ISSN: 2182-2069 / E-ISSN: 2182-2077

An Energy Efficient Secure Multi-Objective Workflow Scheduling in Blockchain-Assisted Cloud using a Hybrid Deep Learning Approach

G. Madhumala[1*], and Dr.R. Suma[2]

[1*]Research Scholar, Department of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Sri Siddhartha Academy of Higher Education, Tumkur, Karnataka, India.
madhumalag@ssit.edu.in, https://orcid.org/0000-0003-0731-7871

[2]Professor, Department of Information Science and Engineering, Sri Siddhartha Institute of Technology, Sri Siddhartha Academy of Higher Education, Tumkur, Karnataka, India.
sumar@ssit.edu.in, https://orcid.org/0009-0001-9510-6291

Received: January 24, 2026; Revised: March 09, 2026; Accepted: April 15, 2026; Published: May 29, 2026

Abstract

Blockchain-assisted cloud computing has emerged as a promising approach for ensuring data integrity and transparency; however, workflow scheduling within such environments remains complex due to resource heterogeneity, task dependencies, and latency constraints. This paper proposes a blockchain-based Dense Resolution Dilated Attention Forward Harmonic Network (Blockchain_DRDAFHNet) for secure, energy-efficient, multi-objective workflow scheduling in the cloud. The system model integrates a Task Pool Coordinator with blockchain miners and Virtual Machines (VMs) to anonymize and execute tasks securely via smart contracts. Scheduling is performed using two categories of parameters: VM parameters (CPU and bandwidth utilization) and task parameters (resource utilization, actual task running time, memory, makespan equivalent of total cost, reliability, task priority, Earliest Start Time, Earliest Finish Time, task length, trust, security level, security overhead, encryption speed, predicted energy via EA-LSTM, and secured data size). The proposed DRDAFHNet integrates a Dense Resolution Network (DRNet) with a Pyramid Dilated Attention Network (PDAN), with layers refined through forward harmonic analysis. Experiments are conducted in CloudSim using Python 3.9.11 with TensorFlow/Keras, across configurations of 10 PM/20 VM, 10 PM/40 VM, and 15 PM/50 VM. Under the 15 PM/50 VM configuration, Blockchain_DRDAFHNet achieves the best residual energy of 0.831 J, resource utilization of 0.576, and makespan of 0.546, outperforming seven existing methods including ALPSO, Actor-Critic, CGA, 2SD-GAT, HGALO-GOA, DNN-IGW-HHO, and RL-MOTS. These results demonstrate the model's robustness, security integration, and suitability for large-scale dynamic cloud scheduling.

Keywords: Work Flow Scheduling, Virtual Machine, Evolutionary Attention-Based Long Short-Term Memory, Dense Resolution Network, Pyramid Dilated Attention Network.

Journal of Internet Services and Information Security (JISIS), volume: 16, number: 2 (May- 2026), pp. 503-527.
DOI: *10.58346/JISIS.2026.I2.032*

*Corresponding author: Research Scholar, Department of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Sri Siddhartha Academy of Higher Education, Tumkur, Karnataka,India.

1 Introduction

Blockchain is specified as a distributed model, which includes a massive volume of data. Moreover, blockchain is a type of decentralized system, which eliminates the requirement of an intermediary third-party authority. Each node accesses the details of every transaction; hence it makes the system as more transparent than centralized communications. The node's presence in the blockchain is unspecified; Therefore, it creates more security in every transaction (Yli-Huumo et al., 2016). The cryptographic primitives as well as protocols like hash functions, and digital signatures provide security support in the blockchain. These primitives confirm that the transactions recorded in the ledger are authenticated, non-repudiated, and integrity-protected (Guo & Yu, 2022; Dasgupta et al., 2019). Furthermore, nodes are used as the major devices or participants, which creates the blockchain technology. In addition, blockchain generates a decentralized system, where all network nodes participate in the verification and validation of data. Moreover, the data in the blockchain is tamper-evident; hence, the blockchain secures the data and minimizes privacy concerns (Murthy et al., 2020).

Recently, cloud technology has emerged as a reliable and efficient solution in both industry and academia. Cloud systems reduce user-side processing complexity and support large-scale applications requiring significant computational resources. Cloud computing operates on a utility model, enabling users to share, access, and transact data anytime and anywhere. It dynamically delivers applications across distributed computational resources. Workflow scheduling is a key component of cloud workflow management, and effective task scheduling significantly enhances system performance (Murthy et al., 2020). Cloud systems also provide high-performance resources over the Internet and are widely used in large-scale applications for efficient network resource management (Meena et al., 2016).

The cloud workflow system enables automation of large-scale distributed e-science and e-business applications, making robust workflow scheduling essential for assigning tasks to virtual resources (Adhikari et al., 2019). Task scheduling improves user experience, energy efficiency, and resource utilization (Mikram et al., 2024). Cloud architecture uses virtualization, where a physical server is divided into isolated VMs running independently. Scheduling is categorized into static and dynamic approaches; static scheduling relies on prior knowledge of resources, while dynamic scheduling adapts to workload changes (Chandrasiri & Meedeniya, 2025). Due to increasing complexity, ML and DL techniques are widely used for resource management and scheduling. ML learns decision rules using reinforcement and supervised learning, while DL and hybrid models better capture usage patterns, reducing cost and delay in cloud workflow scheduling (Swarup et al., 2021; Bal et al., 2022).

In this work, Blockchain_DRDAFHNet-based workflow scheduling in the cloud is developed. The blockchain-assisted cloud is simulated, and workflow scheduling is performed by considering the VM parameters like CPU, and Bandwidth utilization, and task parameters like resource utilization, actual task running time, makespan equivalent of total cost, memory, trust, reliability, task priority, predicted energy, EST, EFT, task length, security level, security overhead, encryption speed, and secured data size. The Evolutionary Attention-based Long-Short Term Memory (EA-LSTM) predicts the energy, and the DRDAFHNet is employed for multi-objective workflow scheduling.

The main contribution of the paper is illustrated below:

- **Blockchain_DRDAFHNet- multi-objective workflow scheduling in the cloud:** Workflow scheduling in cloud is crucial to improve energy efficiency, resource usage, and user experience. The workflow scheduling is performed by the DRDAFHNet using VM and task parameters.

