Journal of Scientific & Industrial Research
Vol. 85, April 2026, pp. 341-350
DOI: 10.56042/jsir.v85i4.8384

Trust-based Energy Aware Secure Load Balancing and Resource Provisioning in Fog Computing using a Multi-Objective Optimization Algorithm

Ruchi Agrawal[1]*, Saurabh Singhal[2] & Ashish Sharma[1]

[1]GLA University, Mathura, Chaumuhan, Uttar Pradesh 281 406, India

[2]Department of Computer Science and Engineering,
Greater Noida Institute of Technology (Engineering Institute), Greater Noida, Uttar Pradesh 201 306, India

Received 29 January 2024; revised 24 April 2024; accepted 05 January 2026

Fog Computing (FC) is gradually essential in diminishing communication latency and enhancing resource usage for Internet of Things (IoT) tasks; though, critical experiments like resource overloading, security weaknesses, and excessive energy consumption frequently hinder its operational potential. The range of this study includes the expansion of a trust-based, energy-aware secure load matching and resource provisioning system, definitely calculated for decentralized fog architectures using a Multi-Objective Optimization Algorithm (MOA). The methodology incorporates a Many-to-Few (M2F) balancer for effective task distribution, ordering trust in node-task assignments to certify consistency. Resource management is significantly improved through the hybridization of Improved Salp Swarm Optimization (ISSO) and Modified Whale Optimization Algorithm (MWOA) for dynamic provision. To bolster system integrity, an intrusion detection system with offloading mechanisms is implemented alongside Hierarchical Collaborative Federated Learning (HCFL) to raise secure, privacy-preserving node association. Key results from performance assessments showed in the Matrix Laboratory (MATLAB) establish a high 88% Average Resource Utilization (ARU) and a steady 1300s response time. The model's strength is further supported by a precision of 99.5%, an F-measure of 91.0%, and a recall of 78.6% during oppositional testing. The individuality of this study lies in its concurrent optimization of trust, energy, and load metrics within a single meta-heuristic framework. This system offers a practical value for mission-critical IoT ecosystems, such as smart grids and industrial automation, where real-time handling and data reliability are mandatory for keeping complete system strength and performance.

Keywords: Decentralized systems, Energy efficiency, Federated learning, Meta-heuristics, Resource management

Introduction

Fog computing enables computation at the edge of the network, thereby supporting emerging applications and services in the evolution of the Internet of Things (IoT).[1] It was introduced to reduce latency and enhance user experience in IoT-enabled environments.[2] Fog computing is a decentralized paradigm where storage, processing, and applications are distributed between data sources and the cloud, thereby improving system responsiveness.[3] It also provides significant benefits in terms of privacy, security, bandwidth utilization, and latency reduction, addressing several limitations of traditional cloud computing systems.[4] Secure load balancing plays a crucial role in fog computing environments by ensuring that fog nodes are neither overburdened nor underutilized.[5] The primary objective of load balancing is to distribute workloads efficiently across available resources to improve system performance and reliability.[6] In addition, load balancing techniques enhance multiple Quality of Service (QoS) parameters such as response time, throughput, energy consumption, and overall system efficiency.[7] An effective workload distribution framework ensures optimal utilization of computing resources and improves system scalability.[8] Balanced systems are characterized by equal workload distribution, efficient resource utilization, improved server performance, reduced energy consumption, faster response times, and enhanced user satisfaction.[9] However, fog computing environments face significant challenges in resource management due to the heterogeneity of devices, dynamic workloads, and stringent performance requirements.[10] These challenges motivate the need for advanced optimization-based approaches. To address these issues, this study proposes a trust-based, energy-aware, secure load balancing and

*Author for Correspondence
E-mail: ruchi.agrawal@gla.ac.in

resource provisioning framework in fog computing using a Multi-Objective Optimization (MOO) algorithm to enhance system scalability, efficiency, and reliability.[11] Load balancing techniques are widely used to distribute user requests efficiently across multiple servers, ensuring improved system flexibility.[12] A load balancer maintains an updated list of available servers, monitors resource availability, and distributes workloads accordingly.[13] It evaluates incoming tasks and assigns them to computing resources based on system capacity and performance metrics.[14]

The rapid expansion of internet usage has significantly influenced nearly all aspects of daily life, making users more dependent on intelligent and automated systems.[15] Users increasingly prefer systems that are simple, cost-effective, and efficient, which has driven the adoption of smart computing paradigms.[16] Although fog computing is still in its early deployment stage, its conceptual foundations are well established.[10] However, practical implementations of fog-based solutions remain limited.[17] Moreover, scheduling and resource provisioning continue to be key open research challenges in fog computing environments.[18] One of the major challenges is designing scalable fog architectures capable of supporting millions of heterogeneous devices while maintaining low latency, efficient bandwidth utilization, and wide geographic coverage.[19] Additionally, modern computing paradigms such as microservices and container-based architectures are transforming software deployment and scalability models.[20,21] The proposed trust-based energy-aware secure load balancing approach aims to improve resource utilization, system stability, and computational efficiency in fog computing environments.[22] Furthermore, the integration of a Multi-Objective Optimization algorithm enhances key performance metrics such as response time, energy consumption, and overall system efficiency.[23] The primary objective of this study is to design and implement a trust-based, energy-aware, secure load balancing and resource provisioning framework for fog computing environments using a Multi-Objective Optimization algorithm.[24] This work bridges the gap by jointly optimizing trust, energy efficiency, and load balancing using a multi-objective optimization framework in fog computing environments.

