DCHM: Dynamic Collaboration of Heterogeneous Models Through Isomerism Learning in a Blockchain-Powered Federated Learning Framework

Zhihao Hao[1,2], Bob Zhang[2,3]*, Haisheng Li[1]*

[1]School of Computer Science and Artificial Intelligence & Beijing Key Laboratory of Commercial Data Security Protection and Intelligent Governance, Beijing Technology and Business University, Beijing, China
[2]Department of Computer and Information Science, Faculty of Science and Technology, University of Macau, Taipa, Macau
[3]Centre for Artificial Intelligence and Robotics, Institute of Collaborative Innovation, University of Macau, Taipa, Macau
hao.zhihao@connect.um.edu.mo, bobzhang@um.edu.mo, lihsh@btbu.edu.cn

Abstract

Solutions to time-varying problems are crucial for research areas such as predicting changes in human body shape over time. While recurrent neural networks have made significant advancements in this field, their reliance on centralized processing has led to challenges such as model silos and data isolation. In response, distributed AI systems like federated learning have emerged to facilitate dynamic collaboration among models; however, they still depend on central coordinators, which pose risks to system security and efficiency. Moreover, traditional federated learning primarily supports homogeneous models and lacks effective strategies for the interaction of heterogeneous models. To address these limitations, we propose a novel method called Dynamic Collaboration of Heterogeneous Models (DCHM), based on Isomerism Learning, which leverages a consortium blockchain network to enhance model credibility and facilitate coordination among heterogeneous models. Additionally, we introduce a Distributed Hierarchical Aggregation (DHA) algorithm that enables permissioned nodes within each group to aggregate local model results and share them for standardized processing. After several iterative cycles, these nodes perform secondary integration of local results to produce global outcomes. Experimental results demonstrate that DCHM effectively analyzes the temporal variability of body shape changes with high efficiency.

Introduction

The changes in human body shape exhibit a time-varying nature, necessitating the use of effective research methods. Consequently, recurrent neural network (RNN) models have been widely applied across various fields due to their strong performance in processing time series data (Weerakody et al. 2021). The zeroing neural network (ZNN), a specialized RNN model designed to tackle dynamic analysis challenges, can observe trends in time-varying parameters through its time derivatives, allowing it to solve dynamic problems without residuals. However, these methods do have certain limitations. For instance, the performance of centralized ZNN models is significantly influenced by factors such as model scale, computational power, and data availability (Yuan et al. 2022), which can be difficult to obtain due to privacy concerns. Many individuals may be unwilling to share their body images online. Therefore, when confronted with the prevalent issues of model and data silos, centralized models cannot guarantee stable performance, which has become a critical factor impacting the implementation of centralized artificial intelligence (Hao et al. 2023b).

Using distributed artificial intelligence (DAI) models can effectively address these challenges. Traditional DAI models have been developed to meet the increasing demands for scalability and computational power. This is particularly true for Federated Learning (FL), which is widely regarded as the next evolution of DAI. In FL, data does not need to be shared; instead, each client uses local private data to train the model independently before sharing only the parameters with a central server. This server then aggregates the parameters for model updates, allowing for accurate tracking of dynamic changes in time-varying problems through collaborative task completion. However, this approach primarily supports the distribution of homogeneous models without accounting for model heterogeneity. Additionally, it still relies on a central server for critical operations, meaning that any corruption of the server could compromise the entire model.

In this context, decentralization is essential and can be effectively implemented using blockchain technology, which serves as a foundational system due to its fully distributed nature. Additionally, the non-modifiable and traceable characteristics of blockchain establish trust between untrusted nodes (clients) within the network. However, issues arising from the consistency of node authority in permissionless blockchains can lead to inefficiencies (Ferdous, Chowdhury, and Hoque 2021). Therefore, dividing node authority within the network has emerged as a viable solution to enhance efficiency, as seen in private blockchains and consortium blockchains (Moore et al. 2023). This approach ensures that the model remains decentralized while aligning the authority distribution of nodes with the operational needs at different levels of the DAI model.

Using distributed artificial intelligence (DAI) models can effectively address these challenges. Traditional DAI models have been developed to meet the increasing demands for scalability and computational power. This is particularly true for Federated Learning (FL), which is widely regarded as the next evolution of DAI. In FL, data does not need to be

*Corresponding Author

shared; instead, each client uses local private data to train the model independently before sharing only the parameters with a central server. This server then aggregates the parameters for model updates, allowing for accurate tracking of dynamic changes in time-varying problems through collaborative task completion. However, this approach primarily supports the distribution of homogeneous models without accounting for model heterogeneity. Additionally, it still relies on a central server for critical operations, meaning that any corruption of the server could compromise the entire model.

In this context, decentralization is essential and can be effectively implemented using blockchain technology, which serves as a foundational system due to its fully distributed nature. Additionally, the non-modifiable and traceable characteristics of blockchain establish trust between untrusted nodes (clients) within the network. However, issues arising from the consistency of node authority in permissionless blockchains can lead to inefficiencies (Ferdous, Chowdhury, and Hoque 2021). Therefore, dividing node authority within the network has emerged as a viable solution to enhance efficiency, as seen in private blockchains and consortium blockchains. This approach ensures that the model remains decentralized while aligning the authority distribution of nodes with the operational needs at different levels of the DAI model.

