

SHIELD: Security based hybrid autonomous deep learning network for load balancing in cloud

Loga Priyadarshini Kathirmalaiyan, Nithya Muthu

Centre for Artificial Intelligence & Data Science (CAIDS), Department of Computer Science and Engineering, Vinayaka Mission Kirupananda Variyar Engineering College, Vinayaka Mission Research Foundation (DU), Salem, Tamil Nadu, India

Article Info

Article history:

Received Feb 6, 2025

Revised Aug 8, 2025

Accepted Oct 22, 2025

Keywords:

Gated recurrent neural network

Internet of Things

Load balancing

Recurrent neural network

Security constraints

ABSTRACT

Load balancing in the internet of things (IoT) enhances the efficiency of the system by dynamically allocating tasks across devices and cloud resources. However, task scheduling struggles with unpredictable tasks, scalability, security risks, and unauthorized access control. To overcome these limitations, a novel security-based hybrid autonomous deep learning network for load balancing in cloud (SHIELD) framework has been proposed for secure task scheduling in cloud resources. Initially, the data received from the IoT devices is passed under certain security constraints to ensure the authenticity of the data. These privacy-preserved data are fed to the task scheduling module, which is employed by the dual DL Network to generate a schedule for resource management. Finally, cloud resources employ optimal allocation of tasks based on the generated schedule to ensure secure load balancing. The proposed framework is simulated by using Cloud Simulator 7G (CloudSim7G). The SHIELD framework is assessed by such metrics, including accuracy, recall, precision, F1-score, and specificity. In comparison, the proposed SHIELD framework achieves a privacy overhead of 14% outperforms the existing QODA-LB, Best-KFF, SPSO-TCS, and VMMISD techniques by achieving 10%, 11%, 12%, and 13% respectively.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Loga Priyadarshini Kathirmalaiyan

Vinayaka Mission Kirupananda Variyar Engineering College, Vinayaka Mission Research Foundation (DU)
Salem, Tamil Nadu, India

Email: kavipriyas13@gmail.com

1. INTRODUCTION

Internet of things (IoT) and cloud computing (CC) have revolutionized data processing and resource management enabling the convenient operation of linked devices [1], [2]. Real-time processing and efficient task scheduling are necessary because of the enormous volumes of data generated through IoT devices [3], [4]. Adaptive load balancing in such systems ensures optimal resource utilization and prevents service interruptions [5], [6]. However, with the surge in data exchange and task scheduling between IoT devices and cloud environments security vulnerabilities have escalated making it imperative to adopt secure mechanisms that safeguard task execution while maintaining system performance [7], [8]. Secure task scheduling in IoT-cloud systems typically involves the implementation of cryptographic techniques, anomaly detection mechanisms, and intelligent decision-making frameworks [9], [10]. Additionally, leveraging machine learning (ML) models for dynamic threat detection and adaptive scheduling further enhances system resilience against potential attacks [11], [12]. Several existing task scheduling techniques such as heuristic-based, metaheuristic, and rule-based algorithms have been developed to optimize resource allocation in IoT-cloud systems. Approaches such as ant colony optimization (ACO), and first-come-first-serve (FCFS)

Scheduling focus on minimizing execution time and energy consumption [13], [14]. While these models address various performance metrics their lack of built-in security features often exposes IoT-cloud systems to vulnerabilities [15], [16]. Enhancing these techniques with secure load-balancing measures is a crucial step toward reliable IoT-cloud integration [17], [18].

Task scheduling and resource allocation in IoT systems have drawn the attention of numerous academics in recent years. This section focuses on dynamic load balancing that compiles the relevant research in these fields. In 2022, Latchoumi and Parthiban [19] suggested by using the quasi-oppositional based learning principle, a QODA-LB technique can be applied to boost the dragonfly algorithm's (DA) typical convergence rate. The suggested QODA-LB technique computes an objective function based on execution time and execution cost. In 2022, Fathalla *et al.* [20] suggested an Allocation of resources for CC systems by employing a multi-goal strategy. Preemption times, average waiting times for advanced reservations, lease pre-emption, and energy usage are all decreased by the best-KFF technique while optimizing the PMS' use of resources. In 2023, Devi *et al.* [21] suggested Utilizing a deep load balancer with improved security and CLOUD load balancing to store data for the IoT. The suggested DLB approach achieves better scalability, cost efficiency, and enhanced security for efficient load balancing. In 2023, Saba *et al.* [22] suggested Cloud-edge load balancing for internet of everything (IoE) applications based on distributed swarm intelligence. The suggested approach lowers network cost, end-to-end latency, success rate, and energy usage by 20%, 17%, and 19%, respectively.

In 2023, Aghapour *et al.* [23] suggested allocating resources and unloading tasks through deep reinforcement learning (DRL). The suggested method's average power consumption and latency value are 92%, 17%, and 12%, respectively. In 2024, Menaka and Kumar [24] suggested the load balancing strategy in CC is supportive particle swarm optimization with time-conscious scheduling (SPSO-TCS). The SPSO-TCS method uses TCS and SPSO to shorten the make-span time and achieve initial load balance. In 2024, Brahman and Anand [25] suggested a robust virtual machine load balancing paradigm. VM migration efficiency increased by 2.5%, projected time lowered by 9.5%, and computational complexity decreased by 8.3% with the aid of the proposed VMMISD. Therefore, one of the primary shortcomings of existing approaches is the requirement for load balancing strategies and resource allocation that consider both task requirements and destination node capabilities.

To overcome these issues, a novel security-based hybrid autonomous deep learning network for load balancing in cloud (SHIELD) framework has been proposed to develop a security-based task scheduling in IoT devices for preserving the sensitive information from unauthorized access and mitigate risks from cyber threats among the network. The key goals of the SHIELD framework are described as follows: The aim of this SHIELD framework is to develop an efficient framework for robust load balancing in IoT environments by incorporating security and privacy-based measures with advanced hybridized dual deep learning (DL) networks. The SHIELD framework provides substantial data security by several security concerns such as data confidentiality, access control, data integrity, anomaly detection, and data availability to ensure that the IoT data is resilient to threats. The hybridized dual recurrent neural network (HDRNN) processes the tasks based on priority and the gated recurrent neural network (GRU) recognizes the dependencies between tasks to contribute to a precise task schedule. A secured load balancing and efficient resource management is facilitated by cloud resources based on the schedule generated by the HDRNN for optimal task execution in IoT environments. The remaining sections of the research is given as follows: Section 2 presents the SHIELD Framework. Section 3 describes the experimental results of the SHIELD framework. Section 4 presents the discussion of the research and section 5 concludes the research with future enhancement.