Literature Survey

Recent advancements in cloud, fog, and edge computing have necessitated efficient and intelligent task scheduling, secure resource allocation, and adaptive load balancing mechanisms. To address these challenges, several research efforts have focused on optimization techniques, artificial intelligence models, and trust-aware frameworks. A Comparative analysis of recent approaches and identified gaps in fog and cloud computing is presented in Table 1. Mangalampalli *et al*. proposed a multi-objective prioritized workflow scheduling approach using deep reinforcement learning to optimize execution time and resource utilization.[25] Saini *et al*. introduced a trust-based service mechanism incorporating k-anonymity and statistical machine learning to enhance cloud security and privacy.[26] Sharma *et al*. developed an AI-driven latency prediction model integrated with federated and reinforcement learning for QoS-aware task scheduling in mobile edge computing.[27] Similarly, Krishna and Mangalampalli proposed a fault-tolerant task scheduling model using deep reinforcement learning to improve system reliability.[28] Nursuwars *et al*. reviewed IoT-fog data transmission techniques and highlighted adaptive mechanisms such as dynamic frequency control.[29] Mangalampalli *et al*. further proposed an efficient deep reinforcement learning-based scheduler for multi-cloud environments, improving scalability and adaptability.[30] Jorquera Valero *et al*. presented a comprehensive analysis of trust management in 5G and beyond, emphasizing trust evaluation requirements and challenges.[31] Sherriff and Lim proposed heuristic-based recovery algorithms for mobile edge computing using relay nodes to enhance fault tolerance.[32] Manogaran *et al*. introduced a double deep Q-learning-based routing strategy combined with federated learning to improve routing efficiency in IoT environments.[33] Alalwany and Mahgoub discussed security and trust management challenges in the Internet of Vehicles using machine learning techniques.[34] Despite these advancements, most studies focus on individual aspects such as scheduling, trust, or energy efficiency. There is a lack of integrated frameworks that simultaneously address trust, energy awareness, and load balancing using multi-objective optimization in fog computing environments.

Proposed Research Methodology

The proposed research methodology aims to address the difficult problems of trust-based, energy-aware, safe load balancing, and resource provisioning in fog computing environments using a multi-objective optimization process. This methodology incorporates several components to certify efficient

Table 1 — Comparative analysis of recent approaches and identified gaps in fog and cloud computing

Approach (Ref.)	Technique	Key Focus	Findings	Limitations / Research Gap
Load balancing survey[1]	SLR	Fog load balancing	Consolidates LB techniques; identifies 60+ methods	No unified optimization model
Trust evaluation[2]	Fuzzy trust model	6G trust management	~15–25% improvement in trust accuracy	No scheduling integration
Energy-aware trust[3]	Optimization	Energy + trust	10–20% energy efficiency gain	Not fog-specific
Trust scheduling[4]	Harris Hawks optimization	Fault-tolerant scheduling	↑ QoS ~18–30%	No energy awareness
Secure fog IoT[5]	Security framework	IoT security	Strong attack mitigation (high)	No load balancing
Trust ML model[6]	Hybrid ML	Privacy + trust	↑ security accuracy ~20%	No resource scheduling
Resource mgmt[7]	Taxonomy	Energy/service mgmt	Theoretical classification only	No real implementation
AI load balancing[8]	Graph neural model	Load balancing	↑ LB efficiency ~25–35%	No trust/security
QoS scheduling[9]	Whale optimization	Resource scheduling	↑ QoS ~20–28%	No trust-awareness
Cloud optimization[10]	Evaluation model	Cloud performance	↑ system efficiency ~15%	Not fog-based
Blockchain LB[11]	Distributed LB	Healthcare IoT	↑ reliability ~20%	Scalability issues
Federated scheduling[12]	FL-based scheduling	Fault tolerance	↑ accuracy ~22%	No trust-energy link
Fog-cloud survey[13]	Unified computing	Integration model	Conceptual roadmap only	No algorithm
Edge-cloud mgmt[14]	Survey	Resource mgmt	Identifies trends	No implementation
Service composition[15]	Comparative analysis	IoT services	Moderate efficiency gain	No scheduling
SDN security[16]	Survey	IoT security	High security coverage	No LB optimization
Fault scheduling[17]	Cluster-based model	QoS scheduling	↑ performance ~25%	No trust-energy
Federated LB[18]	FL clustering	Load balancing	↑ LB efficiency ~30%	Limited security
Fog offloading[19]	Review	Task offloading	Summarizes methods	No quantitative model
Metaheuristic LB[20]	Comparative study	LB algorithms	Best improvement ~20–40% (varies)	No unified framework
Trust offloading[21]	Trust detection	Edge offloading	↑ trust detection ~18%	No energy focus
ML edge computing[22]	ML model	Network optimization	↑ performance ~20%	No scheduling integration
Energy fairness[23]	Optimization	Energy efficiency	↑ energy fairness ~15–25%	No trust/security
SDN routing[24]	Survey	Security + routing	Theoretical insights	No optimization model
DRL scheduling[25]	DRL optimization	Workflow scheduling	↑ efficiency ~30%	No trust/security
Trust ML security[26]	k-anonymity ML	Cloud security	↑ privacy ~25%	No LB integration
AI QoS scheduling[27]	RL + FL	Latency scheduling	↓ latency ~20–30%	No energy-trust
DRL fault tolerance[28]	DRL scheduler	Reliability	↑ fault tolerance ~25%	No multi-objective
IoT-fog adaptation[29]	SLR	Data transmission	Theoretical improvements	No optimization model
Multi-cloud DRL[30]	DRL scheduler	Scalability	↑ scalability ~30%	No trust-awareness
5G trust survey[31]	Survey	Trust systems	Conceptual framework	No implementation
MEC heuristic[32]	Heuristic recovery	Fault recovery	↑ recovery speed ~15–20%	No LB/energy
DRL routing[33]	Q-learning + FL	IoT routing	↑ routing efficiency ~25%	No resource mgmt
IoV trust ML[34]	ML trust model	Vehicle trust	↑ detection accuracy ~20%	No load balancing