Moreover, the combination of DRNet with PDAN develops the DRDAFHNet, where the forward harmonic analysis modifies its layers.

The remaining part of the paper includes the following sections: The motivation and review of existing methods are described in Section 2. Section 3 shows the cloud with a blockchain model along with a schematic description, and Section 4 describes the Blockchain_DRDAFHNet-based workflow scheduling. The experimental analysis of Blockchain_DRDAFHNet is described in Section 5, and a conclusion is given in Section 6.

2 Literature survey

Workflow scheduling in cloud computing has been widely studied to optimize objectives such as energy consumption, execution time, and system efficiency. Existing approaches are mainly categorized into metaheuristic-based methods, which use optimization heuristics, and ML/DL-based methods, which improve scheduling through predictive and intelligent decision-making. This survey reviews key works in both categories, highlighting their techniques, objectives, and limitations, and identifies research gaps that motivate the proposed Blockchain_DRDAFHNet model.

(i) Metaheuristic-Based Workflow Scheduling

Thekkepuryil, et al. (2021) devised the Ant-Lion Particle Swarm Optimization (ALPSO)-based workflow scheduling in the cloud. This model consumed less energy to optimize the scheduling. Still, this model failed to use various Quality of Service (QoS) parameters and trust management to perform the tasks. Ali & Ali, (2023) developed the Catastrophic Genetic Algorithm (CGA) to improve workflow scheduling in the cloud. This model consumed less processing period for completing the task. Nevertheless, it failed to detect the finest solution to improve the convergence and decrease the number of iterations. Mohammadzadeh, et al. (2021) devised the hybrid multi-objective Greedy Antlion Optimization (ALO) with Grasshopper Optimization Algorithm (GOA) (HGALO-GOA) to solve the issues of workflow scheduling. In this model, the consumed energy and the makespan were less. Nevertheless, this method was not effective for practical applications.

Datta & Thippanna, devised the Gravitational Search Algorithm named GSACLD for task scheduling in the Cloud. In the model, the Gravitational Search Algorithm (GSA) solved the scheduling issues and diminished the scheduling times. Still, it failed to consider swarm intelligence methods, and advanced fitness functions to reduce the cost. Zhang & Wang, (2024) developed the Enhanced Whale Optimization Algorithm (EWOA) in task scheduling. This model offered better outcomes in real-world applications while considering several conflicting objectives. However, the efficiency of the model was not validated in more complex and larger cloud environments. Moreover, the adaptability of the model was not investigated in dynamic scenarios (Chaudhary & Kumar, 2018).

(ii) ML / DL-Based Scheduling

Dong et al., (2021) devised the Actor-Critic architecture for scheduling the workflow in a cloud environment. This approach diminished the execution period in workflow scheduling. However, this model failed to solve the dynamic complexities for obtaining effective scheduling. Shakkeera, (2025) devised the Federated Learning with Blockchain Technology (FLBCT) in Microservice-based Mobile Cloud Computing Applications (MSCMCC). This model was used for handling extensive sequential processes. However, ML-based task scheduling and decision-making were not used to improve

scheduling efficiency. Badri et al., (2023) developed the Convolutional neural network optimized modified butterfly optimization (CNN-MBO) algorithm in task scheduling. The model attained lower energy consumption, execution time, and response time. Nevertheless, the QoS parameters were not considered to minimize the task arrival time. Table 1 shows the comparison of the existing models with the proposed framework.

Table 1: Comparison of workflow scheduling approaches in cloud computing

Reference Number	Methods	Objective	Limitation
Thekkepuryil et al., (2021)	ALPSO	Reduce energy consumption	Does not consider QoS or trust
Dong et al., (2021)	Actor-Critic architecture	Minimize execution time	Cannot handle dynamic workflow complexity.
Ali & Ali, (2023)	CGA	Reduce processing time	Poor convergence and iteration efficiency
Mohammadzadeh et al., (2021)	HGALO-GOA	Energy efficiency, makespan reduction	Limited practical applicability
Chaudhary & Kumar, (2018)	GSACLD	Reduce scheduling time	Lacks swarm intelligence and cost optimization
(Shakkeera, 2025)	FLBCT	Handle sequential workflows	ML-based task scheduling efficiency not addressed
Badri et al., (2023)	CNN-MBO	Reduce energy, execution, response time	QoS not fully integrated
Zhang & Wang, (2024)	EWOA	Multi-objective optimization	Not validated in large-scale dynamic environments
Proposed Model	Blockchain_DRDAFHNet	Handles dynamic tasks, heterogeneous VMs, and latency-sensitive workflows; integrates trust, security, and multi-objective optimization	Not yet validated in large-scale real-world cloud environments

Existing cloud workflow scheduling methods mainly optimize limited objectives like energy, time, or makespan, while ignoring key factors such as reliability, trust, security, and resource efficiency. They also lack real-time adaptability for dynamic workflows, heterogeneous VMs, and latency-sensitive tasks, and rarely integrate QoS with blockchain-based security. Moreover, most approaches are not validated in large-scale environments, limiting real-world applicability. To address these gaps, Blockchain_DRDAFHNet integrates DRNet and PDAN within a blockchain framework to enable secure, adaptive, and multi-objective workflow scheduling in cloud computing.

3 System Model

In figure 1 illustrates the cloud–blockchain system model where a fog node ensures end-user privacy and manages communication between blockchain miners, the cloud, and users. The Task Pool Coordinator (TPC) collects tasks and resource information from end users and virtual resources, assigns anonymized task IDs, and considers parameters such as task length, load balance, and CPU capacity for scheduling. The TPC generates smart contracts, and miners propose scheduling solutions based on cost

and execution time. After evaluating multiple iterations, the TPC selects the optimal schedule and forwards it to the cloud for execution. The results are then returned to users, while smart contract data is stored on-chain for future analysis, ensuring secure and privacy-preserving task management.