2. THE SHIELD METHODOLOGY

In this section, a novel Security based hybrid autonomous deep learning network for load balancing in cloud (SHIELD) framework has been proposed to develop a privacy-based task scheduling in IoT devices using dual DL networks. Initially, raw data such as task data, resource data, task constraints, IoT environment information, and historical data are gathered from the IoT devices for dynamic task scheduling. A zero-trust security module is integrated to verifies every input data through real-time evaluations and context-aware policies that detect adversarial attacks. These verified data are fed to several security constraints such as data confidentiality, access control, data integrity, anomaly detection, and data availability for data verification. These verified and secured data along with privacy features are fed to the HDRNN for task scheduling. The HDRNN generates a schedule by processing the tasks based on priority through RNN and the GRU network recognizes the dependencies between tasks. The SHAP method processes the scheduling decision from HDRNN and assigns an importance score to each task according to the task features for the final task schedule. LIME provides a task specific explanation and ensures that the high-priority and sensitive tasks are handled correctly without prioritizing the unauthorized tasks. Finally, the cloud resources allocate the task

based on the schedule generated by the HDRNN to balance the load securely with optimal resource utilization. The block diagram of the developed SHIELD model is represented in Figure 1.

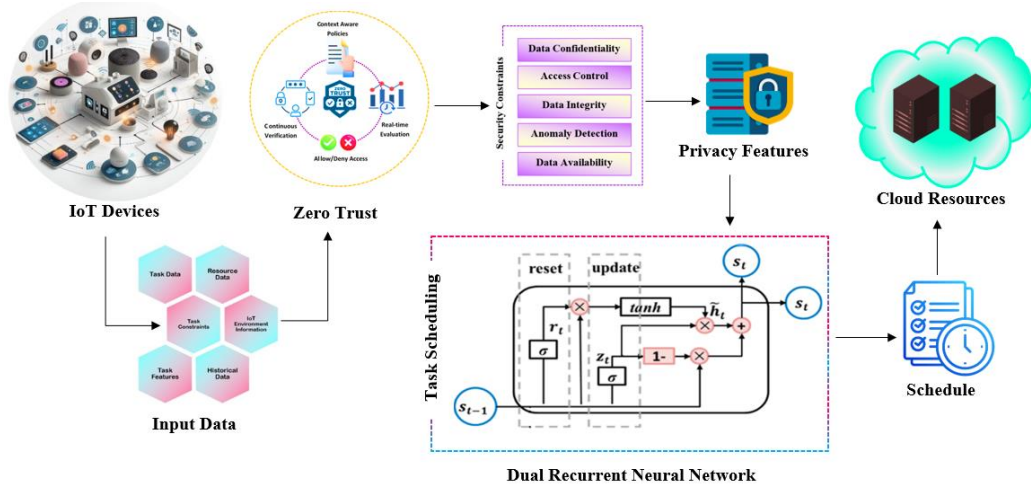


Figure 1. Proposed SHIELD framework

2.1. Zero-trust mechanism

The diverse types of data are gathered from the IoT devices which should contain such security risks. The input data from the IoT devices are categorized into task data, resource data, task constraints, IoT contextual data, task features, and historical data which consist of an operational environment and resource demands. The zero-trust module is an advanced and continuous security mechanism that verifies every input data through real-time evaluations and context-aware policies. These zero-trust robust security mechanism verifies the data and detects the black-box adversarial attacks including model evasion, model extraction, and poisoning attempts for efficient threat mitigation also to ensure secure and verified load balancing among the cloud network.

2.2. Privacy based features

The input data gathered from the IoT devices needs to be verified if the task's security level is less than that of the unassigned task. The secured tasks are extracted from the input data by validating certain security constraints such as access control, confidentiality, anomaly detection, and data availability. Moreover, if the security constraints are evaluated and the obtained solution is infeasible then the task is represented as an unsecured task which will also be filtered. Privacy-based filtering is given in Algorithm 1.

Algorithm 1: Privacy based filtering

Input: Security constraint of tasks

Output: Privacy Based Features

```

1. for  $i = 1$  to  $N$  do
2.   for  $j = 1$  to  $|T|$  do
3.      $v_{ij} \leftarrow \text{random}()$ ,  $x_{ij} \leftarrow \text{random}()$ 
4.     if  $(PC(Res(t_i)) < PC(t_i))$  do
5.        $j = j - 1$ 
6.     end if
7.   end for
8.   while  $(FT(CP) > UD)$  do
9.     for each  $tc \in CP$  do
10.       $v_i.[t_c.id] \leftarrow \text{random}()$ ,  $x_i.[t_c.id] \leftarrow \text{random}()$ 
11.    end for
12.  end while
13. end for

```

These secured data along with certain privacy features are fed to hybridized dual neural network for secure task scheduling.

2.2. Task scheduling via HDRNN

These verified data along with privacy features are fed to the HDRNN which combines recurrent neural network and gated recurrent neural network for task scheduling. The SHIELD methodology for

SHIELD: Security based hybrid autonomous deep learning network ... (Loga Priyadarshini Kathirmalaiyan)

scheduling the secured data utilizing RNN, Dense, and GRU layers. The architectural configuration of the HDRNN is shown in Figure 2. The h_t and x_t determines the hidden state and input.

A hidden state h_t and an input x_t from the previous time step are used by the RNN to create a new h_{t+1} and an output O_t at each time step t . Equations (1) through (2) list the steps.

$$h_{t+1} = f(W_{xh} * x_t + W_{hh} * h_t + b) \quad (1)$$

$$O_t = g(W_{oh} * h_{t+1} + b_o) \quad (2)$$

Here, W_{xh} , W_{hh} , and W_{oh} ensures the input to hidden, hidden to hidden, and hidden to output weight matrices are distinguished by their classification. In this case, b_o and b are biased terms. The hidden layer employs g , and the output layer uses f . It is connected to the dense layer by each neuron from the preceding levels.