Note: Performance improvements represent comparative trends reported in the surveyed literature

and consistent operation in fog computing systems. Firstly, it includes the creation of a trust-based framework to assess the consistency and reliability of fog nodes, improving the security and dependability of the system. Secondly, energy-aware mechanisms are applied to improve resource utilization while lessening energy consumption, certifying sustainability and cost-effectiveness. Load balancing methods are then working to allocate computational tasks proficiently across fog nodes, stopping overloads and blockages to preserve system performance. The flow diagram for the recommended task is presented in Fig. 1.

The assets in fog environments are stretchy; however, frequently obstructed by unexpected factors such as changing network capability owing to weather, communication obstructions, and end-device flexibility. To raise the full utilisation of these resources, this study offers a safe, energy-aware fog computing architecture including a secure load-balancing and provisioning method. The research primarily proposes a load-balancing method termed the Many-to-Few (M2F) balancer, which endlessly distributes network traffic and similarly allocates incoming queries among available servers using vigorous resource distribution. As a result, a fog node's workload fluctuates frequently, making it vital to develop an algorithm that can vigorously update network features.

Therefore, a multi-objective optimisation problem is practical in this work to design the ideal fog node

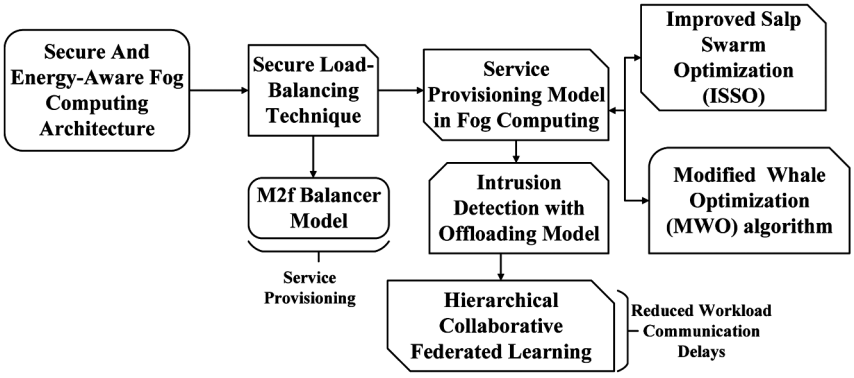

Fig. 1 — Flow Diagram of the Proposed Work

selection. An effective selection method based on the Modified Whale Optimisation Algorithm (MWOA) and Improved Salp Swarm Optimisation (ISSO) is combined to address this issue. Addressing the level of reliability in fog technology is an important concern that is addressed here by applying a Trust Management (TM) system. This is a multi-criteria management challenge since metrics, for example, Quality of Service (QoS), social relationships, and reputation repeatedly reverse one another. The proposed load-sharing plan addresses security by generating a trust-based system for load-offloading. Also, a hierarchical collaborative federated learning-based intrusion detection system has been established to reduce communication postponements and increase capability in the fog layer.

Secure Load Balancing in Fog Computing

Time-sensitive data processing and local data storage are enabled by fog computing. This architecture diminishes the volume and distance of data sent to the Cloud Layer (CL), which in turn decreases the influence of Internet of Things (IoT) security and privacy problems. To certify data integrity, it is vital to build protected load-balancing systems and private depositing, along with safe, proven data possession measures. To overcome these problems, this study exploits a hybrid technique of the Fog-IoT Load Balancing (LB) resource allocation method to extend the network load suitably. In light of this, the research recommends an LB method termed the M2F balancer, which endlessly accomplishes network traffic, collects data on each server's load, and equally allocates incoming queries among available servers through dynamic resource provision. By providing nonstop service, the M2F balancer is a valuable method for creating the most use of assets.

M2F Balancer Model

The M2F balancer system, which consists of four layers: the IoT layer, the Mist layer, the Fog Layer (FL), and the CL, is the focus of this segment. The IoT layer is responsible for gathering pulses from patients between these four levels. The third layer, identified as the fog layer, is applied to regulate the server's state and to change work from one overcrowded server to a new one. The storage and safe transmission of data to and from the mist layer are controlled by the fourth layer, the CL. To the most suitable server, the needed data is transmitted. The M2F balancer assigns the optimized scheduling method and gives priority to incoming requests from various locations to handle them. Consequently, the M2F balancer utilizes the minimum quantity of execution period and waiting period. Information flows across many levels, from the device layer to the CL. Internet influences from the device layer to the CL are a fundamental necessity. The user transmits the request from the device layer, and the subsequent layer provides an acknowledgement. If the mist server is congested, the mission is transmitted to the fog server after the user submits it to the mist server. The task is passed to the CL if the fog server is unable to complete it. Based on the aforementioned parameters, this Adaptive Weight (AW) is indicated as follows:

$$AW = \beta \times (Capacity \times \frac{Storage}{CPU\ usage}) \quad \ldots (1)$$

where, β is the adaptive weight factor, $Capacity$ denotes the node power, $Storage$ is the available RAM, and $CPU\ usage$ represents the current workload. The M2F balancer is in charge of selecting the precise server and handling incoming requests. Two criteria, the qualities of the mist server and predetermined conditions, are exploited to pick the

mist server. The Mist Server Classification (MSC) and Calculating Weight Procedure (CWP) are the essential parts of the M2F balancer. Every cluster has a chief node that controls and maintains the data for the entire server. Information from the slave nodes is sent to the master node by the Mist Server Manager (MSM). The Server Information Table (SIT) preserves track of the mist server's properties.