As a promising paradigm in distributed artificial intelligence (DAI), the emergence of isomerism learning offers significant potential for addressing many of the challenges described above (Hao et al. 2023a). Unlike traditional "blockchain-assisted artificial intelligence" methods, which primarily rely on blockchain to ensure data credibility and thus improve AI model performance (Hao et al. 2021), isomerism learning utilizes the inherent capabilities of blockchain technology to achieve efficient collaboration among heterogeneous models in distributed machine learning frameworks. This transformative approach can therefore be considered a "blockchain-powered artificial intelligence" method, as it goes beyond merely supporting data integrity and instead directly empowers model coordination and system functionality.

Isomerism learning departs from traditional approaches that often focus solely on homogeneous model setups. Instead, it actively supports and encourages diverse model architectures across nodes, enabling these heterogeneous models to tackle different components of a complex problem by leveraging their unique strengths. This flexibility ensures that heterogeneous learning frameworks can fully exploit blockchain's immutable, decentralized, and traceable properties to establish trust among untrusted nodes while simultaneously improving the robustness and scalability of the system. Moreover, the decentralized nature of blockchain-powered isomerism learning eliminates the need for a single central server, thereby removing a critical point of failure and enhancing the overall security and efficiency of the network. By integrating blockchains ability to assign and verify node authority within a consortium network, isomerism learning effectively balances distributed autonomy with the operational demands of dynamic AI systems. This approach not only promotes inter-node trust and collaboration but also provides an adaptive and scalable framework for addressing time-varying problems.

Motivated by these challenges, we propose a Dynamic Collaboration of Heterogeneous Models (DCHM) based on isomerism learning, which leverages a consortium blockchain network to address the time-varying challenges associated with body shape change analysis. To tackle issues such as model heterogeneity (Kairouz et al. 2021), our proposed method employs various computational models to solve problems before collaboratively integrating their results. This approach eliminates the need for aggregating model parameters, as only the computed results are uploaded to the blockchain for sharing. Each iteration ensures the participation of all nodes through weight assignments to the results of each model, facilitating efficient collaboration among heterogeneous models. The main contributions of this paper are summarized as follows:

- We introduce a novel Dynamic Collaboration of Heterogeneous Models (DCHM) to effectively address the increasing challenges faced by centralized models, enabling the resolution of time-varying problems through collaboration among nodes equipped with heterogeneous models.
- We develop a distributed hierarchical aggregation method that utilizes both permissioned and permissionless nodes within the consortium blockchain, transforming the centralized local solution process into a distributed global solution process.
- Extensive experiments demonstrate the superiority of the proposed method, showcasing its capability to outperform many state-of-the-art models developed within a federated learning framework.

Architecture of the Proposed System

In a consortium blockchain network, nodes can be categorized into permissioned and permissionless nodes based on the consensus mechanism. Consequently, during the dynamic aggregation of distributed results, the different operation levels of these two types of nodes allow the overall architecture to be structured into two layers (as shown in Figure 1). This hierarchical processing incorporates the master-worker architecture of federated learning (FL) (Yang et al. 2019b) within a peer-to-peer framework. The system can be organized into different groups based on the number of permissioned nodes, with each permissioned node corresponding to a specific number of permissionless nodes. In each group, I permissionless nodes with varying computational models can collaboratively address the overall problem with the assistance of one permissioned node. Within a specific group, permissionless nodes are equipped with identical models, which differ from those in other groups. Importantly, relevant information is recorded in a shared ledger accessible to all parties, enhancing trust among anonymous nodes without the need for a third party. Additionally, the security of the model is maintained, as the recorded information contains only the results after each iteration and does not disclose the

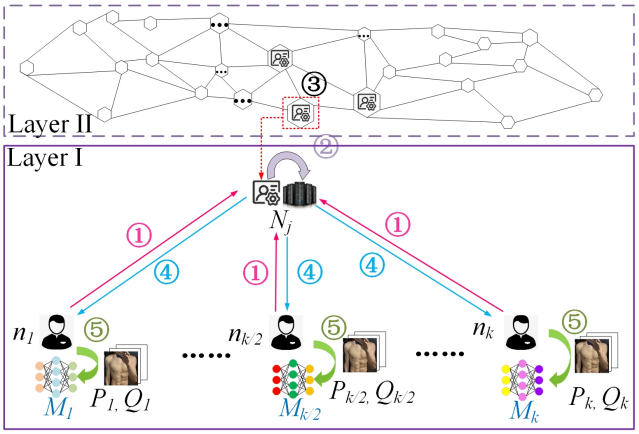

Figure 1: Architecture for the proposed system.

specific parameters of the local models. The process of the proposed system can be divided into five steps as follows:

Step 1: Permissionless nodes execute the solution process locally and send the corresponding parameters to the permissioned node in their group.

Step 2: The permissioned node performs the aggregation operation after receiving all results from the permissionless nodes within the group.

Step 3: The permissioned node shares the local results with other permissioned nodes for further aggregation to obtain the global results.