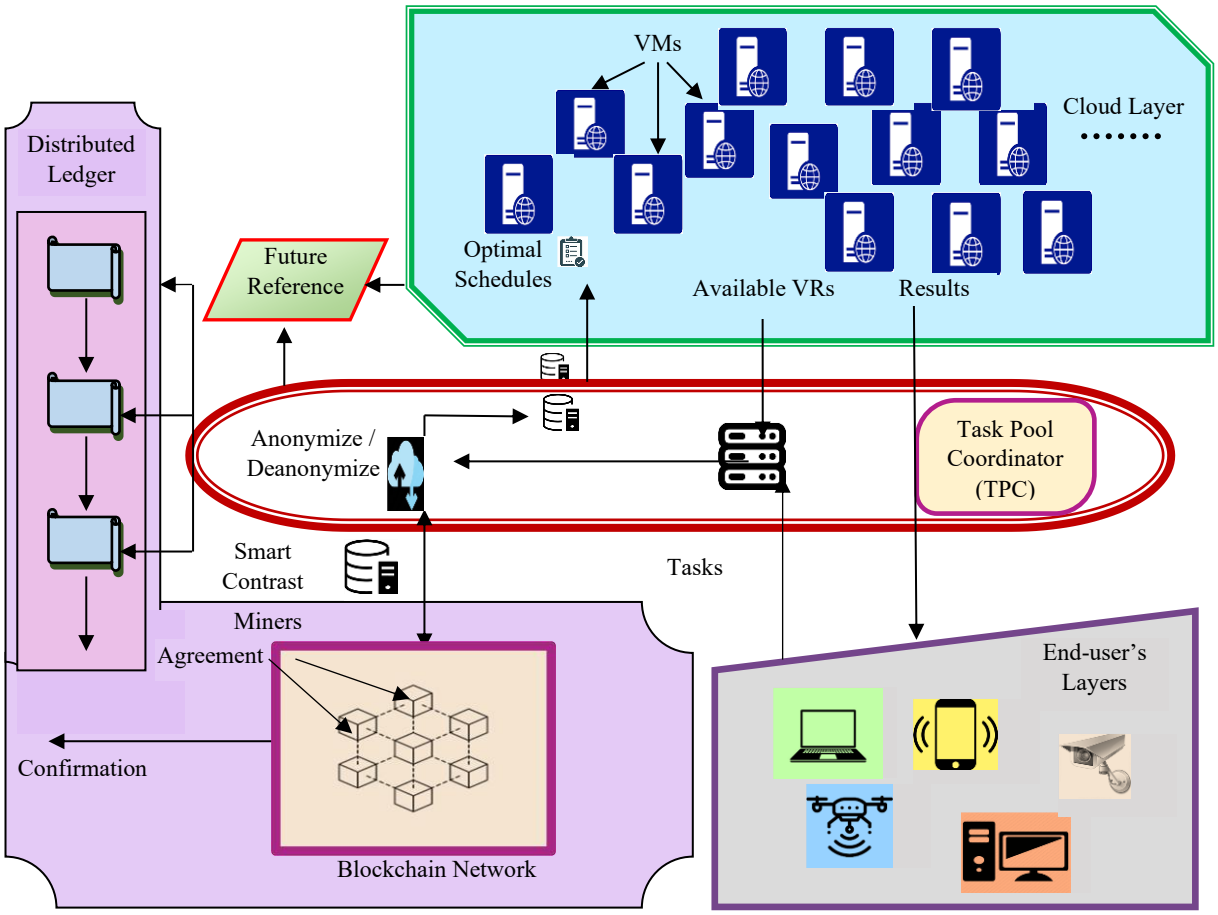

Figure 1: System model for the cloud with blockchain

4 Developed Dense Resolution Dilated Attention Forward Harmonic Network-based multi-objective workflow scheduling in Blockchain-Cloud

This paper develops the Blockchain_DRDAFHNet-based multi-objective workflow scheduling in a blockchain-aided cloud. Initially, the Blockchain-Cloud model is simulated, and workflow scheduling is conducted by considering VM and task parameters. Moreover, the VM parameters like CPU (Thekkepuryil et al., 2021), and bandwidth utilization (Thekkepuryil et al., 2021) as well as the task parameters such as task priority (Choudhary et al., 2018), EST (Mazrekaj et al., 2019), EFT (Mazrekaj et al., 2019), Task length (Ge et al., 2017), actual task running time, predicted energy using EA-LSTM (Li et al., 2019), trust (Naik et al., 2021), and memory capacity (Thekkepuryil et al., 2021), Makespan equivalent of total cost, reliability (Saeedi et al., 2020), resource utilization, security overhead (Huang et al., 2019), security level (Huang et al., 2019), encryption speed (Huang et al., 2019), and secure data size (Huang et al., 2019) are utilized. Moreover, workflow scheduling is performed using the

DRDAFHNet. Figure 2 depicts the schematic view of Blockchain_DRDAFHNet-based workflow scheduling.

