Cloud Computing and Data Science
https://ojs.wiserpub.com/index.php/CCDS/

UNIVERSAL WISER PUBLISHER

Research Article

FLUDITY Unleashed: Dynamic Blockchain-Enabled Federated Learning Architecture for Tactile IoT Applications

Omar Alnajar

Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
E-mail: oorabialnajar@stu.kau.edu.sa

Received: 14 June 2025; **Revised:** 30 July 2025; **Accepted:** 30 July 2025

Abstract: The Tactile Internet of Things (TIoT) demands ultra-reliable, low-latency communication for real-time haptic applications in domains such as remote surgery and human-robot collaboration. Conventional Federated Learning (FL) architectures-centralized, hierarchical, or purely decentralized-each suffer from bottlenecks in scalability, latency, or trust. We propose Federated Learning Using Distributed Infrastructure for TIoT (FLUDITY), a hybrid FL framework leveraging Blockchain, Multi-Edge Computing (MEC), and a Dynamic Aggregation Decision Algorithm (DADA). Evaluated via the TACTO (a fast and flexible tactile simulator) tactile simulator in a remote grasping task, FLUDITY achieves up to 30% reduction in convergence time (880 s vs. 1,258 s), 15% fewer Floating Point Operations (FLOPs) (74.8×10^{12} vs. 88×10^{12}), and 25% lower communication overhead (1,575 MB vs. 2,099.9 MB) compared to static blockchain FL.

***Keywords*:** federated learning, tactile internet of things, block chain, 6G

1. Introduction

The advent of the Tactile Internet of Things (TIoT) has opened up new possibilities for real-time applications that require ultra-reliable and low-latency communication [1]. TIoT refers to a network of interconnected devices equipped with tactile sensors and actuators, enabling them to perceive and interact with the physical world in a haptic manner. This emerging field has gained significant attention due to its potential to revolutionize critical sectors such as healthcare, manufacturing, transportation, and entertainment.

The importance of TIoT applications lies in their ability to enable precise and immersive experiences, which were previously unimaginable. For instance, in healthcare, TIoT can enable remote surgeries [2], where surgeons can manipulate robotic arms with haptic feedback in real-time, overcoming physical distance barriers and providing access to expert care. Similarly, in manufacturing, TIoT can enhance human-robot collaboration [3] by enabling workers to control robotic arms with a sense of touch, leading to increased efficiency and safety.

However, realizing the full potential of TIoT applications comes with significant challenges. One of the major hurdles is the stringent requirements for ultra-low latency and ultra-reliable communication [4]. Conventional network architectures, including 5G, are not inherently designed to meet these demanding needs. The current network infrastructures are limited by factors such as latency, bandwidth constraints, and centralized processing, hindering the seamless deployment of TIoT applications.

DOI: https://doi.org/10.37256/ccds.7120267465

To address these challenges, Deep Learning (DL) has emerged as a promising solution to extract meaningful insights from the vast amounts of tactile data generated by TIoT devices. DL algorithms, particularly Convolutional Neural Networks (CNNs), have shown remarkable success in various sensory tasks, including object recognition and gesture detection [5-6]. By leveraging DL, TIoT applications can benefit from advanced perception capabilities, enabling more intelligent and context-aware interactions.

However, DL also presents limitations when applied to TIoT. The resource-intensive nature of DL algorithms poses challenges in terms of computational power, energy efficiency, and latency. TIoT devices, often resource-constrained, struggle to accommodate the computational demands of DL models. Additionally, privacy and security concerns arise when sensitive data is transmitted to centralized servers for processing [7].

To overcome the limitations of DL for TIoT, Federated Learning (FL) has emerged as a groundbreaking approach [8]. FL enables collaborative model training across distributed TIoT devices without the need to transmit raw data to a centralized server. Instead, FL allows devices to train models locally using their local data and only exchange model updates, thus preserving data privacy and reducing communication overhead.

While FL offers promising solutions, the legacy architecture of centralized FL poses its own limitations [9-10]. Centralized FL relies on a single server to aggregate local updates and build a global model, leading to potential bottlenecks and scalability issues. Additionally, the centralized approach lacks fault-tolerance and is vulnerable to single points of failure. These limitations necessitate the exploration of novel architectures that leverage cutting-edge enablers to empower FL in TIoT applications.

In the context of 6G networks, which are envisioned to support a wide range of emerging technologies, several key enablers come into play to address the limitations of centralized FL. These enablers include Blockchain, Dynamic and Decentralized Algorithms (DADA), Software-Defined Networking (SDN), Network Function Virtualization (NFV), and Multi-Edge Computing (MEC).

Blockchain technology provides a decentralized and immutable ledger [11], ensuring data integrity and transparency in FL processes. DADA algorithms enable dynamic and decentralized decision-making [12], optimizing the allocation of FL tasks and resources in a distributed manner. SDN and NFV play a crucial role in dynamically managing network resources and orchestrating FL workflows [13]. MEC, with its distributed computing capabilities at the network edge [14], enables localized processing and reduced latency, making it ideal for FL tasks in TIoT applications.

By harnessing the power of these enablers, Federated Learning Using Distributed Infrastructure for TIoT (FLUDITY) architecture emerges as a paradigm shift in FL for TIoT applications. FLUDITY leverages the decentralized nature of Blockchain, the intelligent decision-making of DADA algorithms, the flexible and programmable network management of SDN and NFV, and the low-latency processing capabilities of MEC to overcome the limitations of legacy architectures and enable seamless and efficient FL in TIoT applications.

To systematically address these challenges, we present a comparative analysis of federated learning architectures in Table 1, highlighting FLUDITY's advantages in reliability, latency, and dynamic adaptation. While existing approaches-Centralized Federated Learning (CFL), hierarchical, and decentralized (P2P/Blockchain)-each exhibit limitations in scalability, security, or rigid topologies, FLUDITY uniquely combines adaptive aggregation via DADA with blockchain-backed integrity checks. This hybrid design ensures ultra-low latency through context-aware model placement while maintaining end-to-end privacy, overcoming the single-point failures of CFL and the high overheads of pure decentralized solutions. As demonstrated in later sections, this architectural innovation positions FLUDITY as the optimal framework for TIoT applications in 6G networks.

The main contributions of this work are:

1) **Hybrid FL Architecture:** We design FLUDITY, combining Blockchain, SDN/NFV, MEC, and a novel DADA for context-aware aggregation.

2) **Dynamic Aggregation Decision Algorithm:** We for-malize a weighted scoring metric that jointly optimizes FL task requirements and real-time network conditions.

3) **Quantitative Gains:** We demonstrate up to 30% reduction in convergence time, 15% fewer FLOPs, and 25% lower communication overhead compared to static blockchain-based FL.

The structure of this work is organized as follows: Section 2 provides background on the key concepts and technologies used. Section 3 details the main components of the FLUDITY architecture, followed by an explanation of the workflow in 4. The security robustness of the proposed solution is analyzed in 5. Next, 6 presents the experiments and discusses the results. Finally, the conclusion is presented in 7.

Table 1. Comparative analysis of federated learning architectures for tactile internet of things

Metric	Centralized FL (CFL)	Hierarchical FL	DFL (P2P)	DFL (Blockchain)	FLUDITY
Reliability	Single point of failure; vulnerable to server crashes [15]	Partial resilience; edge failures disrupt local clusters [16]	High resilience; gossip protocols maintain connectivity [17]	Byzantine fault tolerance via consensus mechanisms [18]	Hybrid fault tolerance: Dynamic failover between cloud/edge/ blockchain [12]
Latency	High E2E latency (150-300 ms) [19]	Moderate latency (50-100 ms) via edge aggregation [20]	Unpredictable latency (20-200 ms) due to P2P hops [21]	High consensus delay (> 500 ms) [22]	Ultra-low latency (< 10 ms): DADA selects optimal aggregation point [19]
Security	Vulnerable to model poisoning attacks [23]	Edge aggregators susceptible to compromise [24]	Limited trust model between peers [25]	Immutable audit trail via distributed ledger [18]	End-to-end security: Blockchain + encrypted updates + verified communications [22]
Comm. Overhead	O(N) upstream/downstream to central server [26]	O(N) to edge + O(M) to cloud ($M \ll N$) [27]	$O(N^2)$ message complexity in dense networks [28]	High overhead from consensus protocols [22]	Context-aware optimization: Local P2P when $S_n > \tau$, else edge/cloud [13]
Scalability	Limited by server bandwidth [supports 10^3 devices] [29]	Scalable to 10 devices with hierarchical partitioning [16]	Highly scalable via distributed coordination [28]	Limited by blockchain throughput (100 TPS) [30]	Elastic scaling: SDN/NFV dynamic resource provisioning [15]
Privacy	Requires trusted central server [31]	Edge aggregators access model updates [24]	Local data never leaves devices [32]	Metadata exposure risks in public chains [18]	Multi-layer privacy: Local training + encrypted updates + private BC [33]
Dynamic Adaptation	Static cloud-based aggregation	Fixed hierarchical topology	Static peer connections	Fixed consensus protocol	Real-time optimization: DADA continuously adjusts aggregation point [12]

2. Related works

In this section, we provide a comprehensive background on the concepts related to the Tactile Internet of Things (TIoT), Deep Learning (DL), Federated Learning (FL), and the emerging 6G network. We also delve into the enablers that are crucial for the advancement and implementation of these technologies.

2.1 *TIoT*

The Tactile Internet of Things (TIoT) is a concept that involves the integration of IoT devices and haptic technologies to enable real-time interactions and haptic feedback [34-35]. TIoT holds great significance in various domains such as healthcare, robotics, and virtual reality, where low latency and ultra-reliable communication are critical. The ability to provide immersive and interactive experiences through haptic feedback has immense potential for enhancing user engagement and enabling new applications and services.

2.2 *DL*

Deep Learning (DL) is a subfield of machine learning that focuses on training artificial neural networks to automatically learn and extract complex patterns from data. DL has demonstrated remarkable success in various domains such as computer vision and natural language processing [36]. In the context of TIoT, DL can be leveraged to analyze the vast amount of sensory data collected by tactile devices and extract meaningful insights.

However, DL poses challenges in terms of computational power, energy efficiency, and latency. TIoT devices, often resource-constrained, struggle to accommodate the computational demands of DL models [37]. Moreover, privacy and security concerns arise when sensitive data is transmitted to centralized servers for processing [33, 38].