Service Provisioning Model in Fog Computing

Planning an algorithm that can vigorously adjust network settings and permit smart devices to choose a reliable fog node for service provisioning is vital. The best fog nodes for protected service provisioning need to be selected for the research. To solve this multi-objective optimization issue, the research suggests an effective selection method based on the Improved Salp Swarm Optimization (ISSO) and Modified Whale Optimization (MWO) algorithms. The fundamental problem of FC technology to handle the level of trustee trustworthiness is addressed by adopting a Trust Management (TM) system.

Improved Salp Swarm Optimization (ISSO)

The original Salp Swarm Optimization (SSO) algorithm is modified to establish the ISSO algorithm. The swarming behaviour of marine salps inspires SSO. The variables of an optimization problem are defined by salp positions, which are stored in the double-dimensional matrix. The population of salps is represented as:

$$X = \left[x_{i1}, x_{i2}, \ldots x_{ij}, \ldots x_{SV}\right] \quad \ldots (2)$$

where, $i \epsilon [1, 2, 3, \ldots S]$ and $j \epsilon [1, 2, 3, \ldots V]$. V stands for variables, while S stands for population size. Eq. (2) is applied to generate initial Salp locations ($x_{ij\_initial}$) in a matrix of size $S \times V$ at random. Initial salp positions are generated as:

$$x_{ij\_initial} = 1b_j + rand \times (ub_j - 1b_j) \quad \ldots (3)$$

where, they j^{th} variable's lower and upper bounds are $1b_j$, and ub_j, respectively. A random amount among $[0, 1]$is called $rand$. The Salp placements in each population are taken into account when calculating the objective function's (OF) value. Any problem is given an off while taking the parameters up for optimisation into account. By minimising or maximising the value of the OF, the first optimised parameters (the optimal inhabitants of the early Salp positions matrix) are discovered. The food source locations (f) are referred to as these ideal parameters. Following Eq. (3), the leader changes its position. Leader update rule:

$$x_{1j} = \begin{cases} f_j + a_1 \times \left(1b_j + a_2 \times \left(ub_j - 1b_j\right)\right), a_3 \geq 0.5 \\ f_j - a_1 \times \left(1b_j + a_2 \times \left(ub_j - 1b_j\right)\right), a_3 < 0.5 \end{cases} \quad \ldots (4)$$

The position of the j^{th} variable's food source is denoted by $f_j.a_1 = 2e^{-(\frac{4t}{T})}$There have been a total of t iterations, where T is the current iteration. The integers $[0, 1]$. between a_2, a_3 are at random. The followers similarly alter their locations according to the Follower update:

$$x_{ij} = \frac{1}{2} \times \left(x_{ij} + x_{i-1j}\right) \quad \ldots (5)$$

Controlling the population's capacity to do local and global searches modifies SSO. The presentation of the algorithm is enhanced by this control, thus the term ISSO. Eq. (5) and Eq. (6) are utilized in ISSO to update the followers' locations and integrate a dynamic weight factor called W.Improved ISSO update:

$$x_{ij} = \frac{1}{2} \times w_t \times \left(x_{ij} + x_{(i-1)j}\right) \quad \ldots (6)$$

Weight update:

$$w_t = w_{max} \times a_4 - \left(\frac{t}{T}\right) \times w_{max} - w_{min} \quad \ldots (7)$$

where, w_t is the weight element for the t^{th} iteration. The weight element's lower and upper bounds in the range [0,1] are denoted by W_{min} and W_{max} respectively. The literature goes into great detail on how adding w to position updates affects the results. By utilizing it on several benchmark functions, the authors have established and authenticated the recital of the ISSO method. Effectively increased optimisation concert is attained in consequence.

Equations (2–7) are based on Salp Swarm Optimization and its improved variant as discussed in metaheuristic optimization literature.[20]

By containing the vibrant weight element in the calculation for modernizing the followers' placements, the ISSO algorithm performs better overall. According to their prior position and the position of the preceding person, followers' positions are updated. The ISSO method becomes a promising and practicable optimisation technique once the weight factor and position update are added to the search strategy (Table 2).

Modified Whale Optimization Algorithm (MWOA)

In this part, WOA is altered by changing the control parameter and including additional search techniques. Even while the fundamental WOA with the encircling mechanism and spiral path is effective at navigating the search space, it still has to be improved to handle LSGO issues. Therefore, a quadratic interpolation (QI) approach is utilized in the MWOA to increase the exploitation capacity and solution accuracy. In n-dimensional space, QI mathematically determines the lowest point of the quadratic curve travelling through three selected solutions. While the quadratic crossover machine, QI, selects the best search agent $x^* = (x_1^*, x_2^*, \dots, x_n^*)$ and the other two partners $Y = (y_1, y_2, \dots, y_n), Z = (z_1, z_2, \dots, z_n)$ as three parents, and then produces an offspring (new solution) $X = (x_1, x_2, \dots, x_n)$ by the following Quadratic interpolation:

$$x_i = QI(y_i, z_i, X^*) \quad \dots (8)$$

Equation (8) is derived from Whale Optimization Algorithm with quadratic interpolation enhancement.[20,25] The bubble-net attacking style utilized by humpback whales in their distinctive hunting strategy served as the basis for the swarm intelligence optimisation algorithm known as WOA. The humpback whales' primary meal consists of schools of krill or tiny fish. During their hunting season, humpback whales exhibit two distinct behaviours associated with bubble nets. One of these behaviours is known as upward spirals, while the other is referred to as the double loop, which consists of three stages: the coral loop, lobtail, and capture loop. In this process, the whales dive down and generate spiral-shaped bubbles around their prey before ascending towards the surface. WOA coefficient equations:

$$A = 2ar - a \quad \dots (9)$$

$$C = 2r \quad \dots (10)$$

Table 2 — Algorithm forImproved Salp Swarm Optimization

Algorithm 1: Improved Salp Swarm Optimization

Initialize the population of salps
Evaluate the fitness of each salp
Identify leader(s)
Repeat until termination criteria are met:
Update salp positions
Evaluate the fitness of updated salps
Update leader(s)
Check termination criteria
Extract best solution(s)

where, → a is a vector that reduces linearly from two to zero during the iterations. One aim is the original WOA. Kumawat effectively modified the MOWOA approach to address multi-objective problems, showing strong exploration and exploitation in a specific search region. MOWOA has shown earlier convergence and achieved results with fewer parameters. It has effectively addressed many previously unresolved issues. MOWOA has not yet been deployed to tackle the ideal offloading technique for compute offloading in mobile edge computing, making it a persistent problem. Equations (9–10) follow standard Whale Optimization Algorithm formulation.[20]

Intrusion Detection with Offloading Model

Cyber dangers nowadays, such as invasions, are diffused over several networks utilizing scattered networking resources. Intrusion detection systems (IDSs) have already established themselves as a crucial defence against several threats. By shifting costly processes like the process of signature matching to the cloud (i.e., using computational resources from the cloud), a contemporary IDS can incorporate more complex detection methods. The suggested load-sharing plan addresses the security issue by building a system for load-offloading that is based on trust. The paper suggests a hierarchical collaborative federated learning model for intrusion detection based on fog devices in light of the development of fog computing. With the aid of this suggested structure, the burden of communication latency in the fog layer may be lessened.

Hierarchical Collaborative Federated Learning

The suggested hierarchical federated learning-based intrusion detection system (HFed-IDS) leverages edge servers to enable smart meters in different clusters to collaboratively train a shared transformer IDM. This approach allows smart meters to utilize their local data while minimizing core network traffic. The proposed solution, called CFL (Cloud-Free Federated Learning), offers a way for devices to contribute to federated learning without the essential to be connected to a base station (BS) or a cloud server. In CFL, certain devices are linked to the BS directly, while other devices are connected to a specific number of nearby devices. For instance, because of the possibility of a significant transmission delay, OFL, device b cannot be directly linked to the BS and execute FL. Device b can, however, be attached to Device a in CFL to carry out FL. Utilizing broadcast techniques may allow CFL to efficiently

convey local FL model parameters to multiple devices, thus potentially consuming fewer resources compared to OFL. The training of CFL is also iterative. At the start of HFed-IDS, contributing smart meters execute H local gradient update epochs and collect a worldwide model from the cloud server. The participant's trained model is then uploaded to the BS. In contrast to the traditional FL, the base station directly collects the established models at the edge server after receiving the models from each member in the cluster to minimise the number of models that need to be communicated to the cloud server. It is possible to express the aggregation procedure at the base station B_j edge server as follows:

$$W_{B_j}^{(t+1/2)} = \sum_{i \in C_j} \frac{|D_i|}{\sum_{i \in C_j} |D_i|} \omega_i^{(t+1/2)}, \quad \ldots (11)$$

where, $W_{B_j}^{(t+1/2)}$ is the cluster model that has been placed at the base station B_j edge server and C_j is the B_j participant set. Then, rather than sending all of the participants' models to the cloud server for combination, each cluster model $W_{B_j}^{(t+1/2)} (j \in \{1, \ldots, U\})$ is sent to the cloud server. This can considerably cut down on communication costs. You may write the global aggregate as:

where, $\omega^{(t+1)}$ is the global model and $W_{B_j}^{(t+1/2)}$ is the cluster model's aggregation weight, and the cluster C_j total dataset size. The result suggested HFed-IDS not only performs similarly to the FedAvg procedure but also can lower communication charges. All participants are then supplied with the obtained innovative global model for the following iteration. It should be noted that HFed-IDS protects user privacy by not sending raw data to a cloud server. Additionally, the participating smart metres may now acquire a powerful intrusion detection model, enabling them to carry out local assault detection. Additionally, even though the recommended HFed-IDS calls for certain edge servers to accomplish transitional aggregation, its implementation will be simple given that 5G networks would frequently install edge servers at base stations for edge computing applications. Equation (11) follows the Federated Averaging (FedAvg) aggregation mechanism.[12,30]

Note: The mathematical formulations in Equations (2–11) are derived from established metaheuristic optimization and federated learning frameworks reported in the literature.

Experimentation, Results and Discussion

FC is a concept that brings low-latency and context-aware facilities to the network's edge by extending cloud computing capabilities. However, there are substantial difficulties in controlling the resources and LB in fog situations while preserving trust and security. The experiment proposes a novel strategy that combines load balancing, energy awareness, and trust-based decision-making methods utilising a multi-objective optimisation algorithm to tackle these problems. The aim is to optimise the distribution of activities and possessions among fog nodes while taking into account their dependability, energy consumption, and load-balancing needs. The proposed framework is assessed and estimated using MATLAB software. The experiment entails establishing an FC environment, applying the suggested method, and measuring the algorithm's performance with various metrics.