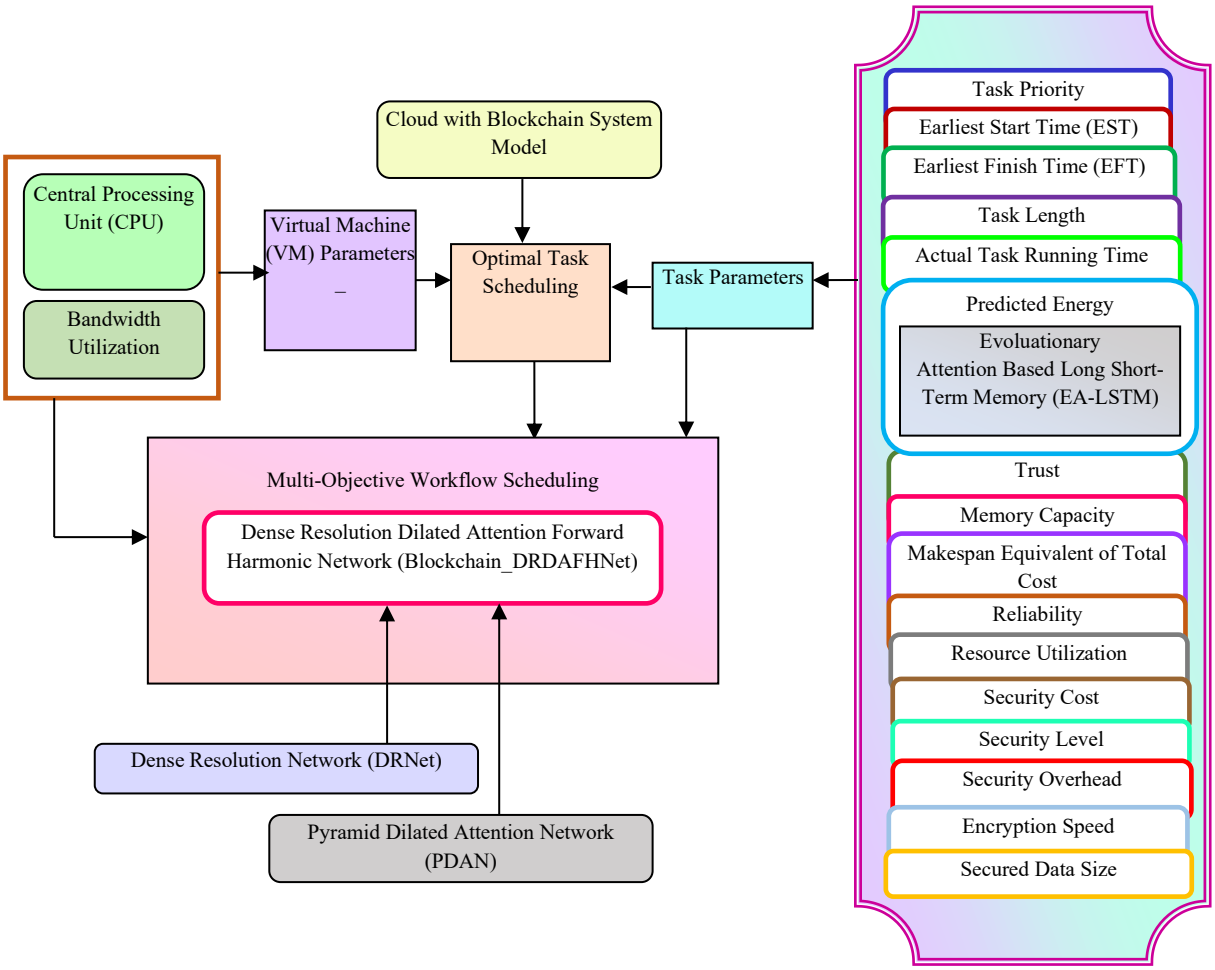

Figure 2: Block diagram of blockchain_DRDAFHNet-based workflow scheduling in blockchain-aided cloud

4.1 Blockchain Model

4.2 Optimal workflow scheduling based on DRDAFHNet

Workflow scheduling is needed for assigning computational tasks across the cloud servers. The task and VM parameters are considered as the major elements in optimal workflow scheduling. Here, the devised DRDAFHNet is utilized for scheduling the task to the available resources.

4.2.1 VM parameters

VM parameters represent the pattern to alter the features of the VM in a virtualized setting. These parameters optimize the performance, security, and functionality of VMs. It is needed for optimizing resource allocation and providing efficient scheduling. Here, the bandwidth utilization, and CPU are considered as the major VM parameters for workflow scheduling.

a) CPU

CPU (Thekkepuryil et al., 2021) is employed as an important parameter in task scheduling, and it schedules the tasks in an organized way. The CPU is responsible for processing data and executing instructions, and the scheduling algorithms are utilized to manage the allocation of CPU time to various processes. Moreover, the CPU parameter is denoted by υ_1.

b) Bandwidth Utilization

The number of resources needed for executing the task is denoted by bandwidth. Moreover, bandwidth utilization (Thekkepuryil et al., 2021) is indicated as the proportion of available bandwidth utilized at a given period.

$$\beta_{ut} = \frac{\beta_{ac}}{\beta_{oc}} \times 100 \quad (1)$$

Here in equation (1), the actually utilized bandwidth is specified by $BW_{ac}$and β_{oc} indicates the overall bandwidth capacity. Furthermore, the bandwidth utilization is symbolized by υ_2.

The dimension of VM parameters is portrayed, where υ specifies the VM parameters.

$$\upsilon = \{\upsilon_1, \upsilon_2\} \quad (2)$$

Where in equation (2), the CPU is denoted by υ_1 and bandwidth utilization is indicated by υ_2.

4.2.2 Task Parameters

Task scheduling is a significant factor in allocating tasks to the resources. Furthermore, it is defined as the selection of resources to complete tasks with less waiting and execution period. The task parameters with their dimension. Moreover, the following task parameters are used for workflow scheduling.

a) Task Priority

Task priority (Choudhary et al., 2018) is considered as an essential metric for scheduling tasks because a set of tasks is generated with low to high priority values. Here, the communication time and average execution time are used to compute task priority. The priority of the task $t_\tau$during this period $\chi$is indicated by $Pr_{t_\tau}(\chi)$ is shown in equation (3).

$$Pr_{t_\tau}(\chi) = \frac{ee_{t_\tau}}{vt(tv_{t_\tau}, ee_{t_\tau}, tv_{Max})} * \frac{1}{\left(ee_{t_\tau} + rf * (\chi - at_{t_\tau})\right)} + \delta \quad (3)$$

The current time is specified by χ, and the value level of $t_\tau$is portrayed by δ. The transformation function is denoted by vt. The estimated time for executing the task$t_\tau$is specified by ee_{t_τ} , and the task value is portrayed by tv_{t_τ}. The maximum task relative value for the task$t_\tau$is indicated by tv_{Max}. Furthermore, arrival time is denoted by at_{t_τ} and rf implies the regulatory factor. Moreover, the task priority is specified by $\Re_1$

b) Earliest Start Time

EST (Mazrekaj et al., 2019) specifies the period at which the tasks are initiated without violating any constraints or dependencies. It is computed by the durations of tasks, and numerous dependencies. The EST of the processor H_ε with respect to the node $a_\phi$is given in equation (4),

$$EST(a_{\phi}, H_{\varepsilon}) = \max\left[\, t_{ea}(H_{\varepsilon}),\ \max_{a_{\Omega} \in Pre(a_{\phi})} (t_{AFT}(a_{\Omega}) + \mathrm{Y}_{\Omega,\phi})\right] \quad (4)$$

The earliest time at which the processor executes the task is termed by t_{ea}. The term $Pr\,e\left(a_{\varphi}\right)$ specifies the set of tasks, and the cost of communication is selected by $Pr\,e\left(a_{\varphi}\right)$. The term $\Re_2$indicates the EST.