2.3 *FL*

Federated Learning (FL) is an emerging approach that enables collaborative training of machine learning models on decentralized devices without the need for centralized data aggregation [39]. FL offers several advantages for TIoT applications, including privacy-preserving machine learning and reduced communication overhead. By training models locally on TIoT devices, FL ensures data privacy while allowing devices to contribute their knowledge to a global model [40]. However, legacy FL architectures have limitations in terms of scalability, communication efficiency, and coordination [41-42].

2.4 *6G networks*

The 6G network is the next generation of wireless communication systems, envisioned to provide unprecedented connectivity, capacity, and low-latency communication [43]. With the exponential growth of TIoT devices and the increasing demand for real-time interactions, current network architectures, such as 5G, are limited in fulfilling the extreme requirements of TIoT applications.

6G aims to address these limitations and provide the necessary infrastructure to support emerging technologies like TIoT, virtual reality, augmented reality, and autonomous systems [44]. It leverages advanced technologies such as mmWave communication, massive MIMO, network slicing, and edge computing to enable ultra-reliable and low-latency communication, meeting the stringent requirements of TIoT applications.

1) Multi-Edge Computing (MEC) is a paradigm that brings computing resources closer to the network edge, enabling faster processing and lower latency for TIoT applications [45]. By deploying computational capabilities at the edge of the network, MEC reduces the dependence on distant cloud servers, allowing TIoT devices to offload computational tasks, including DL inference, to nearby edge servers. This enables faster response times and enhances real-time interactions in TIoT scenarios.

2) Software-Defined Networking (SDN) is an architecture that separates the network control plane from the data plane, providing centralized control and management of network resources [46]. In the context of TIoT, SDN enables dynamic network provisioning, efficient resource allocation, and traffic prioritization, ensuring low-latency and reliable communication for DL and FL tasks.

3) Network Function Virtualization (NFV) is a technology that virtualizes network functions, allowing them to run on standard hardware instead of dedicated appliances [47]. In TIoT scenarios, NFV enhances the agility, scalability, and resource efficiency of the network infrastructure, enabling flexible deployment and scaling of network functions, in cluding DL model deployment and FL coordination.

4) Blockchain is a distributed and decentralized ledger technology that provides a secure and transparent mechanism for data storage and transactions [48]. In TIoT applications, Blockchain can enhance data integrity, privacy-preserving mechanisms, and decentralized control. It enables secure data sharing, trust establishment, and auditability, enhancing the reliability and privacy of DL and FL processes in TIoT environments.

The convergence of TIoT, FL, SDN, NFV, MEC, and the upcoming 6G network opens up new opportunities to address the unique requirements and challenges of real-time haptic communication and interaction. By leveraging these technologies, it becomes possible to achieve low-latency, ultrareliable, and secure TIoT systems, revolutionizing various critical applications and paving the way for a more connected and immersive future.

3. FLUDITY architecture

This section elaborates on the proposed FLUDITY architecture, its components, the roles of each one, and the layers of the architecture. This name (FLUDITY) highlights the fluid nature of architecture, emphasizing the integration of federated learning, Blockchain, and dynamic decision-making. It also references the involvement of IoT and tactile clients. Figure 1 shows this architecture where its components are described as follows:

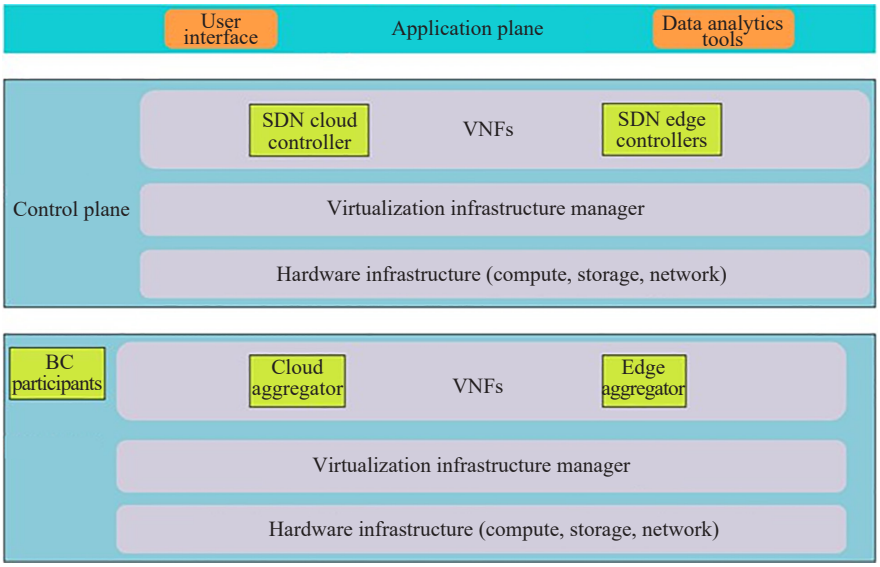

Figure 1. FLUDITY architecture

3.1 *Main components*

1) Cloud Aggregator: The architecture includes a centralized server located in the data plane that acts as a coordinator and aggregator for the federated learning process. It manages the overall training process, coordinates communication between devices, and aggregates the locally trained models.

2) SDN Cloud Controller: This entity is located in the control plane to manage and control the network infrastructure, including traffic routing, resource allocation, and monitoring of network conditions.

3) Edge Aggregators: Located in the data plane, these devices aggregate local updates from IoT and tactile devices and send them to the centralized server for model aggregation. By performing local aggregation of updates from nearby IoT devices or edge nodes, it will reduce the need for transmitting data to the centralized server.

4) SDN Edge Controller: located in the control plane, specifically in the edge layer, responsible for controlling and coordinating edge aggregators and managing FL task execution.

5) TIoT Devices: Located in the data plane, capture and pre-process data, perform local model training, generate local updates, and communicate with other planes.

6) Blockchain Platform: Located in the data plane where participating TIoT devices act as nodes in the Blockchain network. This platform provides a decentralized and secure platform for managing FL tasks, maintaining transaction records, and ensuring trust and privacy.

7) Virtualized Network Functions (VNFs): Located in the data plane, these software-based functions perform specific network tasks, such as data collection and analysis, FL task distribution, and data processing. The role of these various entities depends on the stages of the framework, which will be explained in the upcoming section as follows:

• Stage 1: Comparison of FL Task Score and Network Conditions:

The Data Collection and Analysis VNF collects and analyzes data related to FL tasks and network conditions.

The FL Task Scoring VNF and the Network Condition Scoring VNF calculate the scores for FL tasks and network conditions, respectively.

• Stage 2: FL Task Execution and Convergence:

The FL Task Distribution VNF is responsible for distributing FL tasks to the appropriate entities in the architecture.

Edge and Cloud aggregators These main entities have been explained as separate components since the beginning. However, it is considered a VNFs as well.

3.2 *The architecture from the three planes' perspective*

In general, the placement of devices, controllers, and applications depends on their roles and responsibilities within the overall architecture:

1) Data Plane: Devices and resources involved in data processing, forwarding, and local computations are typically placed in the data plane. This includes IoT devices, edge aggregators, and other network elements responsible for handling data at the network's edge.

2) Control Plane: The control plane consists of controllers responsible for managing and controlling the network infrastructure. These controllers make decisions, enforce policies, and coordinate the network's operation. In the FLUDITY architecture, the cloud SDN controller resides in the control plane.

3) Application Plane: The application plane encompasses the FL application and related components that are responsible for executing the FL tasks and utilizing the resources provided by the data and control planes. This includes the FL task management, model training, and inference processes. This layer can include user interfaces, data analytics tools, or any other software that interacts with the federated learning system:

• **User Interface:** Provides an interface for users to interact with the federated learning system, such as initiating or monitoring FL tasks, viewing the training progress, or accessing the trained models.

• **Data Analytics Tools:** Offers tools or functionalities to analyze and derive insights from the results of federated learning. This can include visualizations, statistical analysis, or further processing of the trained models.

The Application Plane primarily focuses on the front end applications and services that interact with the federated learning system. It serves as the interface for users or other systems to access the functionality provided by the FL architecture.

4. FLUDITY workflow

In this section, we will go through the roles of the most important FLUDITY architecture component, which is the SDN Cloud Controller. This controller contains a decision-making module that applies Dynamic Aggregation Decision Algorithm (DADA) for scoring the FL tasks' requirements V.S the score of real-time network conditions. Then, the details of these FL requirements and network criteria are elaborated along with numerical examples. Finally, the entire workflow steps containing the two stages are addressed.

4.1 *Cloud SDN controller's role*

In the proposed architecture, the responsibility for managing the decision-making process and coordinating with all participants primarily lies with the centralized server. The Cloud SDN Controller acts as the coordinator and orchestrator of the federated learning process. Here's how the server can fulfill this role as depicted in Figure 2:

1) Decision-Making Module: The Cloud SDN Controller includes a decision-making module that evaluates the network conditions, device capabilities, and task requirements to determine the appropriate location of aggregation (centralized, edge aggregator, or between IoT devices). This module can employ algorithms or machine learning techniques to make dynamic decisions based on real-time conditions.

2) Communication and Coordination: The Cloud SDN Controller facilitates communication and coordination with all participants, including edge devices and potential edge aggregators. It establishes reliable communication channels, such as secure connections or protocols, to exchange information, instructions, and updates.

3) Task Assignment: The Cloud SDN Controller assigns specific tasks to edge devices based on their capabilities and proximity to enable efficient aggregation. It distributes the global model, collects local model updates, and manages the aggregation process according to the determined aggregation location.

4) Resource Management: The Cloud SDN Controller monitors the resource utilization of edge devices and ensures a fair distribution of computational resources. It may allocate tasks to devices based on their available resources, such as computational power, memory, or energy levels, optimizing the overall performance and resource utilization.

5) Iterative Process Control: The Cloud SDN Controller controls the iterative process of federated learning by coordinating the rounds or epochs of training. It manages the timing and synchronization of model updates, global

model distribution, and refinement across all participants. By taking on the role of the coordinator and decision-maker, the Cloud SDN Controller can efficiently manage the dynamic decision-making process, provide instructions, and maintain overall control over the federated learning process.