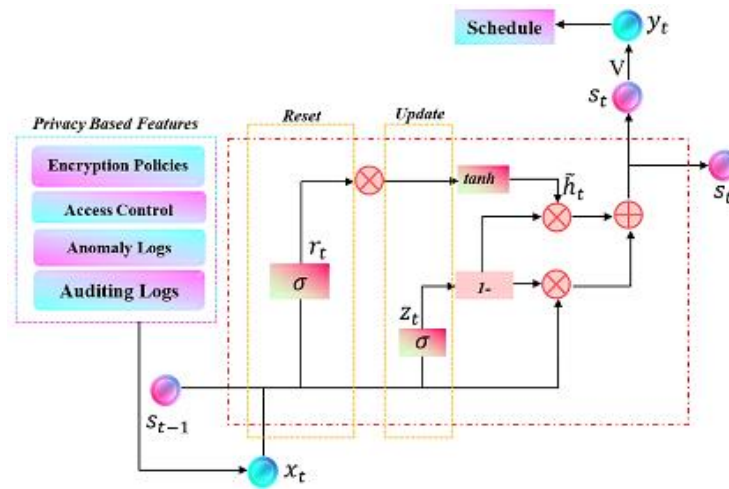


Figure 2. Architecture of HDRNN

Equation (3) describes the ReLU of the dense layer, and the (4) displays the predicted outcome of the dense layer. Where b is biased, x is the input of the dense layer, and W_h is the weight matrix. The gating method allows the GRU and RNN to select update or ignore data from the preceding time step.

$$f(x) = \max(0, x) \quad (3)$$

$$h = f(W_h * x + b) \quad (4)$$

The reset and update gates are the two gates in the GRU that regulate the flow of information. The GRU's output is transformed into a probability distribution across all potential outputs using the SoftMax activation function. Equation (5)–(8) provides a mathematical representation of the GRU operations. W_{xz} , W_{hz} , W_{xr} , W_{hr} , W_{xh} , and W_{hh} represents the weight matrices in input-update, input-reset, input-hidden, and hidden-hidden connections. Here, b_r , b_z , b_x , and b_h are the biased terms.

$$z = \text{sigmoid}(W_{xz} * x + W_{hz} * h + b_z) \quad (5)$$

$$r = \text{sigmoid}(W_{xr} * x + W_{hr} * h + b_r) \quad (6)$$

$$h' = \tanh(W_{xh} * x + W_{hh} * (r * h) + b_h) \quad (7)$$

$$h = z * h + (1 - z) * h' \quad (8)$$

The HDRNN processes the verified IoT task to generate the scheduling decision based on the task priority and task dependency. The SHAP method processes the scheduling decision from HDRNN and assigns an importance score to each input feature for the final task schedule according to the urgency of the task, task size, task deadline, and resource availability. LIME provides a task-specific explanation to the

admins to understand the exact reasons behind the decision for a particular task scheduling. LIME ensures that the high-priority and sensitive tasks were handled correctly without prioritizing the unauthorized tasks. Pruning is a strategy that reduces the size and computational complexity of the model without significantly affecting its performance, increasing the HDRNN's efficacy and efficiency. Pruning is the process of eliminating network elements that have the least effect on the network's output, such as weights and neurons. This approach produces a compact and effective model that enables quicker inference times and lower energy use. To minimize the loss function, the gradient descent is used to update the weights throughout the backpropagation training. Because quantization is necessary for both model size reduction and inference speed, it can be used in situations with limited resources. One important method for compressing models is quantization, which lowers storage needs and improves computation efficiency. Quantization, a technique that increases computing speed and memory use without substantially affecting model performance, involves simplifying the model and reducing the precision of the weights and activations. The HDRNN generates a schedule by processing the tasks based on priority through RNN and the GRU network recognizes the dependencies between tasks. Finally, the cloud resources allocate the task based on the schedule generated by the HDRNN to balance the load securely with optimal resource utilization.

3. RESULT AND DISCUSSION

The experimental assessment of the SHIELD approach is discussed in this section. The PC used for the analysis is the Windows 10 Pro with an Intel (R) Core (TM) i5 CPU, clocking at 3.20 GHz and with 8 GB of RAM. The SHIELD approach is simulated by using CloudSimulator (CloudSim7G).

3.1. Comparative analysis

The SHIELD model is analyzed with several previously developed models and displays experimental results. The accuracy, recall, precision, F1-score, and specificity of the SHIELD strategy are contrasted with existing methods in Figure 3(a). The SHIELD strategy outperforms the QODA-LB, Best-KFF, SPSO-TCS, and VMMISD models and achieves an accuracy of 85%, 90%, 88%, 87%, and 93.5% respectively. The comparison of power consumption for the SHIELD framework is represented in Figure 3(b). The SHIELD framework achieves the lowest power consumption of 18.2W to 24.3W across all task ranges from 2000 to 10,000. This indicates the SHIELD framework's superior energy efficiency by maintaining up to 41% lower consumption at higher loads against the QODA-LB, Best-KFF, SPSO-TCS, and VMMISD techniques.

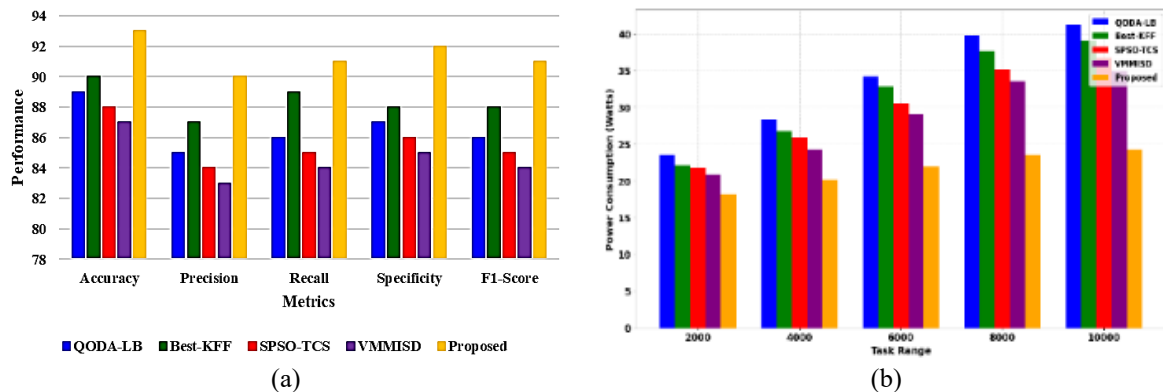


Figure 3. Analysis of (a) performance and (b) power consumption of SHIELD framework