Step 4: The permissioned node sends the aggregated results back to the permissionless nodes.

Step 5: Permissionless nodes update their local results in preparation for the next iteration.

This entire iteration process continues until the maximum number of iterations is reached. Subsequently, all nodes in the network share the parameters after executing the global solution process.

To enhance the overall security of the system, we have designed a verification mechanism to mitigate risks from malicious behaviors, such as nodes submitting tampered results for aggregation. All results recorded on the blockchain are transparent, with their integrity during transmission protected by the Digital Signature Algorithm (DSA) (Mohammed, Joudah, and Mohammed 2024). Permissioned nodes can collect results and identify outliers by removing any that deviate significantly from the overall result cluster. These deviations assist in locating and marking the sender of suspicious results. After multiple iterations, if a marked node's deviations constitute a high proportion of its total submissions, the system classifies it as a malicious node. Consequently, the system places this node in a "silent" state, preventing its participation in subsequent aggregation processes.

Building on this foundation, we introduce smart contracts to further enhance the systems security and efficiency. Through node authentication and incentive mechanisms, smart contracts ensure that only verified nodes participate in the process. They also manage data sharing and enforce access control policies to safeguard sensitive information during interactions. Additionally, smart contracts support programmable workflows, enabling process automation and result verification, which enhance operational efficiency and facilitate seamless collaboration among nodes. However, the security of smart contracts presents significant challenges, primarily due to vulnerabilities arising from code defects, insufficient auditing, and the inherent complexity of the code. To address these issues, we adopted the method outlined in (Hao et al. 2023c), utilizing AI-generated smart contract code templates. This approach parallels current generative AI (Feuerriegel et al. 2024) techniques and improves the usability of smart contract code while reducing the likelihood of errors. As a result, it strengthens the security of smart contracts and the overall blockchain system.

Furthermore, in step 3 of the aggregation process, potential issues such as packet loss and noise during result broadcasting can impact data accuracy. To address these challenges and reduce the risk of tampering during transmission, we enhanced the consensus mechanism based on the Practical Byzantine Fault Tolerance (PBFT) algorithm, as described in (Mao et al. 2019). Specifically, after the permissioned node completes the global aggregation operation, it rebroadcasts the aggregated results to other permissioned nodes. After a specified period, all permissioned nodes count the received results and select the one with the highest frequency as the input for the next iteration. This approach effectively reduces the error rate of results while improving their reliability and accuracy.

Zeroing Neural Network (ZNN)

To enhance clarity, we have carefully organized all the variables utilized in this section and presented them in a structured table format, as illustrated in Table 1. This table includes the variables along with their corresponding descriptions, aiming to simplify the understanding of the proposed method and facilitate an overall comprehension of the research framework.

Variables	Descriptions
$X(t)$	Unknown matrix
$P(t)$, $Q(t)$	Time-varying parameter matrices
$G(t)$	Theoretical value of error
$\dot{G}(t)$, $\dot{P}(t)$, $\dot{Q}(t)$	Time derivative of $G(t)$, $P(t)$, $Q(t)$
$F(\cdot)$	Activation function
$P^{-1}(t)$	Inversion of $P(t)$
n_i	Permissionless nodes
N_j	Permissioned nodes
k	Iteration number
$\theta_{j,k}$	Update step of the jth permissioned node
$W_{j,k}$	Weight assigned
Θ_k	Total update step
M_i	Local model
r	Computation round
δ_r	Permissionless nodes set

Table 1: Descriptions of Variables Utilized.

When the external environment remains unchanged and we focus solely on the influence of time, the time-varying linear equation can be described as follows:

$$P(t)X(t) = Q(t), \tag{1}$$

where $X(t) \in \mathbb{R}^{n\times m}$ denotes the unknown matrix, $P(t) \in \mathbb{R}^{i\times n}$ and $Q(t) \in \mathbb{R}^{i\times m}$ are two time-varying parameter matrices. Among various permissioned nodes, various ZNN models are utilized to solve (1). The main idea of the ZNN model is to establish an error function as

$$G(t) = P(t)X(t) - Q(t), \tag{2}$$

whose theoretical value is zero. Then, the ZNN model defines the time derivative of $G(t)$ as

$$\dot{G}(t) = -\beta F(G(t)), \tag{3}$$

where $\gamma > 0$ and $F(\cdot)$ denotes an activation function. Based on (3), the ZNN model can make $G(t)$ converge to zero, where the corresponding $\xi(t)$ is the solution of (1). Inserting (2) into (3), the ZNN model can be formulated as

$$\dot{\xi}(t) = P^{-1}(t)(-\dot{P}(t)X(t)+\dot{Q}(t)-\gamma F(P(t)X(t)-Q(t))), \tag{4}$$

where $h = \beta\tau$ and $^{-1}$ denotes the inversion of a matrix (Jin et al. 2017). When implementing the ZNN model in digital circuit, (4) can be discretized at $t = k\tau$ as

$$X_{k+1} = X_k - \theta_k, \tag{5}$$

where $\theta_k = P_k^{-1}(-\tau\dot{P}_k X_k+\tau\dot{Q}_k-hF(P_k X_k-Q_k))$ is the update step, τ denotes the sampling interval, and $h = \tau\beta$. According to (5), the topology diagram of the neural units and corresponding connection architecture of the model are shown in Figure 2.