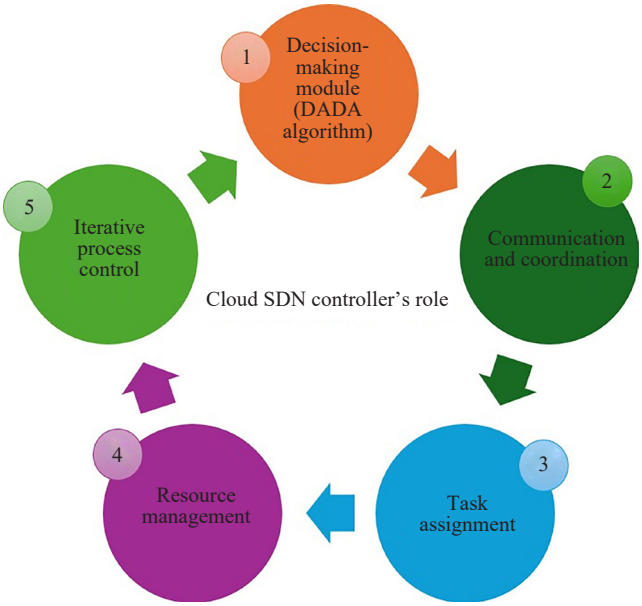

Figure 2. Cloud SDN controller's role

This enables effective coordination and collaboration among all participants while optimizing performance and resource utilization.

4.2 *DADA: Scoring of FL tasks v.s score of real-time network conditions*

Here's an approach of the decision-making Module:

1) Collect Real-Time Data: Gather real-time data related to network conditions, device capabilities, proximity, latency, or any other relevant criteria.

2) Assign Weights: Assign appropriate weights to each criterion based on their relative importance in the decision-making process.

3) Calculate Scores:

a) Calculate scores for each potential location (Cloud Centralized, Edge Aggregator, Blockchain-Enabled Decentralized Collaborative) using the collected real-FL task with the weighted score based on real-time data and weights.

b) Consider the contextual requirements and constraints of the FL tasks and the real-time network conditions.

4) Normalize Scores: Normalize the scores for both the FL tasks and the real-time data-based scores to bring them to a comparable scale (e.g., between 0 and 1).

5) Compare Scores:

a) Compare the weighted score of eachFL task with the weighted score based on real-time data.

b) Consider the contextual requirements and constraints of the FL tasks and the real-time network conditions.

6) Choose Optimal Location: Select the location with the highest overall score as the optimal location for FL aggregation, considering both the FL task-specific scores and the real-time data-based scores.

4.3 *Mathematical formulation*

The normalization and scoring processes are formalized as follows. Let S_t denote the FL task requirement score and S_n the network condition score, both initially on a scale of [a, b]. Normalization to [0, 1] is achieved via min-max scaling: time data and weights.

$$S_t^{'} = \frac{S_t - \min(S_T)}{\max(S_T) - \min(S_T)}, \quad S_n^{'} = \frac{S_n - \min(S_N)}{\max(S_N) - \min(S_N)}. \tag{1}$$

where S_T and S_N represent historical task and network score ranges. The decision metric C combines weighted normalized scores:

$$C = w_t \cdot S_t^{'} + w_n \cdot S_n^{'}. \tag{2}$$

Here, w_t and w_n are weights ($w_t + w_n = 1$) reflecting the relative importance of task requirements vs. network conditions. The optimal location maximizes C:

$$L^{*} = \underset{L \in \{\text{ Centralized, Edge, Decentralized }\}}{\arg\max} C_L. \tag{3}$$

By incorporating FL task scoring into the decision-making algorithm, the system can dynamically consider both the real-time network conditions and the specific requirements of each FL task. This enables a more comprehensive decision-making process, taking into account the changing nature of the FL tasks and the dynamic network conditions.

Algorithm 1 DADA: Dynamic Aggregation Decision Algorithm
Require: S_t, S_n (raw scores), $W = [w_t, w_n]$ (weights)
Ensure: Optimal aggregation location L^{*}
1: $S_t^{'} \leftarrow \text{MinMaxNorm}(S_t)$ {Eq. 1}
2: $S_n^{'} \leftarrow \text{MinMaxNorm}(S_n)$
3: $C_{\text{central}} \leftarrow w_t \cdot S_t^{'} + w_n \cdot S_n^{'}$
4: $C_{\text{edge}} \leftarrow w_t \cdot S_t^{'} + w_n \cdot \left(1 - \left|S_t^{'} - S_n^{'}\right|\right)$ {Penalize score divergence}
5: $C_{\text{decent}} \leftarrow w_n \cdot S_n^{'}$ {Network-critical}
6: $L^{*} \leftarrow \arg\max (C_{\text{central}}, C_{\text{edge}}, C_{\text{decent}})$
7: **return** L^{*}

4.4 *Optimizing weight assignment*

The weights w_t and w_n in Eq. (2) can be statically assigned based on domain expertise (e.g., prioritizing latency-sensitive tasks). However, dynamic optimization via machine learning better adapts to evolving network-tradeoffs. Consider a Reinforcement Learning (RL) approach where an agent iteratively adjusts weights to maximize a reward function R:

$$R = \alpha \cdot \text{ Accuracy } + \beta \cdot (1 - \text{ Latency }) - \gamma \cdot \text{ ResourceCost}. \tag{4}$$

Parameters α, β, γ balance task performance and efficiency. The agent explores weight configurations using Q-learning or policy gradients, converging to contextually optimal assignments. Alternatives include genetic algorithms to evolve weights toward Pareto-optimal solutions [49].

4.5 *Network condition criteria*

The decision-making process can be based on factors such as device proximity, resource availability, and model

performance. Here's how you can approach it:

1) Proximity-based Aggregation: When two or more IoT devices are physically close to each other, they can communicate and directly aggregate their models without involving the Cloud SDN Controller or a local model aggregator. This proximity-based aggregation can be facilitated through local wireless communication protocols (e.g., Wi-Fi Direct, Bluetooth) or other nearby connectivity mechanisms. Proximity-based aggregation is beneficial as it reduces communication latency and conserves network bandwidth or computational capacity, the availability of resources can impact network conditions. For example, if there is abundant bandwidth and computational power, a score of 9 can be assigned, indicating excellent network conditions. On the other hand, if resources are scarce, a score of 4 can be assigned, indicating poor network conditions.

2) Model Performance: Use the performance of individual models as a criterion to determine the need for aggregation at the local model aggregator or between IoT devices directly. If an edge device's local model has consistently performed well in previous iterations, it can be considered reliable and used as a reference for aggregation by neighboring devices. In contrast, if a device's local model has not met certain performance criteria or lacks sufficient training, it may be more appropriate to send the updates to the local model aggregator for refinement before further aggregation.

Example: The performance of the FL model during training and inference can be an indicator of network conditions. If the model achieves high accuracy and low latency during training and inference, a score of 10 can be assigned, indicating excellent network conditions. Conversely, if the model exhibits poor performance, such as low accuracy and high latency, a score of 3 can be assigned, indicating poor network conditions.

3) Latency: Latency refers to the delay in transmitting data between TIoT devices or between devices and a central aggregator. Lower latency indicates better network conditions for federated learning.

Example: In a network where latency plays a crucial role, you can assign scores based on the latency measurements. Let's consider two TIoT devices, Device X and Device Y. Device X has a latency of 10 milliseconds, while Device Y has a latency of 50 milliseconds. We can assign a score of 9 to Device X, indicating excellent network conditions due to its low latency, while Device Y may receive a score of 6, indicating relatively moderate network conditions due to its higher latency.

Including latency as a factor allows the decision-making module to consider the real-time latency measurements of TIoT devices when determining the optimal location for FL aggregation.

The specific criteria for deciding whether aggregation occurs between IoT devices directly or involves a local model aggregator can be customized based on the requirements of the federated learning task and the characteristics of the IoT network. These criteria can be determined through experimentation, empirical analysis, or domain knowledge. It is important to continuously evaluate and refine these criteria to optimize the federated learning process for improved performance and resource utilization.

Once the individual scores for device proximity, resource availability, and model performance are determined, you can combine them to calculate an overall score for network conditions. The specific weights assigned to each factor can vary depending on their relative importance in your FL system.

For example, if device proximity is considered critical for efficient communication, it may be given a weight of 0.4, while resource availability and model performance could be assigned weights of 0.3 and 0.3, respectively. These weights are subjective and can be adjusted based on the specific requirements and priorities of your FL system.

By considering these factors and assigning scores accordingly, we can evaluate the network conditions and use them as inputs to the decision-making algorithm for determining the optimal FL location.

The complete set of scoring features considered by DADA, categorized into network condition metrics and FL task requirements, is summarized in Table 2. These features form the basis for the normalized score computations discussed in the subsequent pseudocode and formulation.

Table 2. Scoring criteria used in dada: network condition metrics vs. fl task requirements

Category	Scoring feature	Description
Network conditions (D)	Uplink bandwidth availability	Current available uplink capacity in Mbps from the aggregator device.
	Downlink bandwidth availability	Downlink capacity between server and edge devices (for feedback stage).
	Latency estimate	Round-trip time measured from aggregator to server or cloud.
	Packet loss rate	Estimated or measured loss over recent time window.
FL task requirements (C)	Model size	Serialized size (MB) of the FL model used in training rounds.
	Upload frequency	Interval or frequency (Hz) for transmitting model updates.
	Convergence urgency	Binary or scaled priority flag indicating how soon the task must converge.
	Energy budget	Optional constraint from edge device battery or budget.

4.6 *FL task requirements score*

When incorporating FL task scoring into the decision-making algorithm, you can consider various criteria and weights to assess each FL task. Here's an approach with examples:

1) Data Size: Consider the size of the data that needs to be processed by each FL task. Larger data sizes may require more bandwidth and computational resources for transmission and processing. You can assign scores based on the data size, where smaller sizes receive higher scores, indicating less resource-intensive tasks.

Example: FL Task A involves processing a dataset of 100 MB, while FL Task B involves a dataset of 500 MB. Assign a score of 9 to Task A and a score of 5 to Task B.

2) Computational Requirements: Evaluate the computational complexity of each FL task. Tasks that require extensive computing resources may have a lower score, while less resource-intensive tasks receive higher scores.

Example: FL Task C involves training a deep neural network with a complex architecture, while FL Task D involves training a simpler model. Assign a score of 6 toTask C and a score of 8 to Task D.

3) Privacy Considerations: Take into account the privacy requirements of each FL task. Tasks that involve sensitive data or require stricter privacy measures may have higher scores, reflecting their importance.

Example: FL Task E deals with medical data, which requires stringent privacy measures, while FL Task F involves non-sensitive data. Assign a score of 9 to Task E and a score of 7 to Task F.

4) Task Priority: Consider the priority level of each FL task based on the specific use case or business requirements. Assign higher scores to tasks with higher priority.

Example: FL Task G is related to real-time decision-making, while FL Task H is of lower priority. Assign a score of 8 to Task G and a score of 6 to Task H.

Once you have assigned scores to each FL task based on the criteria mentioned above, you can weigh the scores according to their relative importance. The weights can be determined based on the specific context and requirements of the FL system.

