately quantify the training behavior of workers. The local models submitted by workers with dishonest training behavior have a large gap in testing accuracy on the watermarking dataset with the local models submitted by worker nodes that honestly perform training, and therefore, dishonest workers can be identified.

Experiment 4: Periodic Validation of the Watermarking Task. Define the observable coefficient as 2 and the tolerable coefficient as 3. When the current watermarking data task reaches the unavailable state, the execution of the watermarking task is canceled, and the above task is continued after the extinction rate of the watermarking task is stabilized. The experimental results are shown in Fig. 10.

Fig. 9. Comparison of the training behavior of workers

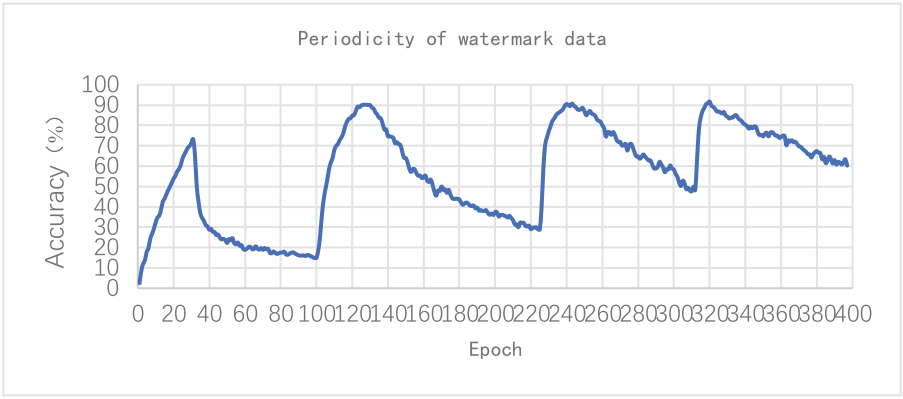

Fig. 10. Periodicity of watermarked data

From the experimental results, it can be seen that the watermarking task is cyclical in nature. And in each cycle of watermarking, its extinction rate is much smaller than the generation rate, and the watermarked dataset of size 800 has reached the unusable state after only 9 rounds at the fourth cycle.

Experiment 5: Watermarking Data Set Rotation Verification. Two watermark datasets of size 800, D_1 and D_2, are added to the training process, and when D_1 reaches the unusable state, the D_2 dataset is used for rotation, and the two datasets are worked on alternately. Before performing the dataset replacement, it is necessary to know the performance of both datasets on the model when they are not involved in the watermarking task, and experiments are conducted on both datasets to obtain the following results (Table 1).

Table 1. Performance of the watermarked dataset on the model when the watermarking task is not joined

Dataset	Major Task Test Set	Watermark dataset
D_1	82.60	9.875
D_2	82.64	9.625

With both watermarked datasets participating in the rotation, the experimental results are shown in Fig. 11.

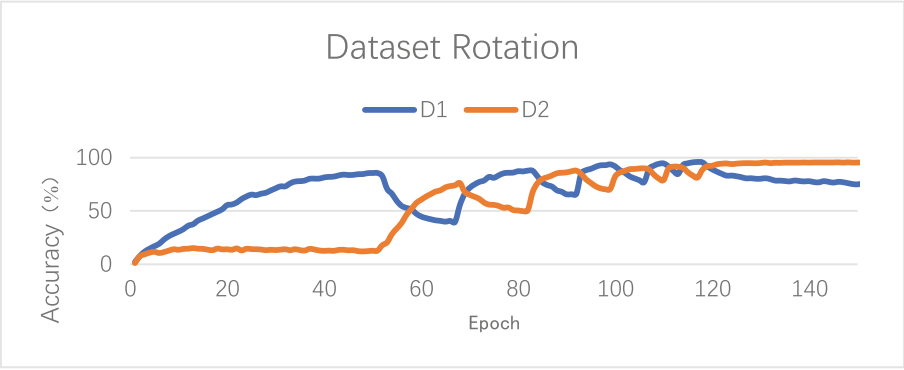

Fig. 11. Rotation of the dataset

From the experimental results, it can be seen that the rotation of watermarking datasets is feasible. Although the availability of the two watermarking datasets is in a gradually decreasing trend during the crossover for training, the watermarking task can be made complete by configuring a suitable number of replacement datasets to cover the whole federated learning process and to quantify the training behavior of the worker nodes completely.

7 Summary

In this paper, we propose a new digital watermarking-based training behavior authenticity supervision mechanism WM-TBM for decentralized federated learning, which motivates worker nodes to remain honest when participating in federated learning by supervising the authenticity of their training behavior. Specifically, the quantification of the authenticity of worker nodes' training behavior is accomplished by transposing the usage scenario of digital watermarking technology to decentralized federated learning worker node training behavior supervision. Theoretical analysis demonstrates that WM-TBM can save significant computational and communication resources compared to the existing scheme DDSM. The effectiveness of WM-TBM and the practicability of dataset replacement theory are then further verified through experiments, which effectively solve the problems in the current research on worker node training behavior supervision mechanism in decentralized federated learning oriented.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (grant numbers U2003206 and U20B2048), the Defense Industrial Technology Development Program (grant number 2020604B004), and the Heilongjiang Provincial Natural Science Foundation of China (grant number LH2021F016).