c) Earliest Finish Time

EFT (Mazrekaj et al., 2019) indicates the earlier time at which a specific task is scheduled. The EFT of the processor H_{ε} with a node $a_{\phi}$is expressed in equation (5),

$$EFT\left(a_{\varphi}, H_{\varepsilon}\right) = EST\left(a_{\varphi}, H_{\varepsilon}\right) + \left(ee_{t_{\tau}}\right)_{\varphi,\varepsilon} \quad (5)$$

Here, the estimated executing time for task completion is illustrated as $ee_{t_{\tau}}$. The EFT is denoted by $\Re_3$.

d) Task Length

Task length (Ge et al., 2017) is specified as the time or effort required to finish a certain task. Moreover, task length is based on the level of interruptions, and resource availability. The execution time matrix $eT(b,c)$ on the VM $'c'$is given in equation (6),

$$eT(b,c) = \frac{t_{len}(b,c)}{VM_{ca}(b,c)} \quad (6)$$

The task length is specified by $t_{len}(b,c)$, and the calculation ability for the VM is specified by $VM_{ca}(b,c)$. The term $\Re_4$symbolizes the task length.

e) Actual Task Running Time

The overall time taken by a task to complete its execution is the actual task running time is shown in equation (7)

$$\Re_5 = \frac{rt}{Compute} \quad (7)$$

Here, the refer time is specified by rt and the actual task running time is denoted by $\Re_5$.

f) Predicted Energy

In task scheduling, energy prediction is used to estimate the energy consumption of energy needed to optimize the resource allocation. Here, the energy required to run the task $(Eg)$is fed to the EA-LSTM for predicting the energy Eg_{Pr}.

i) Architecture of EA-LSTM

EA-LSTM (Li et al., 2019) is formed by adding the attention layer to the basic LSTMs. The structure of EA-LSTM is depicted in figure 3, where the energy $Eg$is fed as the input. The LSTMs are a type of RNN, where the gated structure contains the input, output, and forget gates. The LSTMs are used for memorizing and forgetting the values, and the attained weight in the EA-LSTM is given in equation (8),

$$\omega = \left(\omega^{1}, \omega^{2}, \ldots, \omega^{A}\right) \quad (8)$$

From the attention weights, the input sampling is expressed as equation (9),

$$D_z = \left(d_z^1\omega^1, d_z^2\omega^2, \ldots, d_z^A\omega^A\right) \quad (9)$$

In the LSTM networks, $\tilde{D} = \left(\tilde{D}_1, \tilde{D}_z, \ldots, \tilde{D}_\kappa\right)$is fed as the input. The nonlinear mapping function is learned using the following equation (10) to (14),

$$y^z = \gamma\left(\omega_{dy}\tilde{D}_z + \omega_{\hbar y}\hbar^{z-1} + \omega_{\eta y}\eta^{z-1} + \lambda_y\right) \quad (10)$$

$$\mu^z = \gamma\left(\omega_{d\mu}\tilde{D}_z + \omega_{\hbar\mu}\hbar^{z-1} + \omega_{\eta\mu}\eta^{z-1} + \lambda_\mu\right) \quad (11)$$

$$\eta^z = \mu^z\eta^{z-1} + y^z\, tanh\left(\omega_{d\eta}\tilde{D}_z + \omega_{\hbar\eta}\hbar^{z-1} + +\lambda_\eta\right) \quad (12)$$

$$\vartheta^z = \gamma\left(\omega_{d\vartheta}\tilde{D}_z + \omega_{\hbar\vartheta}\hbar^{z-1} + \omega_{\vartheta y}\eta^{z-1} + \lambda_\vartheta\right) \quad (13)$$

$$\hbar^z = \vartheta^z\, tanh(\eta^z) \quad (14)$$

Here, $\gamma(.)$ implies the sigmoid activation function, ω specifies the connection weights among two cells. The term y^z indicates the input gate state and μ^zdenotes the forget gate. Furthermore, η^zimplies the cell state, ϑ^zdenotes the output gate, ϑ^zspecifies the hidden layer of the present state, and the output vector $\hbar^{z-1}$ yields the predicted output Eg_{Pr}. Furthermore, the term $\Re_6$shows the predicted energy.

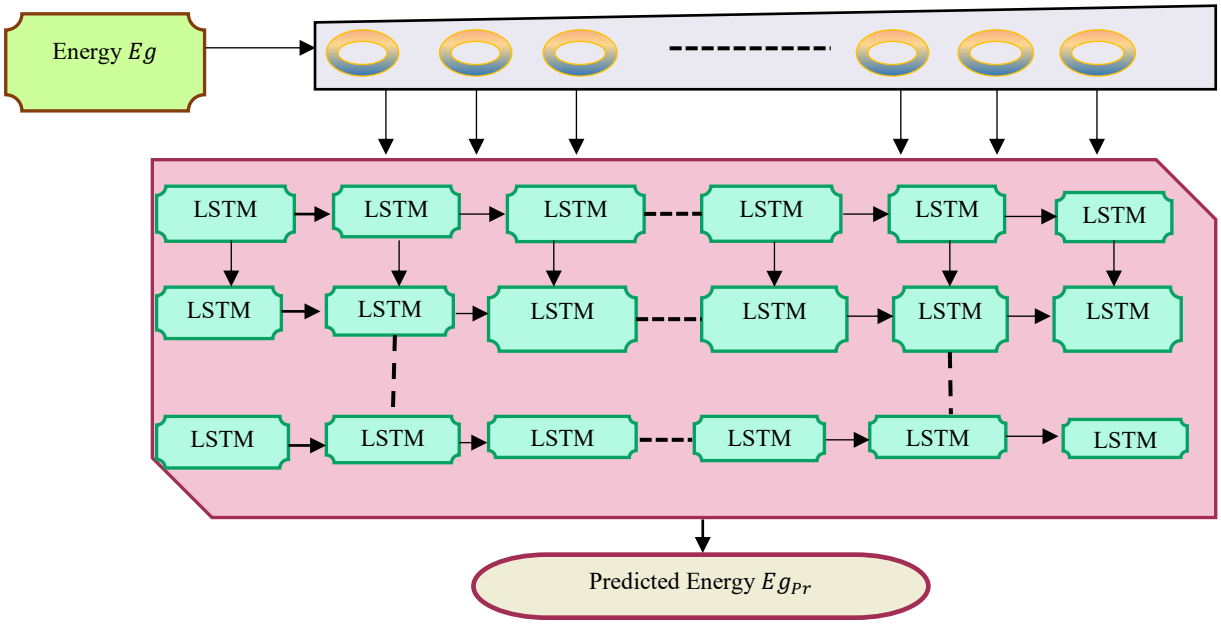

Figure 3: Structure of EA-LSTM

g) Trust

In task scheduling, trust (Naik et al., 2021) is indicated as the level of reliability, integrity, and security of resources for assigning and executing the tasks. Based on the trust factor, tasks are scheduled with high reliability and security. The trust parameter $\Re_7$is expressed as equation (15),