The SHIELD framework consistently outperforms existing techniques in scalability across all tasks is represented in Figure 4. It achieves scalability rates of 89%, 87%, 85%, and 83% for tasks 100, 200, 300, and 400, respectively. The SHIELD framework outperforms existing QODA-LB, Best-KFF, SPSO-TCS, and VMMISD approaches by maintaining the highest scores from 93% at 2000 tasks to 86% at 10,000 tasks respectively. The SHIELD framework shows the superior ability to handle increasing task loads efficiently, with up to 20.3% better scalability at higher task ranges. The proposed SHIELD framework outperforms the existing techniques in resource utilization across all tasks is shown in Figure 5. For tasks 100, 200, 300, and 400, it achieves utilization rates of 89%, 87%, 85%, and 83% respectively. In comparison, QODA-LB achieves 70%, Best-KFF achieves 72%, SPSO-TCS achieves 71%, and VMMISD achieves 74% respectively.

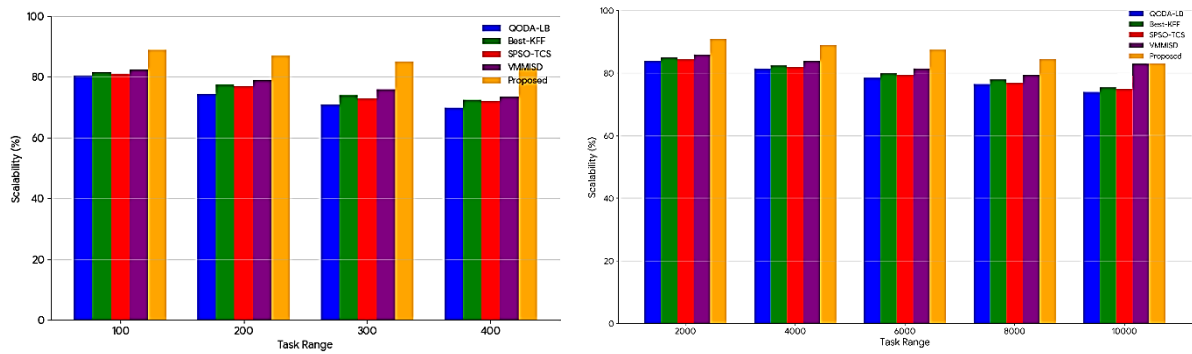


Figure 4. Analysis of scalability

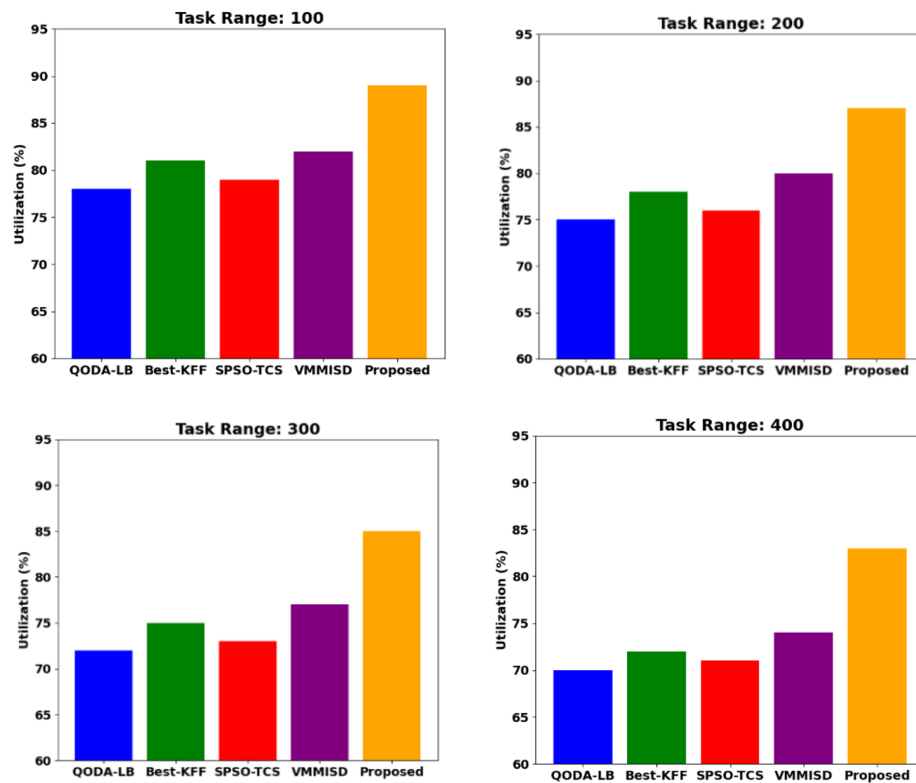


Figure 5. Analysis of resource utilization

The proposed framework incurs slightly higher privacy overheads compared to existing techniques across all tasks which are illustrated in Figure 6. For tasks 100, 200, 300, and 400 it obtains overheads of 14%, 16%, 18%, and 20%, respectively. Figure 7 depicts the performance assessment of the SHIELD model with existing Hybrid DQN-PPO-GNN-RL, HAFedRL, and TF-DDRL techniques. The proposed SHIELD framework achieves decentralized security of 97%, adaptive task scheduling of 92%, and sequence prediction of 94% which outperforms the existing techniques respectively.

Figure 8 shows the comparison of consumption of power by the Edge AI Devices for energy efficient load balancing. The ESP32 device attains the lowest power consumption of 0.08 W, 0.25 W, and 0.4 W for idle power, average power, and maximum power for SHIELD scheduling respectively. In contrast, Jetson Nano consumes high power of 2.6 W, 5.1 W, 5.6 W for idle power, average power, and maximum power which is suitable for power-rich AI-intensive tasks. In real-world deployment, the SHIELD framework will provide efficient performance under heterogeneous IoT devices for robust and scalable load balancing.