Based on the uniform formula of the ZNN model (5), various ZNN models can be updated. Specifically, the conventional ZNN model that employs a linear activation function, along with the first ZNN model referred to as the ZNN_1 model used in DCHM, can be expressed as follows:

$$X_{k+1} = X_k - \theta_{1,k}, \tag{6}$$

where $\theta_{1,k} = P_k^{-1}(-\tau\dot{P}_k X_k + \tau\dot{Q}_k - h(P_k X_k - Q_k))$. Besides this, the second ZNN model termed as ZNN_2 model

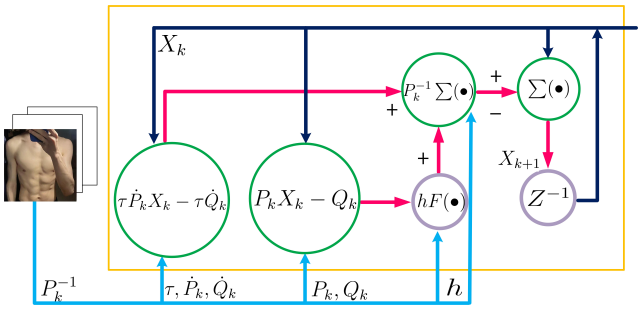

Figure 2: The ZNN model (5) for solving the dynamic linear equation.

estimates its $\theta_{2,k}$ with a bounded activation function, where the function can be expressed as

$$F(G(t)) = \begin{cases} 1, & \text{if } G(t) > 1, \\ G(t), & \text{if } -1 \le G(t) \le 1, \\ -1, & \text{if } G(t) < -1. \end{cases} \tag{7}$$

The third ZNN model, i.e., the ZNN_3 model in this paper is the Newton-Raphson iteration algorithm, which is a special ZNN model with $\dot{P}_k = 0$ and $\dot{Q}_k = 0$. The ZNN_3 model estimates its update step as $\theta_{3,k} = -P_k^{-1}(P_k\xi_k - Q_k)$. To improve the accuracy and robustness of the ZNN_3 model, an integration implemented model (termed as ZNN_4) was proposed, whose $\theta_{4,k} = -P_k^{-1}(P_k X_k - Q_k + \sum_{i=0}^{k}(P_i X_i - Q_i))$.

Distributed Hierarchical Aggregation (DHA)

The proposed integrated scheme for collaborative iteration is inspired by the FedAvg algorithm (Baumgart et al. 2024), which effectively addresses challenges in federated optimization, such as non-IID data, imbalanced data distributions, and unstable communication links. Research has demonstrated that FedAvg outperforms other federated optimization methods, including FedSGD (Fekri, Grolinger, and Mir 2022). Building on these insights, we introduce a Distributed Hierarchical Aggregation (DHA) algorithm, which operates within a consortium blockchain network. The DHA is specifically designed to optimize the finite sum of the objective functions of different local models. At the second layer, the objective function for J models can be expressed as follows:

$$\Theta_k(X_k) = \sum_{j=1}^{J} W_{j,k}\theta_{j,k}(X_k), \tag{8}$$

where $W_{j,k} = \frac{1/G_{j,k}(X_k)}{\sum_{j=1}^{J} 1/G_{j,k}(X_k)}$, $G_{j,k}(X_k)$ denotes the error of the jth group, and $\theta_{j,k}(X_k)$ can represent the update step of the jth permissioned node. Here, $W_{j,k}$ is used to maintain validity while guaranteeing decentralization, because it not only gives a large value to the models with good performance, but also keeps the involvement of the models with general effects. Besides this, $\theta_{j,k}(X_k)$ and $G_{j,k}(X_k)$ are estimated as

$$\theta_{j,k}(X_k) = \frac{1}{I}\sum_{i=1}^{I}\theta_{j,k}^{i}(X_k) \tag{9}$$

and

$$G_{j,k}(X_k) = \frac{1}{I}\sum_{i=1}^{I}G_{j,k}^{i}(X_k), \tag{10}$$

respectively. Moreover, $\theta_{j,k}^{i}(X_k)$ represents the update step and $G_{j,k}(X_k)$ denotes the error of the ith permissionless node in a group belonging to the first layer.

DHA achieves model updating and results aggregation through hierarchical integration, where the model iterates

Algorithm 1: The Distributed Hierarchical Aggregation (D-HA)

Permissionless nodes n_i executes:
Initialize the local model $M_{i,(i=1,2,...,I)}$ provided by N_j and randomly generate X_1
for each iteration k=1,2,...,K **do**
  Samples P_k and Q_k
  Obtain the generation result θ_k^i, G_k^i, and X_{k+1}
end for
Sends (θ_k^i,G_k^i) to the nearest permissioned node N_j

Permissioned nodes N_j executes:
for each model computation round r=1,2,.. **do**
  N_j determines the set δ_r of I permissionless nodes
  for $n_i \in \delta_r$ **in parallel do**
    Steps 1-7
  end for
  N_j executes (9)
  N_j executes (10)
  for j=1,2,...,J **do**
    N_j sends ($\theta_{j,k}(X_k)$, $G_{j,k}(X_k)$) to other permissioned nodes for aggregation as (8)
    N_j broadcasts the aggregated results $\Theta_k(X_k)$ to all permissionless nodes
  end for
end for

the solution as

$$X_{k+1} = X_k + \Theta_k(X_k) = \sum_{j=1}^{J} W_{j,k}\theta_{j,k}(X_k). \tag{11}$$

It allows permissionless nodes to update their local models through multiple iterations and sends the results to the permissioned nodes for aggregation after each iteration process. After obtaining the local optimization results within different groups, the permissioned nodes will broadcast the results to each other for secondary aggregation and obtain the results of the global solution. The process of DHA is summarized in Algorithm 1.