$$\Re_7 = tv_{t(\Theta=1,2,\ldots,\Xi)} = \frac{F*\aleph}{F} \quad (15)$$

Where, tv specifies trust for individual tasks, and $\aleph$indicates the degree of the discrete trust. Moreover, Ndenotes trust in individual tasks.

$$N = \pi\left(1 - k_{pt}\right) \tag{16}$$

Here in equation (16),$k$is given by,

$$k = \frac{1}{1+pt''} \tag{17}$$

In equation (17), the processing time of the task is specified by pt''.

h) Memory Capacity

Memory Capacity (Thekkepuryil et al., 2021) refers to the available memory for storing and executing tasks. Moreover, it ensures that the tasks are scheduled with sufficient memory to prevent delays and failures. The memory capacity (Ca_{Mem}) is given in equation (18),

$$Ca_{Mem} = \frac{Me_{ut}}{Me_{to}} \tag{18}$$

where, Me_{ut}shows the utilized memory Me_{to}implies the whole memory. Furthermore, the memory capacity is symbolized by $\Re_8$

i) Makespan Equivalent of the Total Cost

The Makespan equivalent of the total cost (Choudhary et al., 2018) indicates the total execution period to complete a task. It is based on the booting period of the VM, data execution, and shutdown period of the VM, as well as the transmission time between two VMs. The term $\Re_9$ shows the Makespan equivalent of the total cost is shown in equation (19).

$$\Re_9 = l \times co_{to} \tag{19}$$

Here,$l$represents the cost of the makespan equivalence parameter, and the total cost is indicated by co_{to}.

$$co_{to} = \sum_{b=1}^{o} \sum_{c=1}^{p} M_{s,q} \times co(t_b, \upsilon_c) \tag{20}$$

In equation (20), the term $M_{s,q}$ defines Boolean variable and $co(t_b, \upsilon_c)$ symbolizes the execution cost of task t_b on the $VM\ \upsilon_c$.

j) Reliability

Reliability (Saeedi et al., 2020) is termed as the capacity for completing each task because cloud computing is a heterogeneous computing device. Hence, errors are unavoidable, which generate the Poisson function, and affect the energy consumption. Therefore, increased node reliability is used for attaining effective scheduling. Moreover, the reliability $(\Re_{10})$is formulated in equation (21),

$$\Re_{10} = \prod_{o}^{b=1} Re_{bo} \tag{21}$$

Here, the reliability of $'o'$count of the task is denoted by Re_{bo}.

k) Resource Utilization

Resource Utilization is referred as the percentage of accessible computing resources to execute the task in equation (22).

$$\Re_{11} = \forall f_r \in N_O \begin{cases} \sum_{\forall se_r}^{se_O} RAM^{rqd}(se_r)^* Q_{rs} \leq RAMavailable(f_r) \\ \sum_{\forall se_r}^{se_O} CPU^{rqd}(se_r)^* Q_{rs} \leq CPUavailable(f_r) \\ \sum_{\forall se_r}^{se_O} Storage^{rqd}(se_r{}^*)^* Q_{rs} \leq storageavailable(f_r) \end{cases} \quad (22)$$

where, sespecifies services, and $N_O$represents the cloud nodes. The resource utilization is indicated by $\Re_{11}$.

l) Security Level

Security level (Huang et al., 2019) is indicated as the degree of security needed to execute the task without any unauthorized access, malicious activities, or data breaches. The security level ($\Re_{12}$)is formulated in equation (23),

$$\Re_{12} = SL(ca_v^u) = \frac{co(ca_v^u, cpu_v^x, pf_v^x, sd_\tau)}{co\left(ca_{cf}^u, cpu_v^x, pf_v^x, sd_\tau\right)} \quad (23)$$

where, sd_τimplies the secure data size, which is equal to 100. The count of processor cores is indicated by cpu_v^x, and the processor frequency of the VM is denoted by pf_v^x. Moreover, the u^{th}cryptographic algorithm for the v^{th}security service is indicated by ca_v^u, and the confidentiality service is portrayed bycf.

m) Security Overhead

Security overhead (Huang et al., 2019) is indicated as the time needed to execute the security services. Moreover, it improves the security of resources based on authentication, encryption, access control, and trust verification. The security overhead ($\Re_{13}$)is given in equation (24),

$$\Re_{13} = co(ca_v^u, cpu_v^x, pf_v^x, sd_\tau) = \frac{co(ca_v^u, cpu_v^x, pf_v^x, sd_\tau)}{cpu_v^x} \quad (24)$$

n) Encryption Speed

Encryption Speed (Huang et al., 2019) is defined as the rate at which encryption algorithms process task-related data before its execution. It directly impacts the efficiency of task scheduling, because the encryption creates computational overhead that affects the resource utilization and response time. The mathematical expression for Encryption Speed is denoted as ($\Re_{14}$) is shown in equation (25).

$$\Re_{14} = sp(ca_v^u) = \frac{sd_\tau}{co(ca_v^u, cpu_v^x, pf_v^x, sd_\tau)} \quad (25)$$

o) Secured Data Size

Secured data size (Huang et al., 2019) is defined as the entire quantity of tasks that are protected or encrypted during execution and scheduling. It directly affects the processing time, encryption overhead, network bandwidth, and storage requirements. Moreover, the secured data size is represented by the term ($\Re_{15}$).