The comparison on latency is depicted in Figure 9. The SHIELD framework consistently achieves lower delay compared to existing techniques such as QODA-LB, Best-KFF, SPSO-TCS, and VMMISD

across all tasks. For tasks 100, 200, 300, and 400, it achieves delays of 95 ms, 100 ms, 110 ms, and 120 ms, respectively. The SHIELD framework achieves the lowest latency of 97 ms for 2000 tasks to 249 ms for 10,000 tasks for handling high task loads. In real-world deployment, the efficiency of the SHIELD framework is not varied during network latency fluctuations for high-speed and time-efficient load balancing. The SHIELD framework achieves a superior computational efficiency of 83.9 to 74.3 at 2000 to 10,000 tasks which outperforms the existing QODA-LB, Best-KFF, SPSO-TCS, and VMMISD strategies. In comparison, the SHIELD framework gains a computational efficiency of 20% under large workloads with existing techniques. The proposed SHIELD framework demonstrates higher throughput compared to the existing techniques across all tasks is shown in Figure 10. For tasks 100, 200, 300, and 400 it achieves throughput rates of 300, 290, 280, and 270 tasks respectively. Table 1 presents the experimental outcomes of the SHIELD framework.

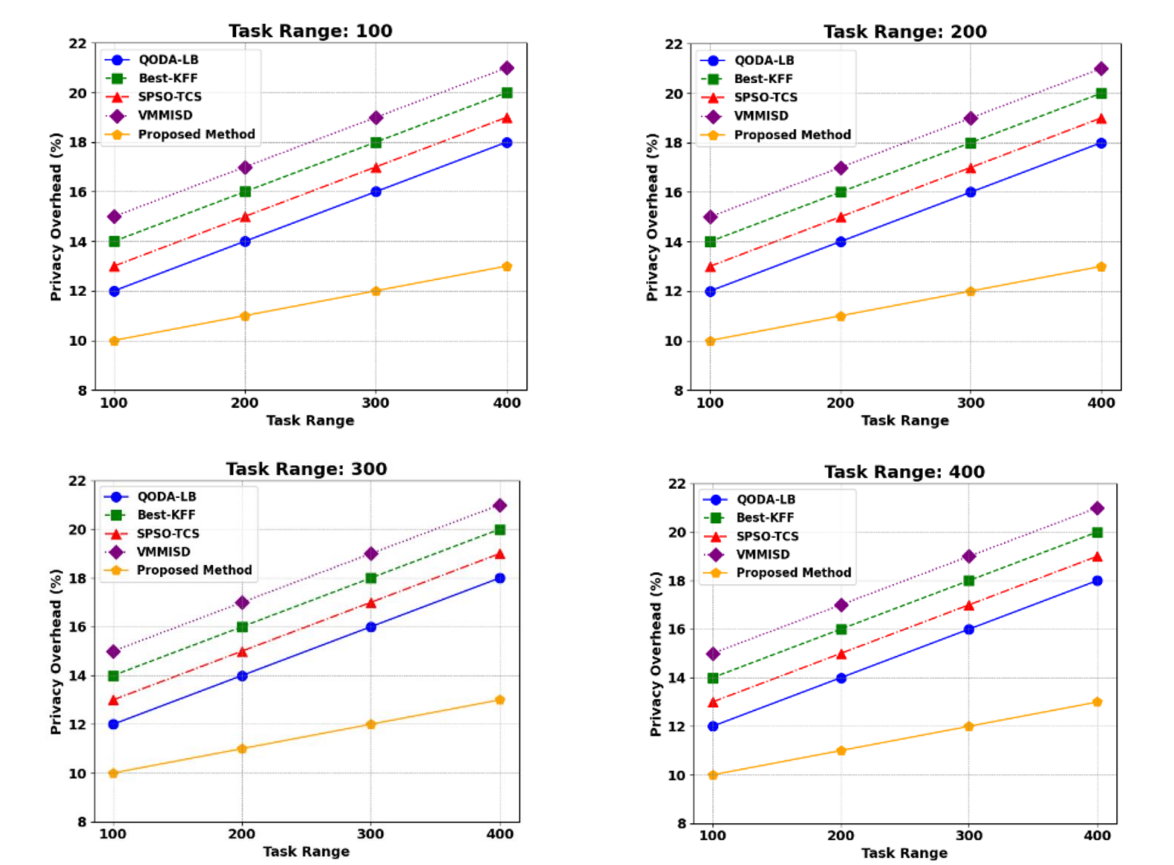


Figure 6. Analysis of privacy overhead

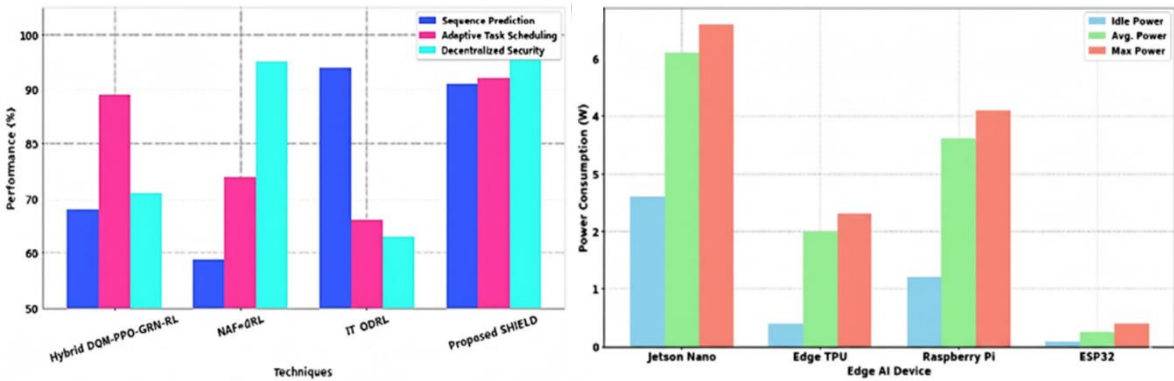


Figure 7. Performance comparison of AI models

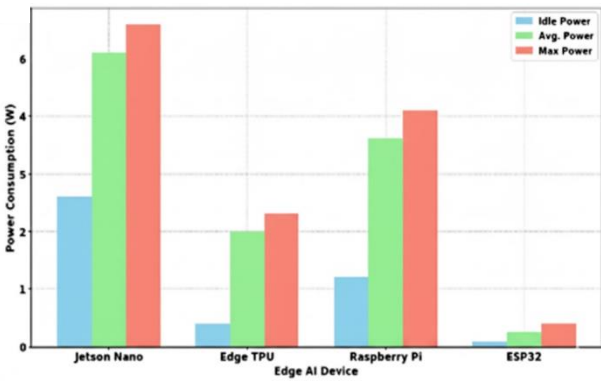


Figure 8. Power consumption of edge AI devices

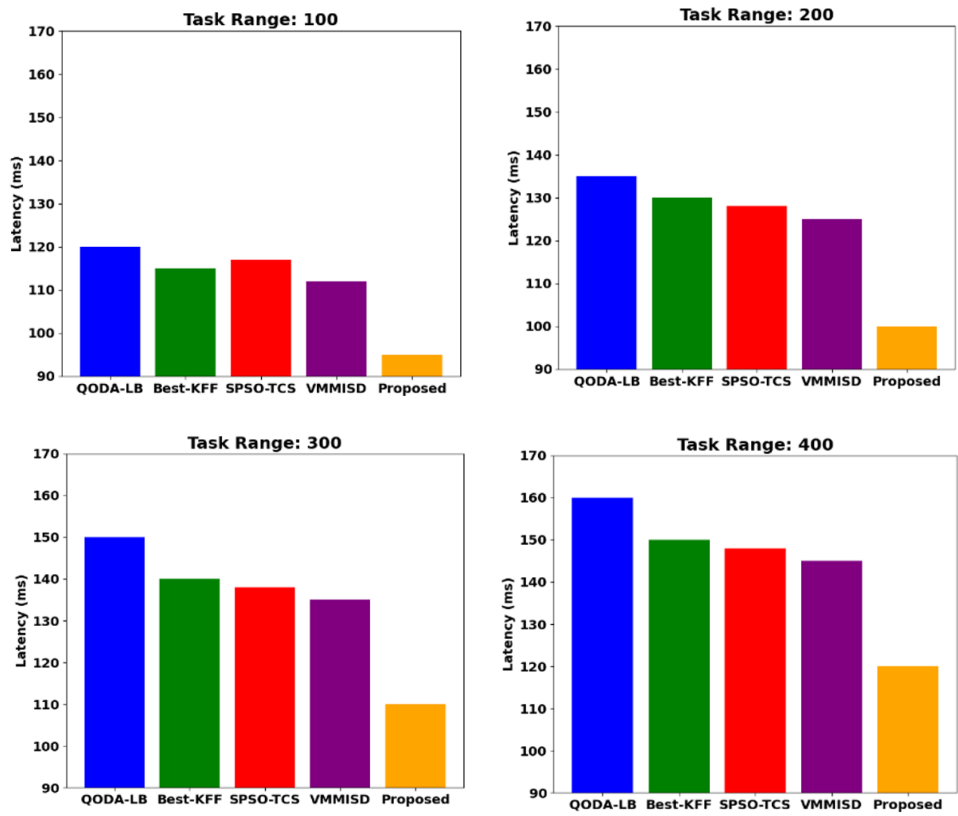


Figure 9. Analysis of latency

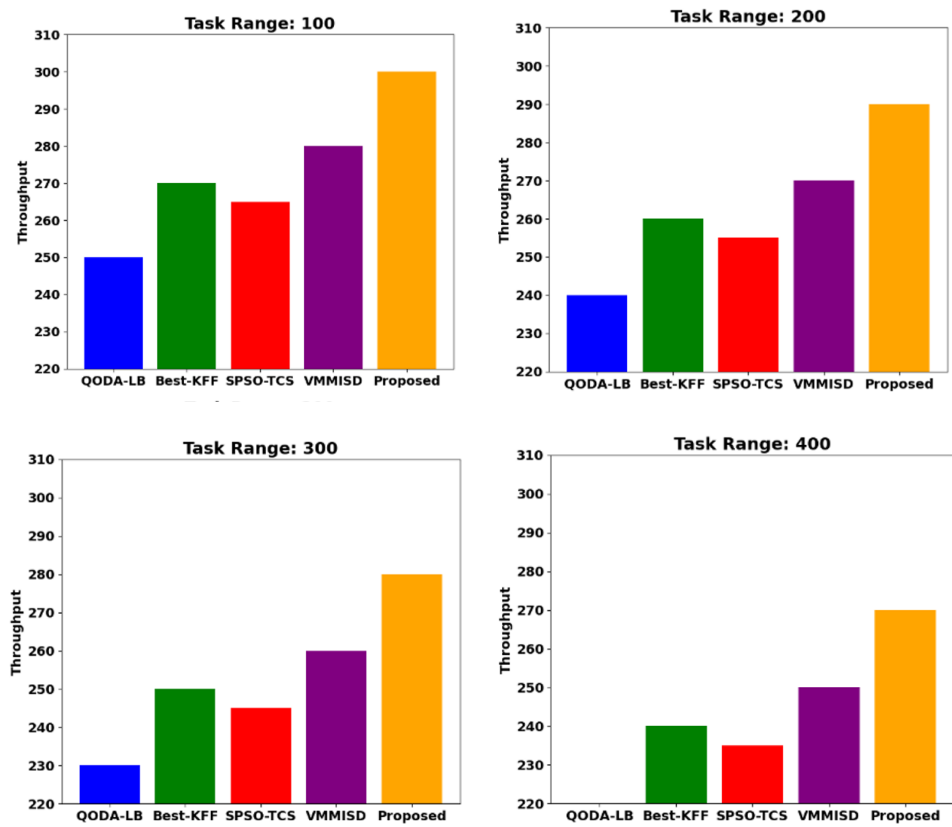


Figure 10. Analysis of throughput

Table 1. Experimental result comparison on various metrics

Metric	SHIELD	QODA-LB	Best-KFF	SPSO-TCS	VMMISD
Accuracy (%)	93.5%	85%	90%	88%	87%
Scalability (%)	86%	74%	76%	75%	78%
Resource utilization (%)	83%	70%	72%	71%	74%
Privacy overhead (%)	20%	16%	17%	18%	19%
Latency (ms)	249 ms	347 ms	332 ms	314 ms	293 ms
Throughput (Mbps)	270 Mbps	220 Mbps	240 Mbps	235 Mbps	250 Mbps
Power consumption (Watts)	24.3 Watts	41.2 Watts	39.1 Watts	36.7 Watts	34.9 Watts
Computational efficiency	74.3	61	64.5	63	66.2