Convergence Analysis

To analyze the convergence of the DCHM model, we provide the following theorem.

Theorem: Utilizing the DCHM to solve the time-varying linear equation, the computed solution globally converges to the theoretical one.

Proof: According to (11), we have

$$X_{k+1} - X_k = [W_{1,k} \quad W_{2,k} \quad ... \quad W_{J,k}] \begin{bmatrix} \theta_{1,k}(X_k) \\ \theta_{2,k}(X_k) \\ ... \\ \theta_{J,k}(X_k) \end{bmatrix}. \tag{12}$$

In addition, since $\sum_{j=1}^{J} W_{j,k} = 1$, the left side of (12) can be expressed as

$$X_{k+1} - X_k = [W_{1,k} \quad W_{2,k} \quad ... \quad W_{J,k}] \begin{bmatrix} X_{k+1} - X_k \\ X_{k+1} - X_k \\ ... \\ X_{k+1} - X_k \end{bmatrix}. \tag{13}$$

Combining (12) and (13) leads to

$$[W_{1,k} \quad W_{2,k} \quad ... \quad W_{J,k}] \begin{bmatrix} X_{k+1} - X_k + \theta_{1,k}(X_k) \\ X_{k+1} - X_k + \theta_{2,k}(X_k) \\ ... \\ X_{k+1} - X_k + \theta_{J,k}(X_k) \end{bmatrix} = 0. \tag{14}$$

According to (14), the DCHM can be deemed as a linear combination of various ZNN models. Since each ZNN model exploited in the DCHM model globally converges to theoretical solution, the DCHM also has global convergence. The proof is thus complete.

Experiments Results and Analysis

In this section, we evaluate the performance of the DCHM. Due to the scarcity of relevant datasets collected over extended periods, most studies typically focus on using models for static analysis. However, it has become increasingly common for fitness enthusiasts to share progress photos online. To leverage this trend, we captured and processed images showcasing the body changes of fitness bloggers. Additionally, since the author also maintains a fitness routine, related photos illustrating the author's body changes were included in the dataset for supplementary comparison. This image collection spanned 90 days, during which we processed the images to extract relevant features, creating a matrix with time-varying parameters. As illustrated in Figure 3, the images displayed represent the author's body shape changes.

Since each image was captured at specific time points and does not reflect continuous changes over time, we constructed equation (1) with the time-varying parameter t as follows:

$$P(t) = \begin{bmatrix} s + \sin(2t) & s - \cos(2t) \\ s + \cos(2t) & s + \sin(2t) \end{bmatrix} \in \mathbb{R}^{2\times 2},$$
$$Q(t) = \begin{bmatrix} s + \sin(2t) + 2 & s + \cos(5t) + 2 \\ s - \cos(3t) & s + \sin(5t) + 1 \end{bmatrix} \in \mathbb{R}^{2\times 2}. \tag{15}$$

where s represents the sampling value from the image, it can also be considered to reflect oscillating changes in reality.

The experiments were conducted using MATLAB 2023b on a system equipped with 128 GB of memory, an Intel(R)

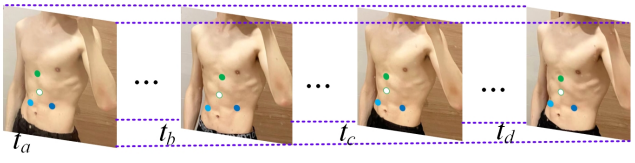

Figure 3: Temporal variations in body shape.