Hence in equation (26), the entire task parameters are denoted by $\Re$.

$$\Re = \{\Re_1, \Re_2, \Re_3, \ldots, \Re_{15}\} \quad (26)$$

4.2.3 Task Scheduling Utilizing DRDAFHNet

Workflow scheduling allocates the tasks to available resources and diminishes the delay. Furthermore, it offered an even distribution of workload across the computing resources. Figure 4 shows the general model for workflow scheduling. Here, the VM parameters$\upsilon = \{\upsilon_1, \upsilon_2, \upsilon_3, \ldots, \upsilon_{\aleph}\}$and task parameters$t = \{t_1, t_2, t_3, \ldots, t_{\beth}\}$are considered as a matrix P with a size $[\alpha \times m]$. The input P is given as input of Blockchain_DRDAFHNet to obtain the output matrix$K$. The scheduled output is based on the VM and task parameters.

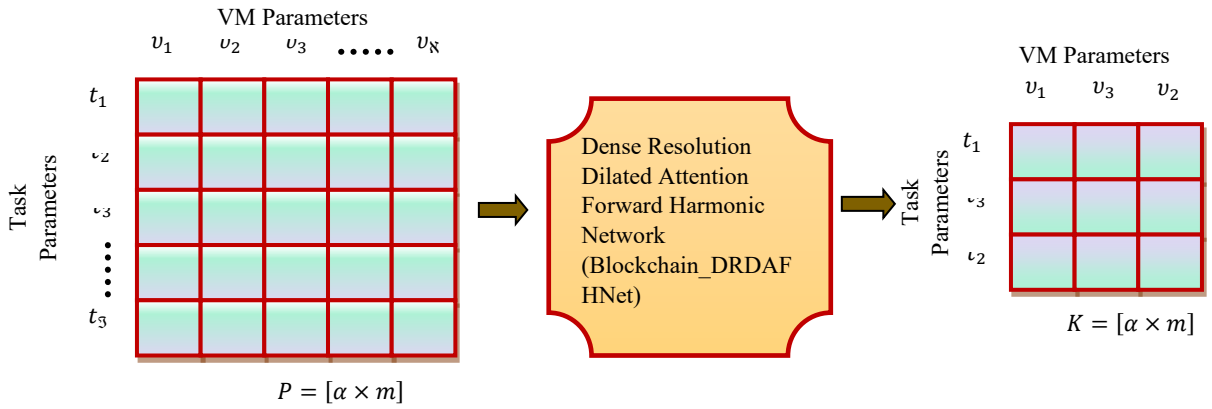

Figure 4: Schematic view for workflow scheduling

a) Structure of DRDAFHNet

The Hybrid DRDAFHNet is designed for multi-objective workflow scheduling in a blockchain-enabled cloud.

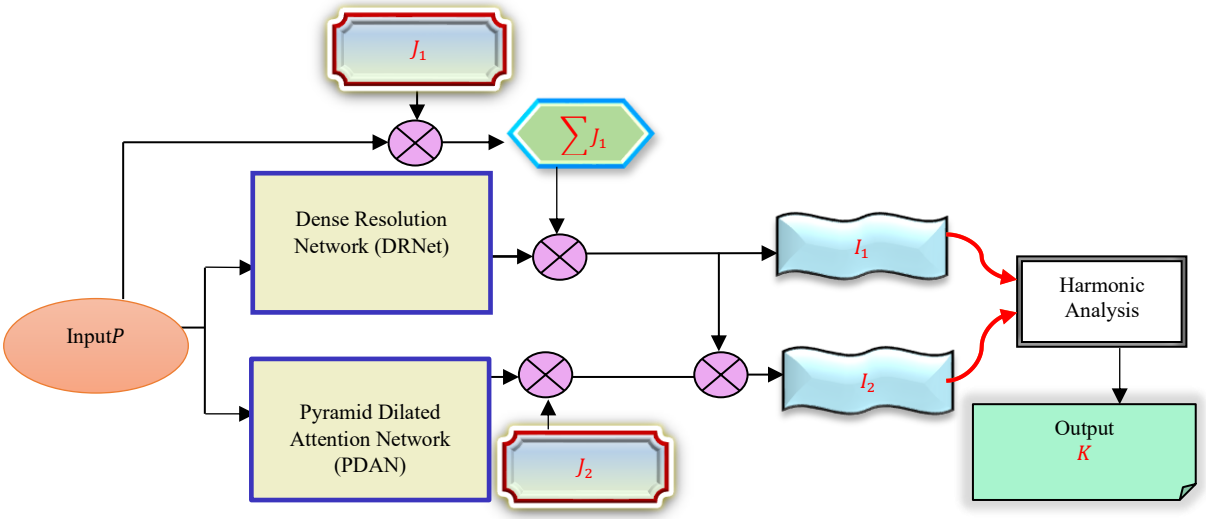

Figure 5: Structure of DRDAFHNet

It combines DRNet and PDAN to overcome limitations of single DL models and reduce overfitting. DRNet enables hierarchical feature extraction through its dense multi-resolution structure, capturing both local and global scheduling patterns. PDAN complements this by modeling long-range dependencies using dilated attention, effectively learning relationships among key scheduling factors such as makespan, execution order, and resource allocation.The structure of DRDAFHNet is portrayed in figure 5, where the matrix P with the size $[\alpha \times m]$is fed as the input of DRNet (Qiu et al., 2021) and PDAN (Dai et al., 2021). The terms J_1 and J_2 shows the weight of DRNet and PDAN. The input$P$is multiplied with J_1, and then compute $\sum J_1$. Afterwards, the output of DRNet is multiplied to $\sum J_1$ to obtain I_1. Later, the output of PDAN is multiplied withJ_2, and then its output is multiplied with I_1 to obtain I_2. At last, I_1 and $I_2$are fed to the forward harmonic analysis to obtain the outcome K. Here, the Adam Optimizer tunes the hyperparameters of DRDAFHNet.

i) DRNet

In figure 6 depicts the schematic view of DRNet (Qiu et al., 2021), where Multi-Resolution (MR), and the Full-Resolution (FR) are employed as the key components. Furthermore, the neighbor indices are created by the Adaptive Dilated Point Grouping (ADPG) algorithm. Here, the matrix P with the size $[\alpha \times m]$is fed as the input of DRNet.