4. DISCUSSION

In this research, a novel SHIELD framework was proposed for secure task scheduling in cloud resources. The SHIELD framework is assessed and compared against several parameters including scalability, resource utilization, privacy overhead, latency, throughput, and power consumption respectively. The comparison of scalability for the SHIELD framework is shown in Figure 4. In large scale IoT workloads, The SHIELD framework outperforms existing QODA-LB, Best-KFF, SPSO-TCS, and VMMISD approaches by maintaining the highest scores from 93% at 2000 tasks to 86% at 10,000 tasks respectively. The comparison of latency for the SHIELD framework is shown in Figure 9. During large IoT workloads, the SHIELD framework achieves the lowest latency of 97 ms for 2,000 tasks to 249 ms for 10,000 tasks for handling high task loads. Figure 7 depicts the performance assessment of the SHIELD model with previously developed AI models. The SHIELD method achieves decentralized security of 97%, adaptive task scheduling of 92%, and sequence prediction of 94% which outperforms the existing Hybrid DQN-PPO-GNN-RL, HAFedRL, and TF-DDRL techniques respectively. The comparison of power consumption for the SHIELD framework is represented in Figure 3(b). At higher workloads, the SHIELD framework achieves the lowest power consumption of 18.2W to 24.3W across all task ranges from 2,000 to 10,000 against the existing QODA-LB, Best-KFF, SPSO-TCS, and VMMISD techniques respectively. The comparison of power consumption of Edge AI Devices for energy efficient load balancing is depicted in Figure 8. The ESP32 device attains the lowest power consumption of 0.08 W, 0.25 W, and 0.4 W for idle power, average power, and maximum power for SHIELD respectively. In real-world deployment, the SHIELD model will provide superior performance under heterogeneous IoT devices for robust and scalable load balancing.

5. CONCLUSION

In this paper, a novel SHIELD framework is developed for secure task scheduling in IoT devices using hybrid DL networks. The SHIELD framework is simulated by using CloudSim7G. A comparative analysis was conducted with the existing techniques such as resource utilization, latency, throughput, privacy overhead, and scalability. In comparison, the SHIELD framework ensures a privacy overhead of 14%, whereas the QODA-LB, Best-KFF, SPSO-TCS, and VMMISD techniques achieve overheads of 10%, 11%, 12%, and 13%, respectively. In the future, the SHIELD framework will integrate an advanced blockchain methodology to hinder the most vulnerable adversarial attacks during load balancing. This integration will enhance the security and privacy of IoT-cloud interactions for secure load balancing. Also, the SHIELD framework will be deployed in Google Cloud, a real-world cloud platform that dynamically adjusts task scheduling based on changing IoT environments and resource availability.

ACKNOWLEDGMENTS

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

FUNDING INFORMATION

Not applicable.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Loga Priyadarshini	✓		✓		✓		✓		✓		✓	✓		
Kathirmalaiyan		✓		✓		✓		✓		✓				✓
Nithya Muthu														

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

INFORMED CONSENT

I certify that I have explained the nature and purpose of this study to the above-named individual, and I have discussed the potential benefits of this study participation. The questions the individual had about this study have been answered, and we will always be available to address future questions.

ETHICAL APPROVAL

My research guide reviewed and ethically approved this manuscript for publishing in this journal.

DATA AVAILABILITY

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.





REFERENCES

- [1] M. Shuaib *et al.*, "An optimized, dynamic, and efficient load-balancing framework for resource management in the internet of things (IoT) environment," *Electronics*, vol. 12, no. 5, p. 1104, Feb. 2023, doi: 10.3390/electronics12051104.
- [2] N. Mazumdar, A. Nag, and J. P. Singh, "Trust-based load-offloading protocol to reduce service delays in fog-computing-empowered IoT," *Computers & Electrical Engineering*, vol. 93, p. 107223, Jul. 2021, doi: 10.1016/j.compeleceng.2021.107223.
- [3] G. Thahniyath and M. Jayaprasad, "Secure and load balanced routing model for wireless sensor networks," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 7, pp. 4209–4218, 2022, doi: 10.1016/j.jksuci.2020.10.012.
- [4] D. K. Sah, T. N. Nguyen, K. Cengiz, B. Dumba, and V. Kumar, "Load-balance scheduling for intelligent sensors deployment in industrial internet of things," *Cluster Computing*, vol. 25, no. 3, pp. 1715–1727, 2022, doi: 10.1007/s10586-021-03316-1.
- [5] N. Agrawal, "Dynamic load balancing assisted optimized access control mechanism for edge-fog-cloud network in Internet of Things environment," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 21, p. 6440, 2021, doi: 10.1002/cpe.6440.
- [6] S. Fugkeaw, R. Prasad Gupta, and K. Worapaluk, "Secure and fine-grained access control with optimized revocation for outsourced IoT EHRs with adaptive load-sharing in fog-assisted cloud environment," *IEEE Access*, vol. 12, pp. 82753–82768, 2024, doi: 10.1109/ACCESS.2024.3412754.
- [7] V. Veeramachaneni, "Enhancing IoT security with dynamic reputation-based trust management," *Recent Innovations in Wireless Network Security*, vol. 7, no. 1, pp. 27–44, 2025.
- [8] K. Dhanushkodi, R. Kumar, P. Mittal, S. S. Das, N. N. S. Suryavenu, and K. Venkataramani, "Enhancing resource utilization and privacy in IoT data placement through fuzzy logic and PSO optimization," *Cluster Computing*, vol. 27, no. 9, pp. 12603–12626, 2024, doi: 10.1007/s10586-024-04542-z.
- [9] M. Jesi, A. Appathurai, M. Kumaran, and A. Kumar, "Load balancing in cloud computing via mayfly optimization algorithm," *Revue Roumaine des Sciences Techniques Serie Electrotechnique et Energetique*, vol. 69, no. 1, pp. 79–84, 2024, doi: 10.59277/RRST-EE.2024.1.14.
- [10] D. Champla, S. G. Ali, B. Muthukumar, and M. M. Sithik, "C-AVPSO: dynamic load balancing using African vulture particle swarm optimization," *International Journal of Data Science and Artificial Intelligence*, vol. 1, no. 2, pp. 1–9, 2023.
- [11] G. Sreetha and G. Santhiya, "SAFE-ACID: a novel security assessment framework for IoT intrusion detection via deep learning," *International Journal of Data Science and Artificial Intelligence*, vol. 2, no. 1, pp. 20–26, 2024.
- [12] S. R. Addula *et al.*, "Dynamic load balancing in cloud computing using hybrid kookaburra-pelican optimization algorithms," *2024 International Conference on Augmented Reality, Intelligent Systems, and Industrial Automation, ARIIA 2024*, vol. 2, no. 4, pp. 98–104, 2024, doi: 10.1109/ARIIA63345.2024.11051893.
- [13] C. S. Singh, S. Naaz, and G. Saranya, "IoT centric data protection using deep learning technique for preserving security and privacy in cloud," *International Journal of Data Science and Artificial Intelligence (IJDSAI)*, 2024.
- [14] A. Alwhelat, O. A. Fadare, F. Al-Turjman, and L. Ibrahim, "Adaptive load balancing in cloud computing using deep deterministic