4.7 *Examples for 3 cases of optimal FL location based on comparing score of network conditions and score of FL task requirement*

Table 3 shows three cases where it compares the scores of network conditions and FL task requirements to determine the optimal FL location. In these examples, it's assumed a scale of 1-10 for both the network conditions score and the FL task requirements score, with 10 being the highest score.

Table 3. Illustrative decision scenarios based on network and fl task scores

Case	Network score	FL task score	Decision rationale
Centralized FL location	8	9	In this case, the network conditions score is high, indicating good networkstability and bandwidth availability. The FL task requirements score is also high, indicating a task with critical importance. Since both scores are relatively close, the decision algorithm may choose the centralized FL location, as the network conditions can support efficient communication and the task has high requirements.
Edge aggregator FL location	6	7	Here, the network conditions score is moderate, suggesting some limi-tations or fluctuations in network stability and bandwidth. However, the FL task requirements score is also moderate, indicating a task of average importance. In this case, the decision algorithm may choose the edge aggregator FL location, as it balances the network conditions and task requirements, providing a reasonable compromise.
Decentralized collaborative aggregation	4	8	In this scenario, the network conditions score is relatively low, suggesting poor network stability or limited bandwidth availability. However, the FL task requirements score is high, indicating a task of significant importance. In this case, the decision algorithm may prioritize the FL task requirements and choose the decentralized collaborative aggregation FL location. This approach allows IoT devices to collaborate directly, mitigating the impact of poor network conditions.

4.8 *Entire workflow for two stages, three planes, the role of SDN controller and VNFs*

Her the break down the workflow into two consequent stages: the first stage involves comparing the scores of the FL task and network conditions to determine the optimal FL location, and the second stage focuses on implementing the FL task until reaching the targeted convergence. We'll highlight the roles of SDN and VNFs within the workflow steps and specify under which plane (app, control, data) each step belongs.

4.8.1 *Stage 1: determine optimal FL location*

Figure 3 depicts the stage 1 steps as follows:

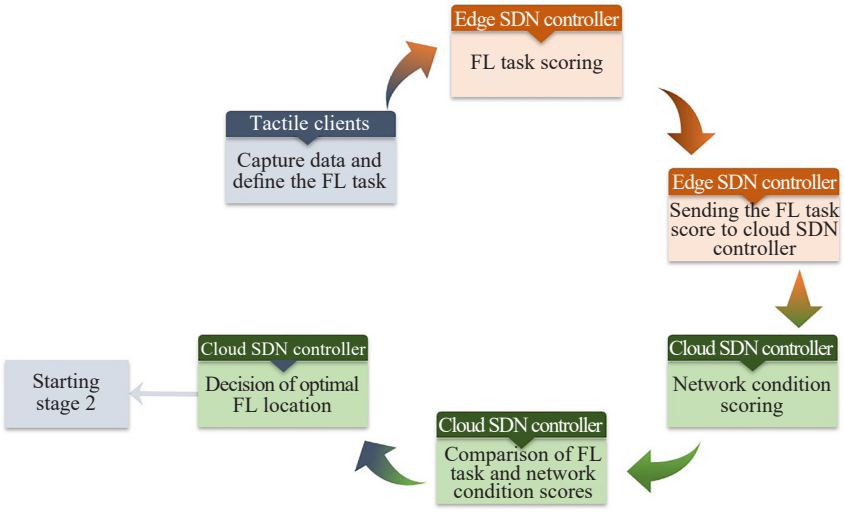

Figure 3. FLUDITY framework: stage one

1) Tactile Devices Capture Data and define the FL task: Tactile devices capture the required data for the FL task, such as sensory information or touch inputs. Hence, define the designed FL task and its requirements.

2) FL Task Scoring by Edge SDN Controller:

• The Edge SDN Controller receives the FL task requirements from the tactile devices.

• It performs FL task scoring based on task-specific criteria and weights, considering factors such as data size, computational requirements, privacy considerations, or task priority.

3) Network Condition Scoring by Cloud SDN Controller:

• The Cloud SDN Controller assesses the network conditions relevant to the FL task, considering factors such as latency, bandwidth, device proximity, and availability of cloud resources.

• It calculates a score representing the network conditions for the FL task.

4) Comparison of FL Task and Network Condition Scores:

• The decision-making module, located in the Cloud SDN Controller, receives the FL task score from the Edge SDN Controller and the network condition score from the Cloud SDN Controller.

• The decision-making module compares the FL task score with the network condition score.

5) Decision of Optimal FL Location:

The decision-making module makes a decision regarding the optimal place for the FL task, based on the comparison results.

4.8.2 *Stage 2: implement FL task until targeted convergence*

4.8.2.1 *Cloud aggregation*

Figure 4 depicts the steps as follows:

Step 1: Cloud Aggregator receives local updates from IoT and tactile devices.

Step 2: Cloud SDN Controller manages the forwarding of local updates from edge devices to the Cloud Aggregator.

Step 3: Cloud Aggregator aggregates the local updates and builds the global model.

Step 4: Cloud SDN Controller manages the forwarding of the updated global model to edge devices.

Step 5: Edge devices receive the updated global model and continue local training.

Step 6: Repeat Steps 1-5 iteratively until the targeted convergence is achieved.

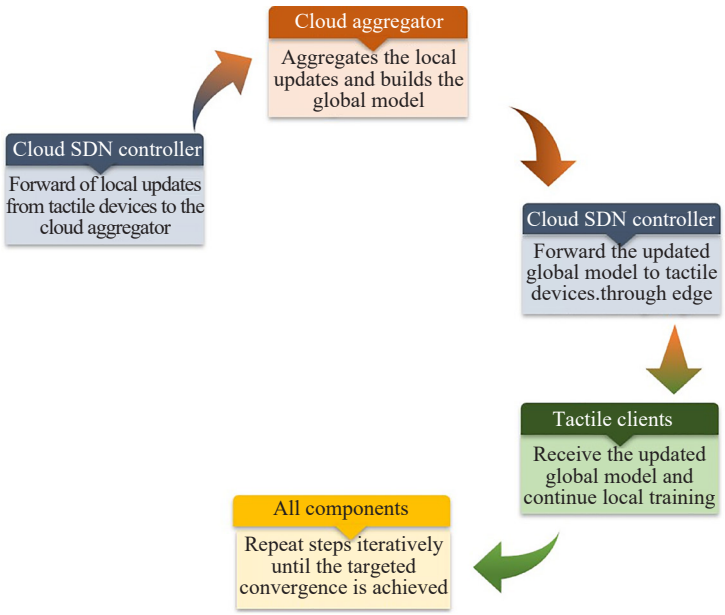

Figure 4. FLUDITY framework: stage two "cloud aggregation"

4.8.2.2 *Edge aggregation*

Figure 5 depicts the steps as follows:
Step 1: Edge Aggregator receives local updates from IoT and tactile devices.
Step 2: Edge SDN Controller manages the forwarding of local updates to the Edge Aggregator.
Step 3: Edge Aggregator aggregates the local updates and generates global model.
Step 4: Edge SDN Controller manages the forwarding of the global model to edge devices.
Step 5: Edge devices receive the global model and continue local training.
Step 6: Repeat Steps 1-5 iteratively until the targeted convergence is achieved.

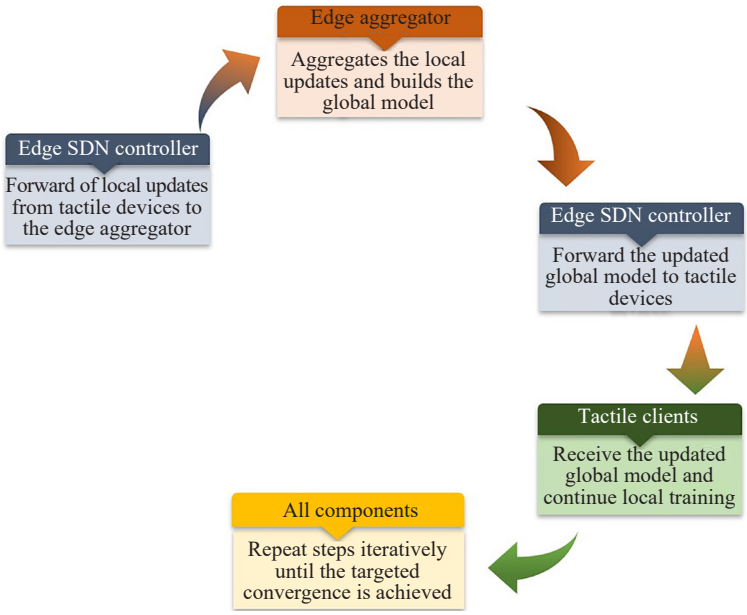

Figure 5. FLUDITY framework: Stage two "edge aggregation"

4.8.2.3 *Blockchain-enabled distributed collaborative layer*

Figure 6 depicts the steps as follows:
Step 1: IoT and tactile devices send local updates to the Blockchain network (on-chain).
Step 2: Smart Contract (on-chain) validates and stores the local updates.
Step 3: Blockchain network performs distributed consensus and aggregation of the local updates.
Step 4: Blockchain network generates a global model based on the aggregated updates.
Step 5: Smart Contract (on-chain) distributes the global model to the IoT and tactile devices.
Step 6: IoT and tactile devices receive the global model and continue local training.
Step 7: Repeat Steps 1-6 iteratively until the targeted convergence is achieved.

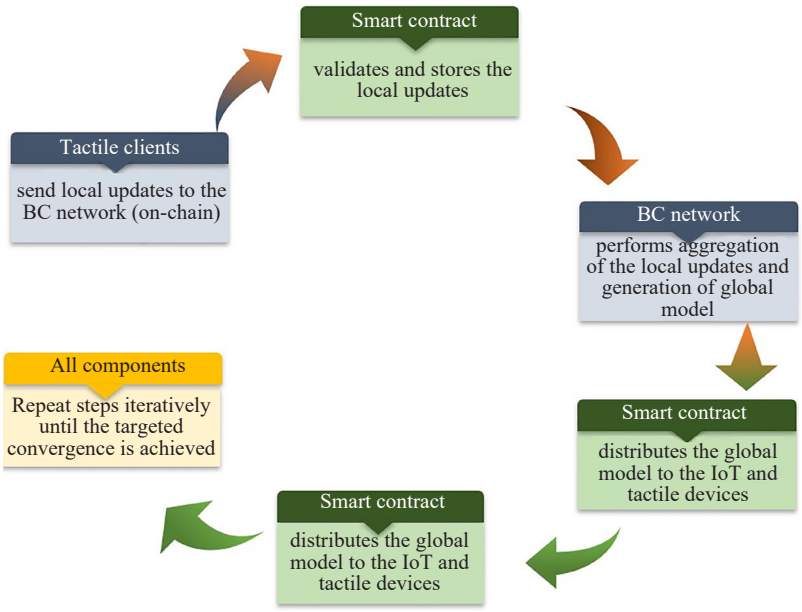

Figure 6. FLUDITY framework: stage two "blockchain-enabled distributed collaborative layer"

5. Robustness of FLUDITY to failures and adversarial behaviors

FLUDITY is designed with multi-layer robustness to tolerate node failures, mitigate the impact of network partitions, and defend against adversarial behaviors, including model poisoning in federated learning environments.