The simulation setup for the proposed method is detailed in Table 3. The study work was shown using MATLAB version R2021a, running on a computer with a Core i3 processor clocked at 3.5 GHz and equipped with DDR3-6GB RAM.

The training and validation loss of fog computing for secure provisioning and load balancing over 50 iterations is shown in Fig. 2. Loss is a metric used in machine learning and optimization algorithms to quantify the deviation between forecast and actual values. In this context, the training loss of 0.6 indicates that during the training phase, the algorithm was able to minimize the error to 0.6, which means it was able to fit the data well and make accurate predictions on the training dataset. On the other hand, the validation loss of 0.3 indicates that the algorithm also performed well on the data, making accurate predictions with low error.

Comparison Analysis

As part of the comparative study, various techniques are evaluated and analysed to determine the most effective approach, supporting the recommended methodology. The comparison analysis is a systematic process used to assess the similarities and differences among two or more subjects, objects,

Table 3 — System configuration for simulation

MATLAB	Version R2021a
Operating System	Windows 10 Home
Memory Capacity	6GB DDR3
Processor	Intel Core i3 @ 3.5GHz

concepts, or methods. This critical approach is widely employed across different fields, including the comparison of power efficiency, average response time, and Makespan of the suggested technique with the compared algorithms, as elucidated in this section.

The performance between the M2F balancer method and the earlier WRR computations highlights the superiority of the former in achieving a higher ARU (Average Resource Utilization), as shown in Fig. 3. The X-axis signifies the number of tasks, while the Y-axis displays the ARU values. The proposed M2F balancer achieved impressive ARU values of 72%, 73%, 88%, and 85%, respectively, outshining the LC and RR techniques with values below 70%. The graphical representation demonstrates that the proposed M2F balancer method excels in fog computing, attaining a remarkable level of ARU, and stands apart from the traditional load-balancing algorithm. By achieving both high resource application and low imbalance, this illustrates that the proposed algorithm outperforms the previous load-balancing algorithms.

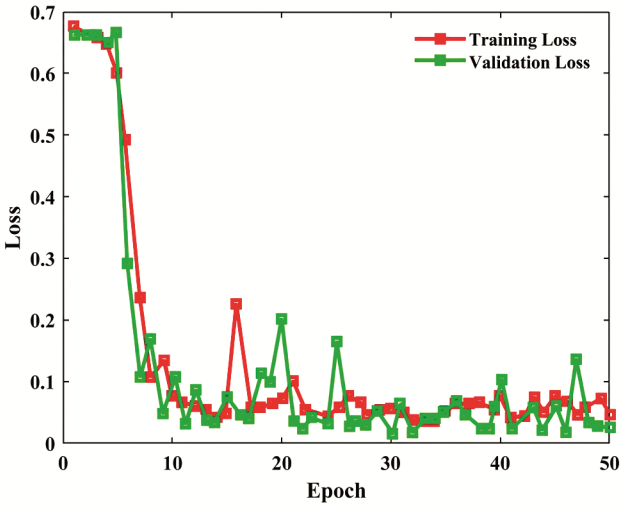

Fig. 2 — Training and Validation Loss of 50 Iterations

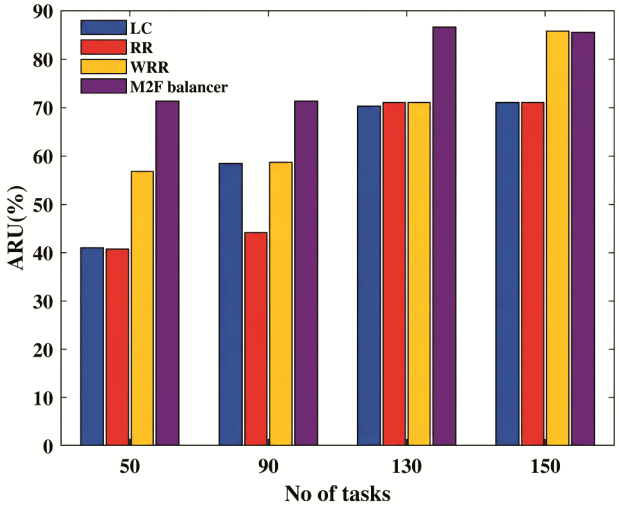

Fig. 3 — Comparison analysis of average resource utilization

The average reaction times achieved by employing various baseline approaches, including A3C, MAD-MC, and DQLCM, on the simulator presented in Fig. 4. The proposed approach demonstrated the fastest average response time, contrasted with the other methods. In a simulation configuration involving 50 hosts, both the A3C and MAD-MC approaches exhibited relatively rapid task completion times on average. However, the suggested technique outperformed the others, showcasing a reduced average reaction time of 1300 seconds. The average response time for the proposed method lies above 1200 seconds, whereas the A3C approach recorded the highest response time, exceeding 1800 seconds. The A3C deep reinforcement learning method combines asynchronous training and an actor-critic architecture, providing certain advantages in certain scenarios.