- policy gradient (DDPG): a reinforcement learning approach,” in *Sustainable Civil Infrastructures*, vol. Part F4042, 2025, pp. 1067–1078.
- [15] V. Gowri and B. Baranidharan, “Adaptive probabilistic neural network based edge data center authentication for secure load balancing in fog computing,” *Applied Soft Computing*, vol. 169, p. 112567, Jan. 2025, doi: 10.1016/j.asoc.2024.112567.
- [16] M. Haris and S. Zubair, “Battle royale deep reinforcement learning algorithm for effective load balancing in cloud computing,” *Cluster Computing*, vol. 28, no. 1, p. 19, Feb. 2025, doi: 10.1007/s10586-024-04718-7.
- [17] K. K. Sowjanya and S. K. Mouleeswaran, “Enhancing cloud security with intelligent load balancing and malicious request classification,” *Cluster Computing*, vol. 28, no. 1, p. 41, Feb. 2025, doi: 10.1007/s10586-024-04754-3.
- [18] M. Ghorbani, N. Khaledian, and S. Moazzami, “ALBLA: an adaptive load balancing approach in edge-cloud networks utilizing learning automata,” *Computing*, vol. 107, no. 1, p. 34, 2025, doi: 10.1007/s00607-024-01380-0.
- [19] T. P. Latchoumi and L. Parthiban, “Quasi oppositional dragonfly algorithm for load balancing in cloud computing environment,” *Wireless Personal Communications*, vol. 122, no. 3, pp. 2639–2656, 2022, doi: 10.1007/s11277-021-09022-w.
- [20] A. Fathalla, K. Li, and A. Salah, “Best-KFF: a multi-objective preemptive resource allocation policy for cloud computing systems,” *Cluster Computing*, vol. 25, no. 1, pp. 321–336, 2022, doi: 10.1007/s10586-021-03407-z.
- [21] K. D. S. Devi, D. Sumathi, V. Vignesh, C. Anilkumar, K. Kataraki, and S. Balakrishnan, “CLOUD load balancing for storing the internet of things using deep load balancer with enhanced security,” *Measurement: Sensors*, vol. 28, p. 100818, 2023, doi: 10.1016/j.measen.2023.100818.
- [22] T. Saba, A. Rehman, K. Haseeb, T. Alam, and G. Jeon, “Cloud-edge load balancing distributed protocol for IoE services using swarm intelligence,” *Cluster Computing*, vol. 26, no. 5, pp. 2921–2931, 2023, doi: 10.1007/s10586-022-03916-5.
- [23] Z. Aghapour, S. Sharifian, and H. Taheri, “Task offloading and resource allocation algorithm based on deep reinforcement learning for distributed AI execution tasks in IoT edge computing environments,” *Computer Networks*, vol. 223, p. 109577, 2023, doi: 10.1016/j.comnet.2023.109577.
- [24] M. Menaka and K. S. S. Kumar, “Supportive particle swarm optimization with time-conscious scheduling (SPSO-TCS) algorithm in cloud computing for optimized load balancing,” *International Journal of Cognitive Computing in Engineering*, vol. 5, pp. 192–198, 2024, doi: 10.1016/j.ijcce.2024.05.002.
- [25] M. G. Brahman and R. V. Anand, “VMMISD: An efficient load balancing model for virtual machine migrations via fused metaheuristics with iterative security measures and deep learning optimizations,” *IEEE Access*, vol. 12, pp. 39351–39374, 2024, doi: 10.1109/ACCESS.2024.3373465.

BIOGRAPHIES OF AUTHORS



Loga Priyadarshini Kathirmalaiyan     is a Ph.D.(IT) scholar at Vinayaka Mission Kirupananda Variyar Engineering College, Vinayaka Mission Research Foundation (DU), Salem, Tamil Nadu. She did her M. Tech Information Technology at Sona College of Engineering, Anna University, Chennai. She did her B. Tech Information Technology at Paavai Engineering College, Anna University, Chennai. Her areas of interest include cloud computing, internet of things. She can be contacted at kavipriyas13@gmail.com.



Nithya Muthu     is a professor at Department of Computer Science and Engineering, Vinayaka Mission Kirupananda Variyar Engineering College, Vinayaka Mission Research Foundation (DU), Salem, Tamil Nadu. She did her Doctorate in Computer Science and Engineering from Govt College of Engineering, and her ME in Computer Science and Engineering at Sona College of Technology, Anna University, Chennai. She had 21 years of academic experience with merely 64 publications from reputed, peer-reviewed National and International Journals. Her areas of interest include artificial intelligence, grid computing, pattern recognition and machine learning. Under the guidance of Dr. Nithya M, 6 research scholars have completed their doctorates. She can be contacted at nithyam@vmkvec.edu.in.