References

1. Jiang, H., et al.: An energy-efficient framework for internet of things underlaying heterogeneous small cell networks. IEEE Trans. Mob. Comput. **21**(1), 31–43 (2020)
2. Dai, X., et al.: Task co-offloading for D2D-assisted mobile edge computing in industrial internet of things. IEEE Trans. Industr. Inform. (2022)
3. Xiao, Z., et al.: TrajData: on vehicle trajectory collection with commodity plug-and-play OBU devices. IEEE Internet Things J. **7**(9), 9066–9079 (2020)
4. Long, W., et al.: Unified spatial-temporal neighbor attention network for dynamic traffic prediction. IEEE Trans. Veh. Technol. (2022)
5. Jiang, H., et al.: Joint task offloading and resource allocation for energy-constrained mobile edge computing. IEEE Trans. Mob. Comput. (2022)
6. Xiao, Z., et al.: Resource management in UAV-assisted MEC: state-of-the-art and open challenges. Wireless Netw. **28**(7), 3305–3322 (2022)
7. Wang, L., et al.: Multiple access mmWave design for UAV-aided 5G communications. IEEE Wirel. Commun. **26**(1), 64–71 (2019)
8. Chen, H., Chen, J., Ding, J.: Data evaluation and enhancement for quality improvement of machine learning. IEEE Trans. Reliab. **70**(2), 831–847 (2021)
9. Konen, J., Mcmahan, H.B., Ramage, D., et al.: Federated optimization: distributed machine learning for on-device intelligence. arXiv:1610.02527 [cs] (2016)
10. Wang, L., Wu, K., Hamdi, M.: Combating hidden and exposed terminal problems in wireless networks. IEEE Trans. Wireless Commun. **11**(11), 4204–4213 (2012)
11. Mcmahan, H.B., Moore, E., Ramage, D., et al.: Communication-efficient learning of deep networks from decentralized data. arXiv:1602.05629 [cs] (2016)
12. Mcmahan, H.B., Moore, E., Ramage, D., et al.: Federated learning of deep networks using model averaging. arXiv:1602.05629 (2016)

13. Yang, Q., Liu, Y., Chen, T., et al.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. **10**(2), 1–19 (2019)
14. Nishant, B.S., Kalapriya, K., Vijay, A., et al.: Ownership preserving AI Market Places using Blockchain. In: 2019 IEEE International Conference on Blockchain (Blockchain), pp. 156–165. IEEE, Atlanta (2020)
15. Lu, Y.-L., Huang, X.-H., Dai, Y.-Y., et al.: Blockchain and federated learning for privacy-preserved data sharing in industrial IoT. IEEE Trans. Industr. Inf. **16**(6), 4177–4186 (2019)
16. Kang, J.-W., Xiong, Z.-H., Niyato, D., et al.: Incentive mechanism for reliable federated learning: a joint optimization approach to combining reputation and contract theory. IEEE Internet Things J. **6**(6), 10700–10714 (2019)
17. Weng, J.-S., Weng, J., Zhang, J., et al.: DeepChain: auditable and privacy-preserving deep learning with blockchain-based incentive. IEEE Trans. Dependable Secure Comput. **18**(5), 2438–2455 (2021)
18. Wang, H., Sreenivasan, K., Rajput, S., et al.: Attack of the tails: yes, you really can backdoor federated learning. In: Advances in Neural Information Processing Systems. (2020)
19. Schyndel, R., Tirkel, A.Z., Osborne, C.F.: A digital watermark. In: 1st International Conference on Image Processing, pp. 86–90. IEEE, Austin (1994)
20. Uchida, Y., Nagai, Y., Sakazawa, S., et al.: Embedding watermarks into deep neural networks. In: 2017 ACM on International Conference on Multimedia Retrieval, pp. 269–277. ACM, New York (2017)
21. Rouhani, B.D., Chen, H., Koushanfar, F.: DeepSigns: an end-to-end watermarking framework for ownership protection of deep neural networks. In: Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems(ASPLOS), pp. 485–497. ACM, Providence (2019)
22. Adi, Y., Baum, C., Cisse, M., et al.: Turning your weakness into a strength: watermarking deep neural networks by backdooring. In: 27th USENIX Security Symposium, pp. 1615–1631. USENIX, Baltimore (2018)
23. Guo, J., Potkonjak, M.: Watermarking deep neural networks for embedded systems. In: 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 1–8. IEEE, San Diego (2018)
24. Jebreel, N.M., Domingo-Ferrer, J., Sánchez, D., et al.: KeyNet: an asymmetric key-style framework for watermarking deep learning models. Appl. Sci. **11**(3), 999–1021 (2021)
25. Zhang, J.-L., Gu, Z.-S., Jang, J.-Y., et al.: Protecting intellectual property of deep neural networks with watermarking. In: 2018 on Asia Conference on Computer and Communications Security, pp. 159–172. ACM, Incheon (2018)
26. Zhu, R., Zhang, X., Shi, M., et al.: Secure neural network watermarking protocol against forging attack. EURASIP J. Image Video Process. **37**(2020) (2020)
27. Li, Z., Hu, C.-Y., Zhang, Y., et al.: How to prove your model belongs to you: a blind-watermark based framework to protect intellectual property of DNN. In: The 35th Annual Computer Security Applications Conference, pp. 126–137. ACM, San Juan (2019)
28. Atli, B.G., Xia, Y.-X., Marchal, S., et al.: WAFFLE: watermarking in federated learning. arXiv:2008.07298 [cs] (2020)
29. Li, F.-Q., Wang, S.-L.: Towards practical watermark for deep neural networks in federated learning. arXiv:2105.03167 [cs] (2021)
30. Chen, M., Niu, X., Yang, Y.: The reach developments and applications of digital watermarking. J. Commun. **22**(5), 71–79 (2001)
31. Boenisch, F.: A systematic review on model watermarking for neural networks. Front Big Data **4**, 1–16 (2021)

32. Wang, T., Kerschbaum, F.: Robust and undetectable white-box watermarks for deep neural networks. arXiv:1910.14268 [cs] (2019)
33. Somy, N.B., et al.: Ownership preserving AI market places using blockchain. In: 2019 IEEE International Conference on Blockchain (Blockchain). IEEE (2019)