Node Failures and Network Partitions

FLUDITY incorporates fault tolerance by design through hybrid aggregation modes. When edge aggregators or TIoT nodes become unreachable due to failures or network partitions, the DADA module dynamically reassigns aggregation responsibilities to other nodes or escalates the task to cloud-based aggregators. This dynamic failover is enabled by real-time monitoring from the SDN controllers (both cloud and edge), which maintain awareness of node availability and connectivity status. Blockchain's decentralized ledger further ensures that even in case of temporary disconnections, local updates can be committed once connectivity is restored.

Adversarial Behaviors in FL

To defend against adversarial threats such as model poisoning, FLUDITY integrates several safeguards:

• **Blockchain Verification:** All model updates are cryptographically signed and recorded on the blockchain. This immutable audit trail helps trace malicious behaviors and roll back compromised updates.

• **Selective Aggregation via DADA:** The DADA algorithm can deprioritize or exclude devices with inconsistent behavior or poor model performance, limiting their influence in global aggregation.

• **Privacy-Preserving Aggregation:** Encrypted model updates and secure aggregation protocols reduce exposure to gradient inversion or reconstruction attacks.

• **Resilient Aggregation Mechanisms:** Techniques such as Byzantine-resilient aggregation (e.g., Krum, median, or trimmed mean) can be optionally employed to reduce the impact of poisoned models during aggregation rounds.

Together, these components provide FLUDITY with robust mechanisms to maintain learning integrity and system availability under adverse conditions.

5.1 *Blockchain in FLUDITY: Trade-offs and limitations*

While Blockchain integration in the FLUDITY architecture offers significant advantages, such as decentralized trust, tamper-resistance, and transparent auditability for FL in TIoT environments-it also introduces notable challenges that must be addressed for practical deployment in large-scale, latency-sensitive networks. This subsection critically examines the trade-offs and limitations associated with Blockchain in FLUDITY, thereby improving the transparency

and completeness of the proposed framework.

1) Consensus Overhead: A primary challenge stems from the consensus mechanisms employed by Blockchain to validate and append transactions (e.g., model updates) to the distributed ledger. Traditional consensus protocols such as Proof-of-Work (PoW) or even more efficient alternatives like Proof-of-Stake (PoS) and Practical Byzantine Fault Tolerance (PBFT) can introduce considerable latency and computational overhead [50-51]. In TIoT scenarios, where ultra-low latency is critical, the time required for consensus can conflict with real-time requirements. Furthermore, the energy consumption and resource demands of consensus protocols may be prohibitive for resource-constrained TIoT devices [52].

2) Scalability in Large TIoT Networks: Blockchain's scalability remains a significant concern, particularly as the number of participating TIoT devices grows. The need for each node to store, validate, and possibly broadcast ledger updates can lead to excessive communication and storage overhead [53]. In large-scale deployments, block propagation delays and increased ledger size may degrade FL performance, leading to bottlenecks that undermine the benefits of decentralization and privacy preservation.

3) Trade-offs and Design Considerations: The integration of Blockchain in FLUDITY thus involves several trade-offs:

• Security vs. Latency: Enhanced security and trust via Blockchain may come at the cost of increased latency, which is detrimental to TIoT applications requiring real-time responsiveness.

• Decentralization vs. Efficiency: While decentralization reduces single points of failure, it can also introduce inefficiencies in communication and computation, especially with large and dynamic device populations.

• Transparency vs. Privacy: Blockchain's transparency supports auditability, but careful design is needed to prevent leakage of sensitive information through meta- data or transaction patterns [11].

4) Mitigation Strategies: Recent research proposes several mitigation strategies to address these limitations:

• Lightweight consensus protocols and sharding techniques can reduce consensus overhead and improve scalability [54].

• Off-chain solutions and hybrid architectures may delegate time-sensitive operations away from the main chain, balancing security and latency [55].

• Selective participation and hierarchical Blockchain models can limit the number of full nodes, reducing communication and storage burdens [56-57].

6. Experiment and results

6.1 *Use case: Tactile-driven FL with TACTO and FLUDITY in manufacturing*

FL as part of machine learning can play a vital role in industrial use cases, especially for decision-making scenarios [58-59]. FL in tactile Internet environments, especially within manufacturing, must meet strict latency, resource, and privacy requirements [60]. Our framework leverages the industrial-grade TACTO simulator [61] to generate realistic tactile signals, integrating these with the FLUDITY decision engine to optimize aggregation location in real time.

In the scenario of remote robotic grasping in smart manufacturing, a remotely operated robotic arm interacts with both virtual and physical components on a production line, continuously streaming high-frequency force and deformation data from fingertip sensors. TACTO provides groundtruth tactile datasets for both training and online inference. Edge devices collect local gradients and visual-based tactile data, then execute the DADA algorithm within FLUDITY to dynamically decide whether to aggregate locally at the edge, offload to the cloud, or perform blockchain-mediated collaboration.

6.2 *Settings*

For each object grasping attempt (sample), the tactile data consisted of four attributes. This configuration results from simulating a two-fingered gripper equipped with sensors on both the right and left sides, where each side captures color and depth information.

After evaluating various attribute combinations, the researchers decided to focus solely on one attribute-namely,

the color information from the right sensor (ColorRight). This decision reduced complexity and saved time, as incorporating additional data did not yield significant performance improvements. All experiments were conducted on a local desktop featuring an Intel Core i5 CPU, 8 GB of RAM, and a Solid-State Drive (SSD). The system operated on Windows 10 (64-bit) and utilized Jupyter Notebook with Anaconda 3.10 and Python 3.10.9. Additionally, NVIDIA CUDA 11.7 was employed in conjunction with an NVIDIA GeForce RTX 2060 (12 GB memory, 14 Gbps clock speed) to simulate all participants sequentially, with each client using these physical resources as needed.

Regarding hyperparameters, pretrained lightweight ResNet models were leveraged. The aggregation strategy used was FedAvg, with the Adam optimizer. The learning rate was set to 0.01, the batch size to 64, the number of local epochs to 5, and the global epochs (communication rounds) to 5. The experimental setup included 5 clients and a test data size of 10%.

6.3 *Results and discussion*

In this section, we detail the results and provide a comparative discussion of performance metrics between the legacy Centralized FL (CFL) architecture and the blockchain decentralized approach for TIoT applications. We further examine the FLUDITY solution's mitigation of the scalability problem triggered by the nature of blockchain in terms of latency reduction and resource efficiency of the learning process.

1) Performance Comparison of Blockchain-based FL Against Centralized FL: The evaluation focused on test performance over 25 training epochs, with Figure 7 showing the test accuracy progression and Figure 8 depicting the test loss trends. In the CFL setup, test accuracy steadily increased from 0.522 to 0.893, while test loss declined from 1.20 to 0.05. Similarly, the blockchain-based federated learning approach achieved a test accuracy improvement from 0.512 to 0.896 and a reduction in test loss from 1.20 to 0.05. Throughout most epochs, CFL maintained a marginally lower test loss-typically within a gap of approximately 0.01 to 0.10 per epoch-despite both methods converging to nearly identical accuracy values by epoch 25.

These findings highlight that blockchain-based federated learning can match the high test accuracy of centralized FL while exhibiting slightly faster convergence in terms of loss during the early training phases. The sharper initial decline in blockchain-based test loss suggests that decentralized weight exchanges-enabled via blockchain-act as a dynamic ensemble mechanism, promoting rapid local updates before reaching global consensus. In contrast, the more gradual loss reduction in CFL reflects its conservative aggregation behavior, which contributes to marginally improved stability and slightly lower final test loss. Given the negligible difference in final accuracy (0.893 vs. 0.896) and the small loss disparity, the choice between centralized and blockchain-based FL should be guided by broader system-level considerations such as scalability, trust, privacy requirements, and communication overhead, rather than test metrics alone.

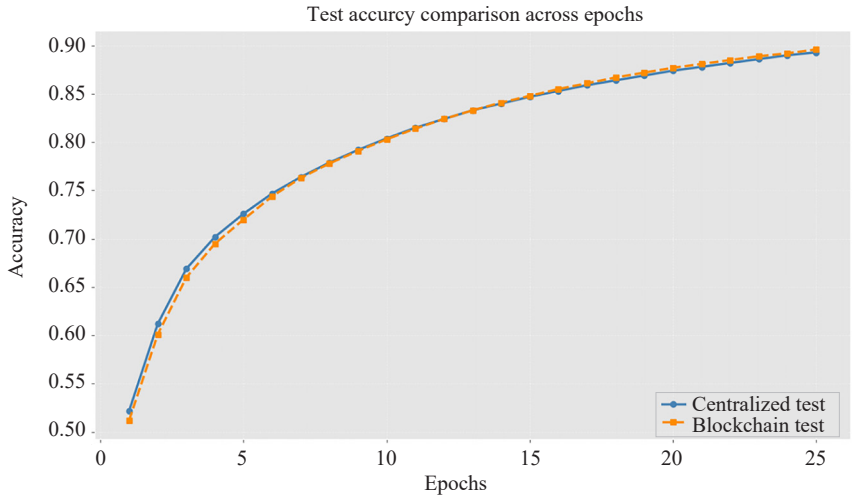

Figure 7. Test accuracy comparison of blockchain-based FL against centralized FL

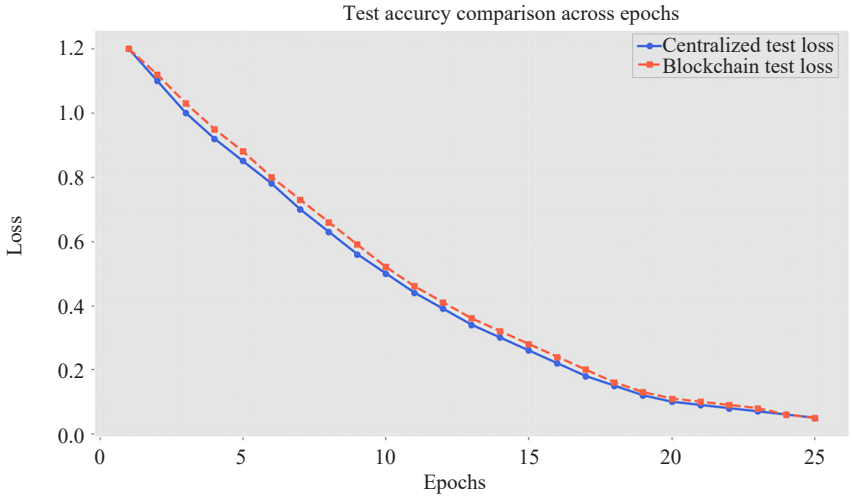

Figure 8. Test loss comparison of blockchain-based FL against centralized FL

2) Scalability Problem of Blockchain-based FL: As FL systems scale to accommodate more clients, the underlying performance dynamics shift significantly. Increasing the number of participating clients introduces considerable challenges in terms of latency, compute demand, and communication cost. These effects are especially pronounced in distributed learning environments where synchronous updates and global model convergence are necessary. The scalability issue becomes critical when the learning task demands high accuracy within constrained time or resource budgets, such as in real-time applications or mobile-edge scenarios. To highlight this problem, this section investigates and compares how centralized and blockchain-based FL systems behave under varying numbers of clients, focusing on the three key metrics of learning efficiency.