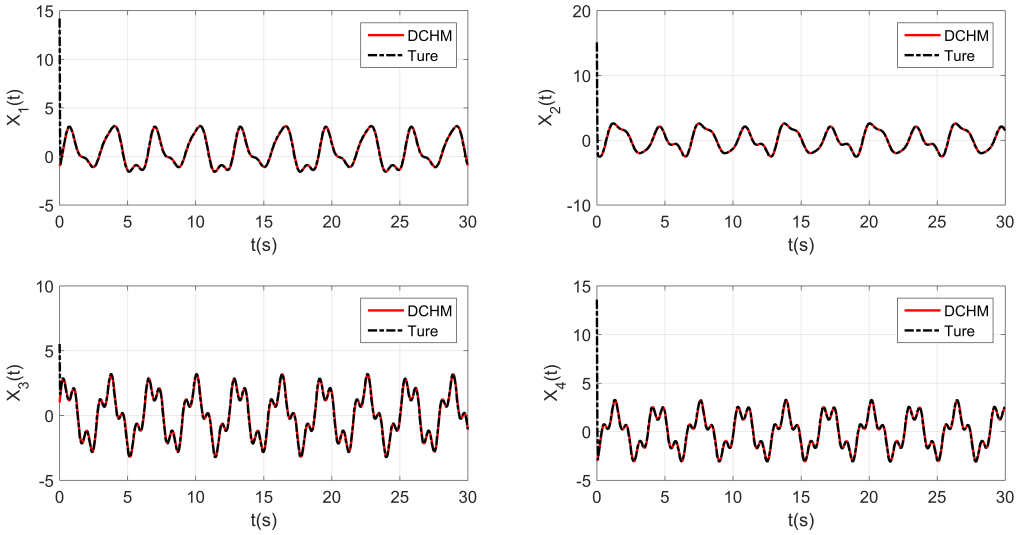

Figure 4: Comparison with its true value.

Results / Time	$X_1(t)$			$X_2(t)$			$X_3(t)$			$X_4(t)$		
	Calculated Results	True Value	Deviation	Calculated Results	True Value	Deviation	Calculated Results	True Value	Deviation	Calculated Results	True Value	Deviation
3	0.3941	0.5142	0.1201	-1.9070	-1.9515	0.0445	1.2380	1.2520	0.0140	-1.6520	-1.6700	0.0180
6	-1.3420	-1.3580	0.0160	-0.8806	-0.9180	0.0374	-1.1460	-1.1425	0.0035	-1.8240	-1.8120	0.0120
9	-0.7451	-0.7448	0.0003	-1.0440	-1.0820	0.0380	-0.6743	-0.6000	0.0743	-3.0580	-3.0850	0.0270
12	-0.9368	-0.9685	0.0317	-0.5801	-0.5657	0.0144	-0.6538	-0.6880	0.0342	-1.0740	-1.0130	0.0610
15	-1.0810	-1.1010	0.0200	0.3629	0.3000	0.0629	-2.7920	-2.7950	0.0030	-1.0560	-1.2000	0.1440
18	-1.1060	-1.0090	0.0970	-0.6935	-0.6938	0.0003	-1.7820	-1.7500	0.0320	-1.6800	-1.6970	0.0170
21	-0.5987	-0.6662	0.0675	1.3370	1.2920	0.0450	-1.6240	-1.7770	0.1530	0.6766	0.6833	0.0067
24	-1.5650	-1.5660	0.0010	0.0454	0.0483	0.0029	-3.1740	-3.2250	0.0510	0.5872	0.4000	0.1872
27	-0.1612	-0.1689	0.0077	1.6290	1.6300	0.0010	-1.4640	-1.4690	0.0050	0.2244	0.2136	0.0108
30	-0.9435	-0.9422	0.0013	1.4780	1.4980	0.0200	-1.0940	-1.0750	0.0190	2.4840	2.5110	0.0270
Avg			**0.0363**			**0.0266**			**0.0389**			**0.0511**

Table 2: Comparison of DCHM Model Calculated Results with True Value at Selected Time Points.

Core(TM) i3-3110M CPU, and an NVIDIA 1080Ti graphics card. The test network comprises 4 $\times$ 5 nodes, consisting of a total of 4 permissioned nodes, each linked to 5 permissionless nodes. The Ethereum platform provides the necessary tools to build a consortium blockchain (Zhang, Jin, and Cui 2018), which can be deployed across multiple virtual machines running Ubuntu Linux v18.04 within an OpenStack environment. Each virtual machine is allocated 1 virtual CPU core, 2 GB of memory, and 10 GB of persistent storage.

To mitigate the impact of communication on the integration process, all virtual machines are interconnected within a low-latency local network, achieving an average round-trip time of less than 1 millisecond for communication between nodes. Network behavior is monitored using the Python Web3 Library, while Elastic Search is utilized for storing block information. The sampling time interval for all models is set to 0.03 seconds, and to minimize the influence of varying computational times across models, the communication time is fixed at 0.035 seconds. With these parameters, we executed the DCHM model and observed the discrepancies between the models generated results and the true values over the interval of [0s, 30s].

The schematic diagram comparing the calculated solution to the theoretical solution synthesized by DCHM is presented in Figure 4. It is clear that each dynamic change value $X(t)$ at the four sampling points depicted in Figure 3 generated by the DCHM closely approximates its true value. Furthermore, the overall trend of the calculated solution aligns well with the true values. As shown in Table 2, the deviations of DCHM are 0.0363, 0.0266, 0.0389, and 0.0511, respectively. These low deviations reflect a high level of accuracy and reliability, indicating that the model effectively captures the underlying trends. Consequently, DCHM serves as a robust tool for conducting and facilitating the dynamic analysis of body shape. To demonstrate the effectiveness of the DCHM, we evaluated its steady-state error against state-of-the-art models, including Model_1 (Jin et al. 2018), Model_2 (Wang et al. 2022), Model_3 (Zhang, Ma, and Cai 2008), and Model_4 (Wang et al. 2021). All models were modified based on a federated learning prototype (Yang et al. 2019a) and were run for 968 rounds of integration, as shown in Figure 5. Our findings reveal that the DCHM exhibits a faster convergence speed, with results fluctuating within a certain range over time due to the collaborative influence of the integrated models. These oscillations reflect the interactions among the