The comparison between the efficiency of the suggested method and the previously used load-balancing algorithm, showcasing their performance results, is displayed in Fig. 5. The Y-axis signifies the makespan level, while the X-axis shows the number of tasks. The figure clearly demonstrates that the M2F balancer method beats the prior load balancing algorithm, attaining an unusually low makespan level of 70 ms for the 50th task in fog computing. This establishes the important development delivered by the

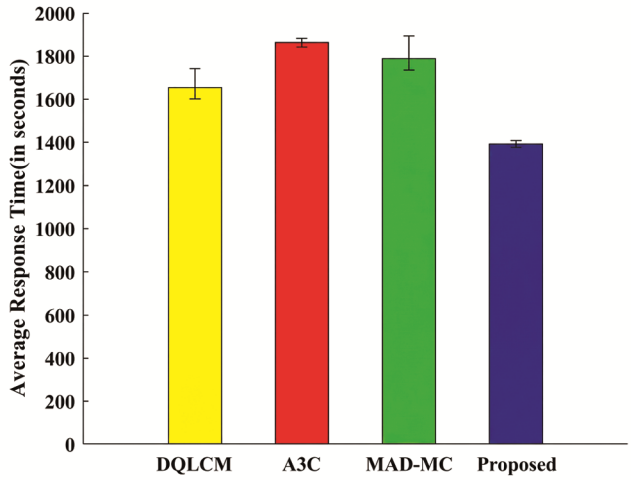

Fig. 4 — Comparison analysis of the average response time

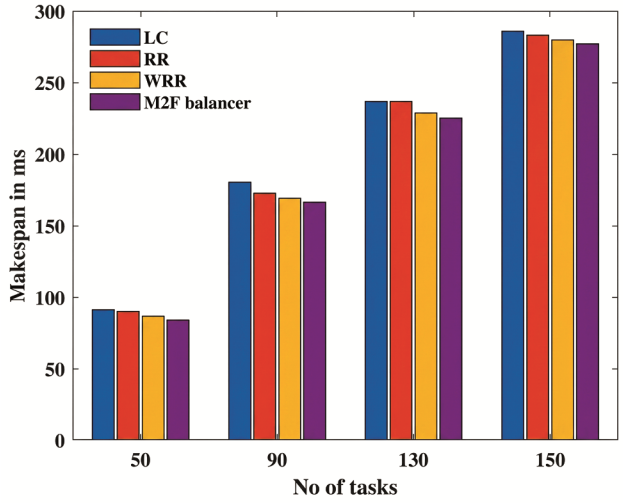

Fig. 5 — Makespan for M2F balancer with the compared algorithms

recommended technique in restricting the makespan associated with the WRR load-balancing algorithm. Task 150 marks the maximum value between the other values, with the M2F balancer recording a value above 250 ms for the suggested method.

Conclusions

The suggested system delivers a complete solution for Fog Computing (FC) by successfully incorporating secure resource provisioning, collaboration, and load balancing. By applying the M2F balancer alongside hybridized ISSO and MWOA, the framework exploits resource utilization while certifying defense against network attacks through hierarchical federated learning. Experimental results specify important enhancements in response time, cost, and energy consumption, attaining a high attack detection precision of 99.5%. A key restriction of this study is the dependence on pretend environments, which may not capture the full difficulty of stochastic practical network variations. Future research will focus on scalability analysis and real-time applications across various industrial IoT sectors. Potential applications include smart grid management and autonomous traffic systems where low-latency resource allocation is essential. Also, incorporating innovative machine learning for adaptive trust assessment could significantly improve the long-term resilience of the decentralized system against developing cyber threats.

References

1 Aldossary D, Aldahasi E, Balharith T & Helmy T, A systematic literature review on load balancing techniques in fog computing: architectures, strategies, and emerging trends, *Computers*, **14(6)** (2025) 217.

2 Saeedi Taleghani E, Maldonado Valencia R I, Sandoval Orozco A L & García Villalba L J, Trust evaluation techniques for 6G networks: a comprehensive survey with fuzzy algorithm approach, *Electronics*, **13(15)** (2024) 3013.

3 Isah S, Agushaka J O & Agber S, Trust-aware energy optimization techniques in mobile ad-hoc networks: current trends and future directions, *Proc Faculty Sci Conf*, **1** (2024) 65–79.

4 Mangalampalli S, Karri G R, Gupta A, Chakrabarti T, Nallamala S H, Chakrabarti P, Unhelkar B & Margala M, Fault-tolerant trust-based task scheduling algorithm using Harris hawks optimization in cloud computing, *Sensors*, **23(18)** (2023) 8009.

5 Alvi A N, Ali B, Saleh M S, Alkhathami M, Alsadie D & Alghamdi B, Secure computing for fog-enabled industrial IoT, *Sensors*, **24(7)** (2024) 2098.

6 Saini H, Singh G, Dalal S, Moorthi I, Aldossary S M, Nuristani N & Hashmi A, A hybrid machine learning model with self-improved optimization algorithm for trust and privacy preservation in cloud environment, *J Cloud Comput*, 13(**1**) (2024) 157.

7 Hashemi S M, Sahafi A, Rahmani A M & Bohlouli M, Service and energy management in fog computing: A taxonomy, approaches, and future directions, *J Electr Comput Eng Innov*, **12(1)** (2024) 15–38.

8 Rajammal K & Chinnadurai M, Dynamic load balancing in cloud computing using predictive graph networks and adaptive neural scheduling, *Sci Rep*, **15(1)** (2025) 22181.

9 Kumar M, Samriya J K, Dubey K & Gill S S, QoS-aware resource scheduling using whale optimization algorithm for microservice applications, *Softw Pract Exp*, **54(4)** (2024) 546–565.

10 Yin X, Zhang X, Pei L, Hu R, Ye K & Cai K, Optimization and benefit evaluation model of a cloud computing-based platform for power enterprises, *Sci Rep*, **15(1)** (2025) 26366.

11 Tareen F N, Alvi A N, Malik A A, Javed M A, Khan M B, Saudagar A K J, Alkhathami M & Hasanat M H A, Efficient load balancing for blockchain-based healthcare system in smart cities, *Appl Sci*, **13(4)** (2023) 2411.