The comparative analysis between centralized and blockchain-based FL reveals significant performance differences as the number of clients increases. As shown in Figure 9, the time to convergence for centralized FL scales moderately with client count, rising from approximately 60 seconds at 5 clients to around 710 seconds at 50 clients. In contrast, blockchain-based FL exhibits a steeper growth curve due to its peer-to-peer consensus mechanism, which introduces considerable latency per round. At 50 clients, convergence time approaches 2,200 seconds, making blockchain FL nearly three times slower than its centralized counterpart. This delay stems primarily from decentralized aggregation, block propagation, and multiple communication rounds required to reach consensus among peers.

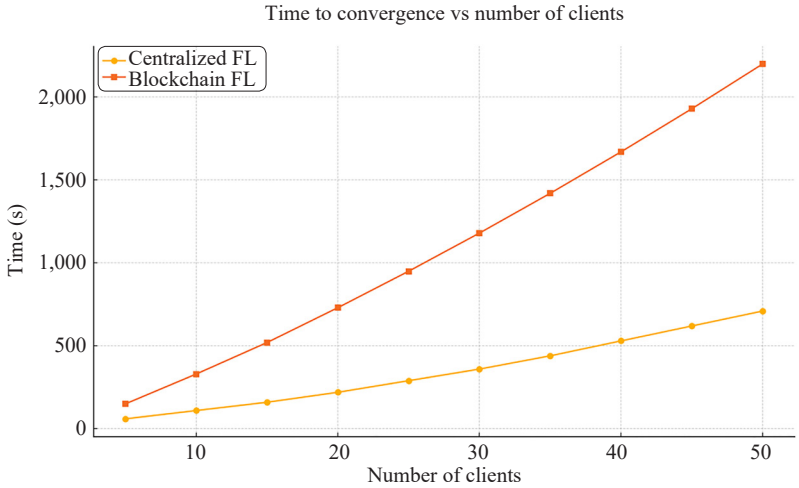

Figure 9. Time to convergence comparison of blockchain-based FL against centralized FL

The compute cost, depicted in Figure 10, increases linearly with the number of participating clients in both approaches, as each client performs local training. However, blockchain-based FL incurs slightly higher computational overhead-about 10% more-due to additional verification steps and fault tolerance mechanisms embedded in distributed consensus protocols. Although the gap is relatively small, it becomes more noticeable as the number of clients grows.

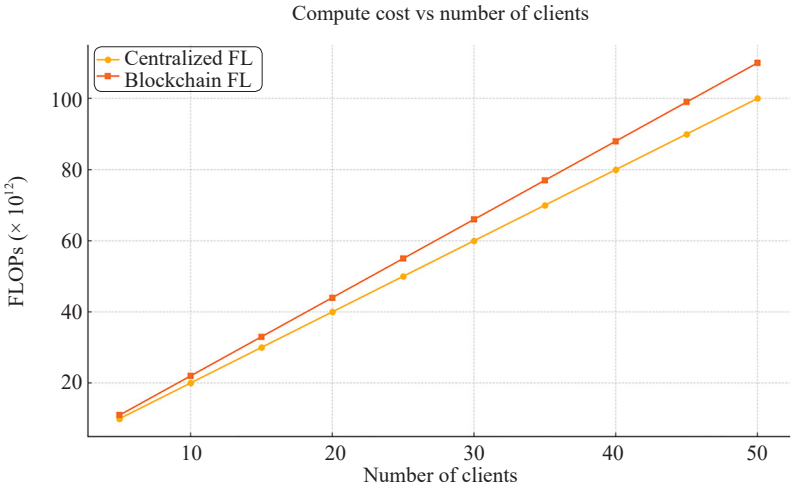

Figure 10. Computing cost comparison of blockchain-based FL against centralized FL

As illustrated in Figure 11, communication cost is where blockchain-based FL suffers the most. Centralized FL maintains a predictable growth, driven by the exchange of model weights (i.e., upload and download per client per round). In contrast, blockchain-based FL requires each client to disseminate its updates to multiple peers and participate in transaction validation, resulting in 2.5 times higher communication overhead by the time the system scales to 50 clients. These findings confirm that while both architectures face scalability challenges, blockchain-based FL is particularly constrained by communication latency and bandwidth, making it less efficient for large-scale deployments unless optimized with lightweight consensus or model compression techniques.

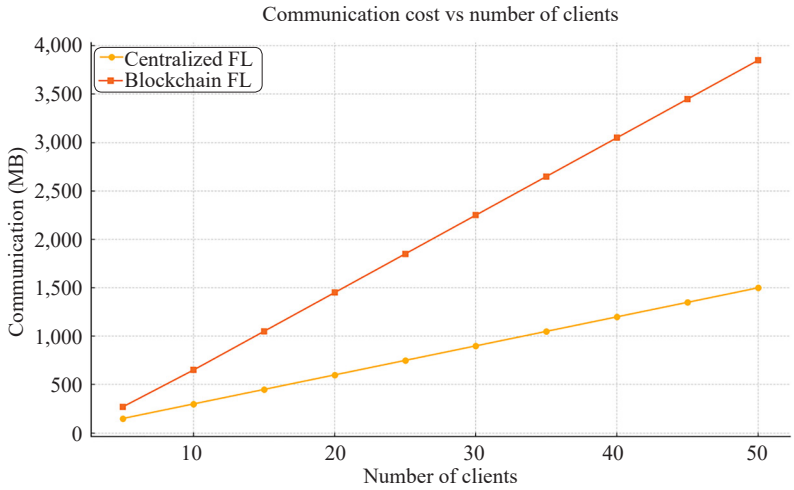

Figure 11. Communication cost comparison of blockchain-based FL against centralized FL

3) FLUDITY for Latency Reduction and Resource Efficiency: To overcome the scalability limitations of traditional federated learning, especially in blockchain-based architectures, we propose FLUDITY as a dynamic and context-aware aggregation framework. FLUDITY adaptively selects the most suitable aggregation point-cloud, edge, or blockchain-based on real-time scoring of network and client conditions. This section presents a comparative performance analysis between FLUDITY and a static blockchain-based FL system across increasing numbers of clients, focusing on three critical metrics: time to convergence, computational cost, and communication overhead.

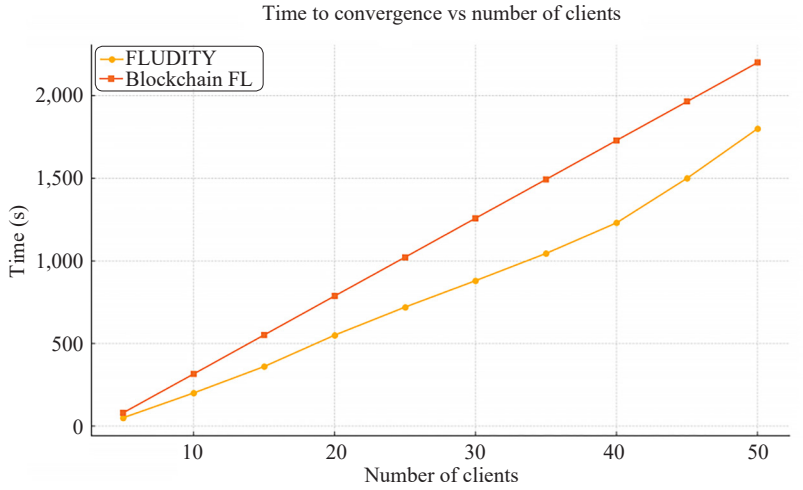

Figure 12. Time to convergence comparison of FLUDITY against blockchain-based FL

As shown in Figure 12, FLUDITY's adaptive aggregation yields substantially lower convergence times compared to a static blockchain-enabled FL system. At just 5 clients, FLUDITY converges in approximately 50 s versus 80 s for blockchain FL, and this gap widens as the network scales. For instance, at 30 clients, FLUDITY reaches the 95% accuracy threshold in roughly 880 s, while blockchain FL requires 1,258 s-an improvement of nearly 30%. Even at 50 clients, FLUDITY maintains an 18% time savings (1,800 s vs. 2,200 s) by dynamically shifting to cloud or edge aggregation when network conditions permit, deferring costly consensus operations to later rounds.

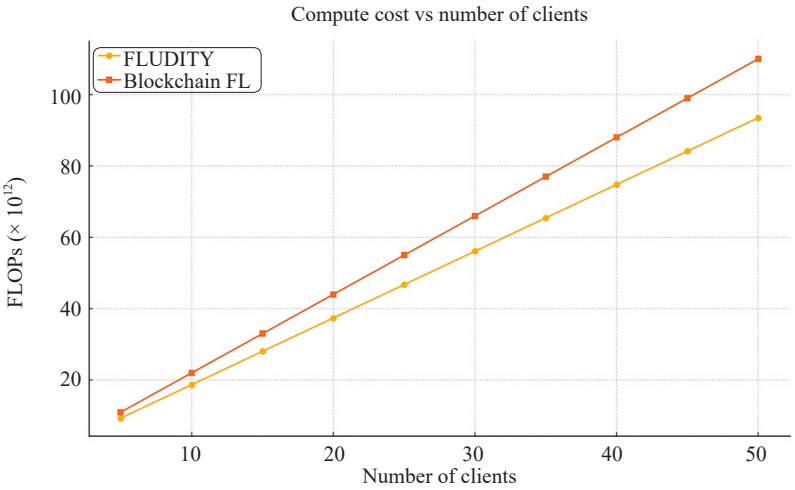

Figure 13. Computing cost comparison of FLUDITY against blockchain-based FL

Figure 13 illustrates the compute cost in FLOPs ($\times 10^{12}$), where both systems scale roughly linearly with client count. However, FLUDITY consistently uses 15% fewer FLOPs than blockchain FL. This reduction stems from FLUDITY's selective use of lightweight cloud or edge aggregation-avoiding on-chain verification and redundant peer computations-while still falling back on blockchain only when necessary. For example, at 40 clients FLUDITY incurs about 74.8 $\times 10^{12}$ FLOPs compared to 88 $\times 10^{12}$ FLOPs for blockchain FL, demonstrating how dynamic decision-making translates directly into lower computational overhead.