Models \ Time	2.28	5.60	8.12	11.90	14.42	18.17	20.72	24.40	26.99	28.00
Model_1	0.01853	0.01947	0.02009	0.02004	0.01893	0.02025	0.01910	0.02160	0.02083	0.02190
Model_2	0.03618	0.03917	0.03965	0.04055	0.04013	0.04102	0.03952	0.04393	0.04115	0.04564
Model_3	0.08272	0.14790	0.16150	0.13650	0.11430	0.14540	0.13930	0.19570	0.14780	0.27690
Model_4	0.14960	0.15410	0.14850	0.15410	0.14870	0.15420	0.14870	0.15420	0.14850	0.14730
DCHM	**0.01435**	**0.01150**	**0.01431**	**0.01165**	**0.01430**	**0.01185**	**0.01449**	**0.01175**	**0.01474**	**0.01941**

Table 3: Comparison of errors of different models at selected time, which corresponds to specific rounds.

performances of different models, indicating that while DCHM converges quickly, it is influenced by the diverse characteristics of its heterogeneous components. This variability implies that changes in individual model performance can affect the overall system dynamics, thereby underscoring DCHM's adaptability in dynamic environments.

Due to the imbalance of the communication time and sampling time, as well as the change of calculation results, the curve corresponding to each model is oscillating within a range. It can be seen from Figure 5 that the proposed model is significantly better than the Model_3 and the Model_4. However, it is difficult to see the difference between the proposed model and the Model_1 and Model_2. Therefore, for better analysis, we selected the errors of all models at 65, 160, 232, 340, 412, 519, 592, 697, 771 and 800 rounds for comparison, as shown in Table 3. The average errors of the Model_3 and Model_4 model are 15.4802×10^{-2} and 15.079×10^{-2}, respectively. For the DCHM model, the average value is 1.3835×10^{-2}, which is the smallest compared to the average error of 2.0074×10^{-2} for Model_1 and 4.0694×10^{-2} for Model_2. Therefore, DCHM is very efficient in terms of computational accuracy.

In addition, we also use different metrics to better evaluate the performance of these different models. Here, Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) have been adapted because they can accurately measure the difference between the theoretical and calculated values. The value E_{MAE} of MAE can be defined as $E_{MAE} = \frac{1}{n}\sum_{i=1}^{n}|true_i - computed_i|$, where

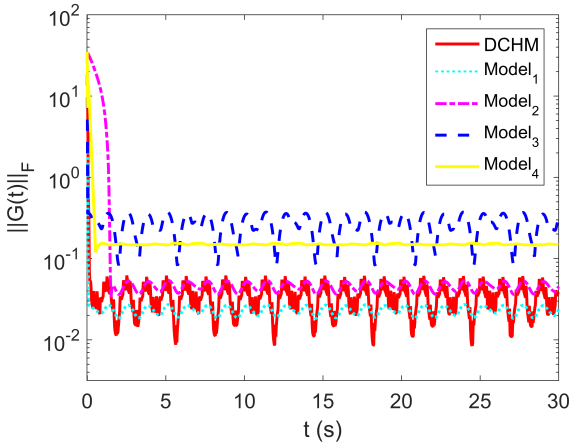

Figure 5: Comparison of different models.

Models	E_{MAE}	E_{RMSE}
Model_1	2.30×10^{-2}	2.31×10^{-2}
Model_2	4.55×10^{-2}	4.58×10^{-2}
Model_3	26.29×10^{-2}	27.47×10^{-2}
Model_4	14.93×10^{-2}	14.93×10^{-2}
DCHM	3.29×10^{-2}	3.53×10^{-2}

Table 4: Comparison between DCHM and other models.

$true_i$=0. And the value E_{RMSE} of RMSE can be expressed as: $E_{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(true_i - computed_i)^2}$. Based on these, the relevant values can be shown in Table 4. As illustrated in this table, the proposed DCHM model can outperform the many models like Model_2, Model_3 and Model_4 because it has the lowest value in terms of E_{MAE} and E_{RMSE}. Hence, the proposed method is effective. But there is still a slight gap compared to Model_1 (0.99×10^{-2} for MAE and 1.22×10^{-2} for RMSE). This is due to the weight distribution scheme adopted for decentralization. In the future, we will use higher-precision models for integration and try more weight distribution schemes to narrow the gap, such that the overall model performance is close to the higher-precision model.

Conclusion

In this paper, a distributed intelligent computing model based on heterogeneous learning, termed DCHM, is proposed to address challenges from temporal changes while resolving model and data silos and ensuring privacy protection. By implementing a consortium blockchain network, trusted interactions between unfamiliar nodes are facilitated without exposing sensitive model parameters, maintaining data integrity and security throughout the collaboration. Extensive experimental results show that the model matches the performance of leading high-precision models and significantly outperforms most state-of-the-art approaches, underscoring its superiority. Additionally, this method improves the performance of models that need substantial computing power, like large language models (LLMs). It decentralizes computational workloads, creating a distributed LLM (DLLM) that addresses the limitations of centralized systems and provides scalable solutions for large AI models. Distributing processing across nodes reduces the risk of overload and enhances reliability. This approach shows the potential to overcome barriers in computation-intensive applications and advance distributed artificial intelligence.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62277001), the Science and Technology Development Fund, Macao S.A.R (FDCT) 0028/2023/RIA1, the Beijing Natural Science Foundation (No.L233026), and the Scientific Research Program of Beijing Municipal Education Commission (KZ202110011017).