12 Aburukba R, Federated learning-driven IoT request scheduling for fault tolerance in cloud data centers, *Mathematics*, **13(13)** (2025) 2198.

13 Hemmati A, Khaledian N & Rahmani A M, Toward a unified computing paradigm: a survey and roadmap for data management in integrated IoT, fog, and cloud services, *Peer-to-Peer Netw Appl*, **19(2)** (2026) 54.

14 Fatima R, Anjum S, Junaidi S F & Rafa R, Resource management in the edge–cloud continuum: trends, algorithms, and open challenges, *J Syst Eng Electron*, **35(12)** (2025) 176.

15 Retima F & Benharzallah S, A comparative analysis of service composition approaches for the internet of things, *Ing Syst Inf*, **30(1)** (2025) 111–135.

16 Rahdari A, Jalili A, Esnaashari M, Gheisari M, Vorobeva A A, Fang Z, Sun P, Korzhuk V M, Popov I, Wu Z & Tahaei H, Security and privacy challenges in SDN-enabled IoT systems: causes, proposed solutions, and future directions, *Comput Mater Continua*, **80(2)** (2024) 2511–2533.

17 Mushtaq S U, Sheikh S, Nain A, Bharany S, Ghoniem R M & Taye B M, CRFTS: a cluster-centric and reservation-based

fault-tolerant scheduling strategy to enhance QoS in cloud computing, *Sci Rep*, **15(1)** (2025) 32233.

18 Chennam K K, Maheswari U V, Aluvalu R, Chinthaginjala R, Wahab M N A, Zhao X & Tolba A, Load balancing for cloud computing using optimized cluster based federated learning, *Sci Rep*, **15(1)** (2025) 41328.

19 Rezaee M R, Abdul Hamid N A W, Hussin M & Zukarnain Z A, Fog offloading and task management in IoT-fog-cloud environment: review of algorithms, networks, and SDN application, *IEEE Access*, **12** (2024) 39058–39080.

20 Zhou J, Lilhore U K, Hai T, Simaiya S, Jawawi D N A, Alsekait D M, Ahuja S, Biamba C & Hamdi M, Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing, *J Cloud Comput*, **12(1)** (2023) 1–21.

21 Yang X, Zeng Z, Liu A, Xiong N N & Zhang S, ADTO: a trust active detecting-based task offloading scheme in edge computing for internet of things, *ACM Trans Internet Technol*, **24(1)** (2024) 1–25.

22 Al Moteri M, Bhatia Khan S & Alojail M, Machine learning-driven ubiquitous mobile edge computing as a solution to network challenges in next-generation IoT, *Systems*, **11(6)** (2023) 308.

23 Vu T T, Chu N H, Phan K T, Hoang D T, Nguyen D N & Dutkiewicz E, Energy-based proportional fairness in cooperative edge computing, *IEEE Trans Mobile Comput*, **23(12)** (2024) 12229–12246.

24 Isyaku B & Abu Bakar K, Managing smart technologies with software-defined networks for routing and security challenges: a survey, *Comput Syst Sci Eng*, **47(2)** (2023) 1839–1879.

25 Mangalampalli S, Karri G R, Ratnamani M V, Mohanty S N, Jabr B A, Ali Y A, Ali S & Abdullaeva B S, Efficient deep reinforcement learning based task scheduler in multi cloud environment, *Sci Rep*, **14(1)** (2024) 21850.

26 Saini H, Singh G, Dalal S, Lilhore U K, Simaiya S & Dalal S, Enhancing cloud network security with a trust-based service mechanism using k-anonymity and statistical machine learning approach, *Peer-to-Peer Netw Appl*, **17(6)** (2024) 4084–4109.

27 Sharma G, Khare R, Kulkarni N, Pagare S & Tiwari V, Design of an iterative AI-driven latency prediction and QoS-aware task scheduling in mobile edge computing: A federated and reinforcement learning process, *EPJ Web Conf*, **328** (2025) 01071.

28 Mallu S R K & Mangalampalli S, A novel fault-tolerant aware task scheduler using deep reinforcement learning in cloud computing, *Appl Sci*, **13(21)** (2023) 12015.

29 Sugiartana N F M, Sastra N P, Wiharta D M & Sambas A, Adaptation in IoT-fog data transmission: SLR and future perspectives on dynamic frequency control, *J Tek Elektro*, **17(1)** (2025) 20–30.

30 Jorquera Valero J M, Sanchez Sanchez P M, Gil Perez M, Huertas Celdran A & Martinez Perez G, Cutting-edge assets for trust in 5G and beyond: requirements, state of the art, trends, and challenges, *ACM Comput Surv*, **55(11)** (2023) 1–36.

31 Gomo S & Lim C K, Heuristic algorithms for mobile edge computing recovery system based on ad-hoc relay nodes, *Int J Comput Math*, **1(4)** (2024).

32 Manogaran N, Raphael M T M, Raja R, Jayakumar A K, Nandagopal M, Balusamy B & Ghinea G, Developing a novel adaptive double deep Q-learning-based routing strategy for IoT-based wireless sensor network with federated learning, *Sensors*, **25(10)** (2025) 3084.

33 Alalwany E & Mahgoub I, Security and trust management in the internet of vehicles (IoV): challenges and machine learning solutions, *Sensors*, **24(2)** (2024) 368.

34 Shankar S R, Raminaidu C H, Ravibabu D & Gupta V M N S S V R, A survey to raise the awareness of road accidents due to not wearing helmet, *Int J Ind Eng Prod Res*, **31(3)** (2020) 367–377.