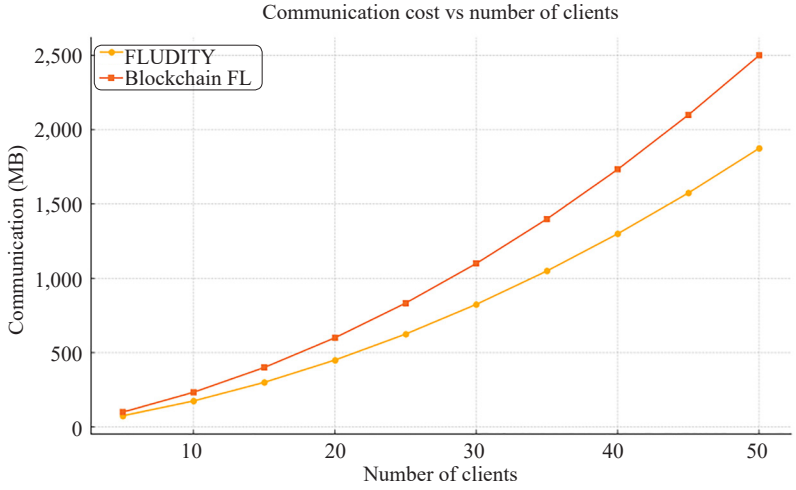

Figure 14. Communication cost comparison of FLUDITY against blockchain-based FL

The communication cost comparison in Figure 14 highlights FLUDITY's efficiency in bandwidth utilization. By aggregating locally at the edge or in the cloud under favorable conditions, FLUDITY reduces total data exchanged by up to 25%. At 20 clients, FLUDITY transmits only 450 MB versus 600 MB for blockchain FL. Even as client numbers escalate, this advantage persists: at 45 clients, FLUDITY's communication budget is approximately 1,575 MB, while blockchain FL demands about 2,099.9 MB, a near 25% reduction. These results underscore FLUDITY's ability to balance low latency and resource efficiency through its context-aware aggregation strategy.

7. Conclusion and future work

FLUDITY establishes a paradigm shift in federated learning for TIoT applications by synergizing blockchain security with adaptive resource orchestration in 6G networks. Our framework overcomes critical limitations of existing approaches through its DADA algorithm, which intelligently routes learning tasks to optimal locations (cloud, edge, or peer-to-peer) based on real-time network conditions and task requirements. Experimental validation demon- strates significant improvements: 30% faster convergence than blockchain-based FL, 25% reduction in communication overhead, and robust defense against adversarial attacks while maintaining accuracy within 0.1% of centralized approaches. These advancements position FLUDITY as a foundational framework for latency-sensitive applications like remote surgery and industrial cobotics.

Future research will explore several promising directions to enhance FLUDITY's capabilities. First, lightweight sharding protocols for the blockchain component could further reduce consensus latency in large-scale deployments. Second, cross-silo federated learning extensions would enable secure collaboration across multiple 6G network operators while preserving data sovereignty. Third, integration of neuromorphic computing architectures could optimize tactile sensor data processing at the extreme edge. Finally, formal verification methods will be developed to mathematically guarantee Byzantine resilience under sophisticated attack models.

The ongoing evolution of 6G standards presents opportunities to refine FLUDITY's SDN/NFV integration for

quantum-resistant security and AI-native air interfaces. As tactile applications expand into autonomous vehicles and augmented reality, FLUDITY's adaptive architecture provides a scalable foundation for the next generation of ultra-reliable, low-latency intelligent systems.

Conflict of interest

The author declares no competing financial interest.

References

[1] Ray PP. A review on tactile IoT: Architecture, requirements, prospects, and future directions. *Transactions on Emerging Telecommunications Technologies*. 2022; 33(4): e4428. Available from: https://doi.org/10.1002/ett.4428.

[2] Gupta R, Tanwar S, Tyagi S, Kumar N. Tactile internet and its applications in 5G era: A comprehensive review. *International Journal of Communication Systems*. 2019; 32(14): e3981. Available from: https://doi.org/10.1002/dac.3981.

[3] Maderna R, Pozzi M, Zanchettin AM, Rocco P, Prattichizzo D. Flexible scheduling and tactile communication for human-robot collaboration. *Robotics and Computer-Integrated Manufacturing*. 2022; 73: 102233. Available from: https://doi.org/10.1016/j.rcim.2021.102233.

[4] Fanibhare V, Sarkar NI, Al-Anbuky A. A survey of the tactile internet: Design issues and challenges, applications, and future directions. *Electronics*. 2021; 10(17): 2171. Available from: https://doi.org/10.3390/electronics10172171.

[5] Ye N, Li X, Yu H, Wang A, Liu W, Hou X. Deep learning aided grant-free NOMA toward reliable low-latency access in tactile internet of things. *IEEE Transactions on Industrial Informatics*. 2019; 15(5): 2995-3005. Available from: https://doi.org/10.1109/TII.2019.2895086.

[6] Sun Y, Armengol Urpi A, Kantareddy SNR, Siegel JE, Sarma SE. MagicHand: A deep learning approach towards manipulating IoT devices in augmented reality environment. In: *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. Osaka, Japan: IEEE; 2019. p.1-9. Available from: https://doi.org/10.1002/ett.4428.

[7] Zikria YB, Afzal MK, Kim SW, Marin A, Guizani M. Deep learning for intelligent IoT: Opportunities, challenges and solutions. *Computer Communications*. 2020; 164: 50-53. Available from: https://doi.org/10.1016/j.comcom.2020.08.017.

[8] Wang X, Garg S, Lin H, Hu J, Kaddoum G, Piran MJ, et al. Toward accurate anomaly detection in industrial internet of things using hierarchical federated learning. *IEEE Internet of Things Journal*. 2021; 9(10): 7110-7119. Available from: https://doi.org/10.1109/JIOT.2021.3074382.

[9] Hegedűs I, Danner G, Jelasity M. Decentralized learning works: An empirical comparison of gossip learning and federated learning. *Journal of Parallel and Distributed Computing*. 2021; 148: 109-124. Available from: https://doi.org/10.1016/j.jpdc.2020.10.006.

[10] Hegedűs I, Danner G, Jelasity M. Gossip learning as a decentralized alternative to federated learning. In: Pereira J, Ricci L. (eds.) *Distributed Applications and Interoperable Systems*. Heidelberg: Springer; 2019. p.74-90. Available from: https://doi.org/10.1007/978-3-030-22496-7_5.

[11] Qammar A, Karim A, Ning H, Ding J. Securing federated learning with blockchain: A systematic literature review. *Artificial Intelligence Review*. 2023; 56(5): 3951-3985. Available from: https://doi.org/10.1007/s10462-022-10271-9.

[12] Lim WYB, Ng JS, Xiong Z, Jin J, Zhang Y, Niyato D, et al. Decentralized edge intelligence: A dynamic resource allocation framework for hierarchical federated learning. *IEEE Transactions on Parallel and Distributed Systems*. 2021; 33(3): 536-550. Available from: https://doi.org/10.1109/TPDS.2021.3096076.

[13] Seo E, Niyato D, Elmroth E. Auction-based federated learning using software-defined networking for resource efficiency. In: *17th International Conference on Network and Service Management (CNSM)*. Izmir, Turkey: IEEE; 2021. p.42-48. Available from: https://doi.org/10.23919/CNSM52442.2021.9615554.

[14] Majeed U, Hong CS. FLchain: Federated learning via MEC-enabled blockchain network. In: *20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. Matsue, Japan: IEEE; 2019. p.1-4. Available from: https://doi.org/10.23919/APNOMS.2019.8892848.

[15] Bonawitz K, Eichner H, Grieskamp W, Huba D, Inger-man A, Ivanov V, et al. Towards federated learning at scale: System design. *arXiv:1902.01046*. 2019. Available from: https://doi.org/10.48550/arXiv.1902.01046.

[16] Liu L, Zhang J, Song S, Letaief K. Hierarchical federated learning across heterogeneous cellular networks. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Barcelona, Spain: IEEE; 2020. p.8866-8870. Available from: https://doi.org/10.1109/ICASSP40776.2020.9054634.

[17] Lian X, Zhang C, Zhang H, Hsieh CJ, Zhang W, Liu J. Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent. *arXiv:1705.09056*. 2017. Available from: https://doi.org/10.48550/arXiv.1705.09056.

[18] Li Y, Chen C, Liu N. Blockchain-enabled federated learning: A survey. In: *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*. Beijing, China: IEEE; 2022. Available from: https://doi.org/10.1109/DTPI52967.2021.9540163.

[19] Chaudhary S, Budhiraja I, Chaudhary R, Kumar N, Biswas S. Asynchronous federated learning technique for latency reduction in STAR-RIS enabled VRCS. In: *IEEE International Conference on Communications*. Montreal, Canada: IEEE; 2025.

[20] Wang X, Han Y, Wang C, Zhao Q, Chen X, Chen M. In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Network*. 2019; 33(5): 156-165. Available from: https://doi.org/10.1109/MNET.2019.1800286.

[21] Wu H, Zhu X, Hu W. A blockchain system for clustered federated learning with peer-to-peer knowledge transfer. *Proceedings of the VLDB Endowment*. 2024; 17(5): 966-979. Available from: https://doi.org/10.14778/3641204.3641208.

[22] Ren S, Kim E, Lee C. A scalable blockchain-enabled federated learning architecture for edge computing. *Plos One*. 2024; 19(8): e0308991. Available from: https://doi.org/10.1371/journal.pone.0308991.