References

Baumgart, G. A.; Shin, J.; Payani, A.; Lee, M.; and Kompella, R. R. 2024. Not All Federated Learning Algorithms Are Created Equal: A Performance Evaluation Study. *arXiv preprint arXiv:2403.17287*.

Fekri, M. N.; Grolinger, K.; and Mir, S. 2022. Distributed load forecasting using smart meter data: Federated learning with Recurrent Neural Networks. *International Journal of Electrical Power & Energy Systems*, 137: 107669.

Ferdous, M. S.; Chowdhury, M. J. M.; and Hoque, M. A. 2021. A survey of consensus algorithms in public blockchain systems for crypto-currencies. *Journal of Network and Computer Applications*, 182: 103035.

Feuerriegel, S.; Hartmann, J.; Janiesch, C.; and Zschech, P. 2024. Generative ai. *Business & Information Systems Engineering*, 66(1): 111–126.

Hao, Z.; Wang, G.; Mao, D.; Zhang, B.; Li, H.; Zuo, M.; Zhao, Z.; and Yen, J. 2021. A novel method for food market regulation by emotional tendencies predictions from food reviews based on blockchain and saes. *Foods*, 10(6): 1398.

Hao, Z.; Wang, G.; Zhang, B.; Fang, L.; and Li, H. 2023a. An Isomerism Learning Model to Solve Time-Varying Problems Through Intelligent Collaboration. *IEEE/CAA Journal of Automatica Sinica*, 10(8): 1772–1774.

Hao, Z.; Wang, G.; Zhang, B.; Feng, Z.; Li, H.; Chong, F.; Pan, Y.; and Li, W. 2023b. A Novel Public Sentiment Analysis Method Based on an Isomerism Learning Model via Multiphase Processing. *IEEE Transactions on Neural Networks and Learning Systems*.

Hao, Z.; Zhang, B.; Mao, D.; Yen, J.; Zhao, Z.; Zuo, M.; Li, H.; and Xu, C.-Z. 2023c. A novel method using LSTM-RNN to generate smart contracts code templates for improved usability. *Multimedia Tools and Applications*, 82(27): 41669–41699.

Jin, L.; Li, S.; Hu, B.; Liu, M.; and Yu, J. 2018. A noise-suppressing neural algorithm for solving the time-varying system of linear equations: A control-based approach. *IEEE Transactions on Industrial Informatics*, 15(1): 236–246.

Jin, L.; Li, S.; Liao, B.; and Zhang, Z. 2017. Zeroing neural networks: A survey. *Neurocomputing*, 267: 597–604.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210.

Mao, D.; Hao, Z.; Wang, F.; and Li, H. 2019. Novel automatic food trading system using consortium blockchain. *Arabian Journal for Science and Engineering*, 44: 3439–3455.

Mohammed, Q. A. A.-S.; Joudah, M.; and Mohammed, H. 2024. A survey on digital signature schemes. In *AIP Conference Proceedings*, volume 3232. AIP Publishing.

Moore, E.; Imteaj, A.; Rezapour, S.; and Amini, M. H. 2023. A survey on secure and private federated learning using blockchain: Theory and application in resource-constrained computing. *IEEE Internet of Things Journal*.

Wang, G.; Hao, Z.; Zhang, B.; and Jin, L. 2022. Convergence and robustness of bounded recurrent neural networks for solving dynamic Lyapunov equations. *Information Sciences*, 588: 106–123.

Wang, G.; Huang, H.; Shi, L.; Wang, C.; Fu, D.; Jin, L.; and Xiuchun, X. 2021. A noise-suppressing Newton-Raphson iteration algorithm for solving the time-varying Lyapunov equation and robotic tracking problems. *Information Sciences*, 550: 239–251.

Weerakody, P. B.; Wong, K. W.; Wang, G.; and Ela, W. 2021. A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing*, 441: 161–178.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019a. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2): 1–19.

Yang, Q.; Liu, Y.; Cheng, Y.; Kang, Y.; Chen, T.; and Yu, H. 2019b. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3): 1–207.

Yuan, S.; Zhao, H.; Zhao, S.; Leng, J.; Liang, Y.; Wang, X.; Yu, J.; Lv, X.; Shao, Z.; He, J.; et al. 2022. A Roadmap for Big Model. *arXiv preprint arXiv:2203.14101*.

Zhang, H.; Jin, C.; and Cui, H. 2018. A method to predict the performance and storage of executing contract for ethereum consortium-blockchain. In *International Conference on Blockchain*, 63–74. Springer.

Zhang, Y.; Ma, W.; and Cai, B. 2008. From Zhang neural network to Newton iteration for matrix inversion. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 56(7): 1405–1415.