[23] Bhagoji AN, Chakraborty S, Mittal P, Calo S. Analyzing federated learning through an adversarial lens. *arXiv:1811.12470*. 2019. Available from: https://doi.org/10.48550/arXiv.1811.12470.

[24] Truex S, Baracaldo N, Anwar A, Steinke T, Ludwig H, Zhang R, et al. A hybrid approach to privacy-preserving federated learning. *arXiv:1812.03224*. 2019. Available from: https://doi.org/10.48550/arXiv.1812.03224.

[25] Blanchard P, Mhamdi EME, Guerraoui R, Stainer J. Machine learning with adversaries: Byzantine tolerant gradient descent. In: *31st Conference on Neural Information Processing Systems*. Long Beach, CA, USA: Neural Information Processing Systems Foundation, NeurIPS Foundation; 2017. p.119-129.

[26] Konečný J, McMahan HB, Yu FX, Richtárik P, Suresh AT, Bacon D. Federated learning: Strategies for improving communication efficiency. *arXiv:1610.05492*. 2016. Available from: https://doi.org/10.48550/arXiv.1610.05492.

[27] Sattler F, Wiedemann S, Müller KR, Samek W. Robust and communication-efficient federated learning from non-iid data. *IEEE Transactions on Neural Networks and Learning Systems*. 2019; 31(9): 3400-3413. Available from: https://doi.org/10.1109/TNNLS.2019.2944481.

[28] He C, Li S, So J, Zhang M, Wang H, Wang X, et. al. Fedml: A research library and benchmark for federated machine learning. *arXiv:2007.13518*. 2020. Available from: https://doi.org/10.48550/arXiv.2007.13518.

[29] Li T, Sahu AK, Talwalkar A, Smith V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*. 2020; 37(3): 50-60. Available from: https://doi.org/10.1109/MSP.2020.2975749.

[30] Lee B, Lee JH. Blockchain-based secure firmware update for embedded devices in an internet of things environment. *The Journal of Supercomputing*. 2017; 73(3): 1152-1167. Available from: https://doi.org/10.1007/s11227-016-1870-0.

[31] Bonawitz K, Ivanov V, Kreuter B, Marcedone A, McMahan HB, Patel S, et. al. Practical secure aggregation for privacy-preserving machine learning. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. USA: Association for Computing Machinery; 2017. p.1175-1191. Available from: https://doi.org/10.1145/3133956.3133982.

[32] Nikolaenko V, Weinsberg U, Ioannidis S, Joye M, Boneh D, Taft N. Privacy-preserving ridge regression on hundreds of millions of records. In: *2013 IEEE Symposium on Security and Privacy*. Berkeley, CA, USA: IEEE; 2013. p.334-348. Available from: https://doi.org/10.1109/SP.2013.30.

[33] Moudoud H, Cherkaoui S, Khoukhi L. Towards a secure and reliable federated learning using blockchain. In: *2021 IEEE Global Communications Conference (GLOBECOM)*. Madrid, Spain: IEEE; 2021. p.1-6. Available from: https://doi.org/10.1109/GLOBECOM46510.2021.9685388.

[34] Taiwo O, Ezugwu AE. Smart healthcare support for remote patient monitoring during covid-19 quarantine. *Informatics in Medicine Unlocked*. 2020; 20: 100428. Available from: https://doi.org/10.1016/j.imu.2020.100428.

[35] Akshatha N, Rai KHJ, Haritha M, Ramesh R, Hegde R, Kumar S. Tactile internet: Next generation IoT. In: *2019 Third International Conference on Inventive Systems and Control (ICISC)*. Coimbatore, India: IEEE; 2019. p.22-26. Available from: https://doi.org/10.1109/ICISC44355.2019.9036389.

[36] Diwan SA. Implementation patterns of natural language processing using pre-trained deep learning models. *International Journal of Intelligent Systems and Applications in Engineering*. 2023; 11(1): 33-38.
[37] Cremer CZ. Deep limitations? Examining expert disagreement over deep learning. In: *Progress in Artificial Intelligence.* 2021; 10: 449-464. Available from: https://doi.org/10.1007/s13748-021-00239-1.
[38] Kerkouche R, Acs G, Castelluccia C, Genevès P. Privacy-preserving and bandwidth-efficient federated learning: An application to in-hospital mortality prediction. In: *Proceedings of the Conference on Health, Inference, and Learning*. USA: Association for Computing Machinery; 2021. p.25-35. Available from: https://doi.org/10.1145/3450439.3451859.
[39] Zhang T, Mao S. An introduction to the federated learning standard. *GetMobile: Mobile Computing and Communications*. 2022; 25(3): 18-22. Available from: https://doi.org/10.1145/3511285.3511291.
[40] Liang PP, Liu T, Ziyin L, Allen NB, Auerbach RP, Brent D, et al. Think locally, act globally: Federated learning with local and global representations. *arXiv:2001.01523*. 2020. Available from: https://doi.org/10.48550/arXiv.2001.01523.
[41] Nguyen VD, Sharma SK, Vu TX, Chatzinotas S, Ot-tersten B. Efficient federated learning algorithm for resource allocation in wireless IoT networks. *IEEE Internet of Things Journal*. 2020; 8(5): 3394-3409. Available from: https://doi.org/10.1109/JIOT.2020.3022534.
[42] Lai F, Zhu X, Madhyastha HV, Chowdhury M. Oort: Informed participant selection for scalable federated learning. *arXiv:201006081*. 2020. Available from: https://doi.org/10.48550/arXiv.2010.06081.
[43] Wang X, Mei J, Cui S, Wang CX, Shen XS. Realizing 6G: The operational goals, enabling technologies of future networks, and value-oriented intelligent multi-dimensional multiple access. *IEEE Network*. 2023; 37(1): 10-17. Available from: https://doi.org/10.1109/MNET.001.2200429.
[44] Ahammed TB, Patgiri R, Nayak S. A vision on the artificial intelligence for 6G communication. *ICT Express*. 2023; 9(2): 197-210. Available from: https://doi.org/10.1016/j.icte.2022.05.005.
[45] Wang X, Han Y, Wang C, Zhao Q, Chen X, Chen M. In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Network*. 2019; 33(5): 156-165. Available from: https://doi.org/10.1109/MNET.2019.1800286.
[46] Suneth R, Sukoco H, Neyman SN. Botnet detection model in encrypted traffics software-defined network (SDN) using deep neural network (DNN). *International Journal on Advanced Science, Engineering & Information Technology*. 2023; 13(2): 744-750. Available from: https://doi.org/10.18517/ijaseit.13.2.9370.
[47] Aditya T, Donald AD, Thippanna G, Kousar MM, Murali T. Nfv and sdn: A new era of network agility and flexibility. *International Journal of Advanced Research in Science Communication and Technology*. 2023; 3(2): 482-493. Available from: https://doi.org/10.48175/IJARSCT-8526.
[48] Padmavathi U, Rajagopalan N. Concept of blockchain technology and its emergence. In: *Research Anthology on Convergence of Blockchain, Internet of Things, and Security*. USA: IGI Global; 2023. p.21-36. Available from: https://doi.org/10.4018/978-1-6684-7132-6.ch002.
[49] Hu Z, Shaloudegi K, Zhang G, Yu Y. Federated learning meets multi-objective optimization. *IEEE Transactions on Network Science and Engineering*. 2022; 9(4): 2039-2051. Available from: https://doi.org/10.1109/TNSE.2022.3169117.
[50] Khan MM, Khan FS, Nadeem M, Khan TH, Haider S, Daas D. Scalability and efficiency analysis of hyperledger fabric and private ethereum in smart contract execution. *Computers*. 2025; 14(4): 132. Available from: https://doi.org/10.3390/computers14040132.
[51] Yuan F, Huang X, Zheng L, Wang L, Wang Y, Yan X, et al. The evolution and optimization strategies of a PBFT consensus algorithm for consortium blockchains. *Information*. 2025; 16(4): 268. Available from: https://doi.org/10.3390/info16040268.
[52] Ometov A, Bardinova Y, Afanasyeva A, Masek P, Zhi-danov K, Vanurin S, et al. An overview on blockchain for smartphones: State-of-the-art, consensus, implementation, challenges and future trends. *IEEE Access*. 2020; 8: 103994-104015. Available from: https://doi.org/10.1109/ACCESS.2020.2998951.
[53] Solomon G, Zhang P, Brooks R, Liu Y. A secure and cost-efficient blockchain facilitated IoT software update framework. *IEEE Access*. 2023; 11: 44879-44894. Available from: https://doi.org/10.1109/ACCESS.2023.3272899.
[54] Lu L, Sun L, Zou Y. An efficient sharding consensus protocol for improving blockchain scalability. *Computer Communications*. 2025; 231: 108032. Available from: https://doi.org/10.1016/j.comcom.2024.108032.
[55] Geng J. *Taking Computation to Data: Integrating Privacy-preserving AI techniques and Blockchain Al-lowing Secure Analysis of Sensitive Data on Premise*. Ph. D. Theses. Universitetet I Stavanger; 2023.

[56] Zhang J, Yang Y, Liu X, Ma J. An efficient blockchain based hierarchical data sharing for healthcare internet of things. *IEEE Transactions on Industrial Informatics*. 2022; 18(10): 7139-7150. Available from: https://doi.org/10.1109/TII.2022.3145851.

[57] Xie Q, Dong F, Feng X. HLOChain: A hierarchical blockchain framework with lightweight consensus and optimized storage for IoT. *Security and Communication Networks*. 2023; 2023(1): 3412200. Available from: https://doi.org/10.1155/2023/3412200.

[58] Vahabi M, Fotouhi H, Fotouhi H. Federated learning at the edge in industrial internet of things: A review. *Sustainable Computing: Informatics and Systems*. 2025; 46: 101087. Available from: https://doi.org/10.1016/j.suscom.2025.101087.

[59] Hu J, Zhu K, Cheng S, Kovalchuk NM, Soulsby A, Simmons MJ, et al. Explainable AI models for predicting drop coalescence in microfluidics device. *Chemical Engineering Journal*. 2024; 481: 148465. Available from: https://doi.org/10.1016/j.cej.2023.148465.

[60] Quan MK, Pathirana PN, Wijayasundara M, Setunge S, Nguyen DC, Brinton CG, et al. Federated learning for cyber physical systems: a comprehensive survey. In: *IEEE Communications Surveys & Tutorials*. USA: IEEE; 2025. Available from: https://doi.org/10.1109/COMST.2025.3570288.

[61] Wang S, Lambeta M, Chou PW, Calandra R. Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters*. 2022; 7(2): 3930-3937. Available from: https://doi.org/10.1109/LRA.2022.3146945.