-FR Branch

The Fully Convolution (FC) layer serves as an FR module, where the feature learning is based on a consistent number of points in the embedding space. The per-point feature is maintained during upsampling. Hence, it is critical to learn the precise representations of point-wise features. In addition, FR is based on the cascaded error-minimized elements. Here, the ADPG creates an adaptable neighborhood, and error-minimizing blocks (E-M) learns full-resolution points. To attain detailed knowledge of abstract embedding, the learning of features at different scales is combined.

-MR Branch

The MR block represents the lightweight down/up-sampling structure that is used to scrutinize point clouds. The skip links with the propagated features are densely linked to improve the connection among the different point cloud resolutions, and feature embedding scales. Furthermore, the MR branch offers a comprehensive channel-wise element of point clouds.

-Feature Mapping

An acceptable merging strategy is utilized for combining the feature maps from the FR and MR branches. Here, CNNs integrate the feature maps through multiplication, summation, and concatenation. The Max-pooling and Multilayer Perceptrons (MLPs) summarize the information from MR module.

Once the sigmoid function is applied, the outcome of DRNet is expressed in equation (27),

$$I_I = \left[B \times \ell\left[C\left(\underset{\Phi}{MAX}(P)\right)\right]\right] * \sum J_I \times P \tag{27}$$

Here, the Error Minimizing (E-M) unit is denoted by C, and the weight is specified by J_I. Moreover, the outcome of DRNet is denoted by the term I_I.

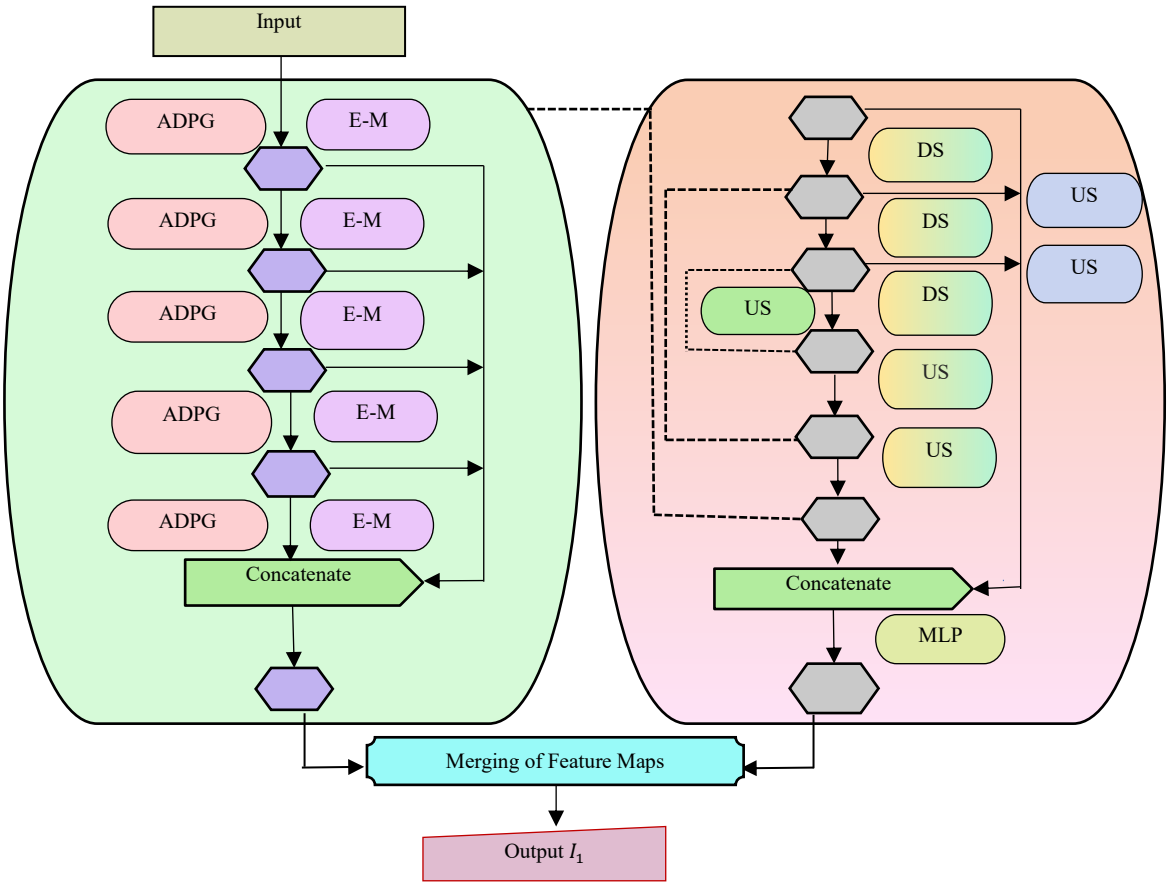

Figure 6: Schematic depiction of DRNet

ii) PDAN

In the PDAN (Dai et al., 2021), a pyramid of Dilated Attention Layer (DALs) with similar kernel sizes is included. The DAL is employed as the key component in PDAN. The DAL is followed by a ReLU and a bottleneck with residual links. Based on the attention mechanism within the kernels, the DAL extracts more precise feature representations from neighboring frames. The 1-dimensional (1D) convolution is specified by the bottleneck, which processes over time as per the kernel size-1. The DAL with several dilation rates obtains attention weights on various temporal scales. The feature maps are processed by the DAL and preserve the spatial details. The temporal layer enables the network and simulates the long, and short action patterns using local frames. Then the feature is applied to the bottleneck layer, which lightens the model by reducing the size of the channel. Hence, it maintains a constant temporal length. The structure of PDAN is displayed in figure 7, where the matrix P with the size $[\alpha \times m]$is fed as the input, and the outcome is denoted by I_2. Moreover, the outcome of PDAN(I_2) is formulated in equation (28) and (29),

$$I_2 = Sigmoid\left(G_{R_{X+1}}P\right) \times J_2 \times P \tag{28}$$

$$I_2 = Sigmoid\left(G_{R_{X+1}}P\right) \times J_2 \times \left[\left[B \times \ell\left[C\left(\underset{\Phi}{MAX}(P)\right)\right]\right] * \sum J_1 \times P\right] \tag{29}$$

where, $G_{R_{X+1}}$specifies the action group.

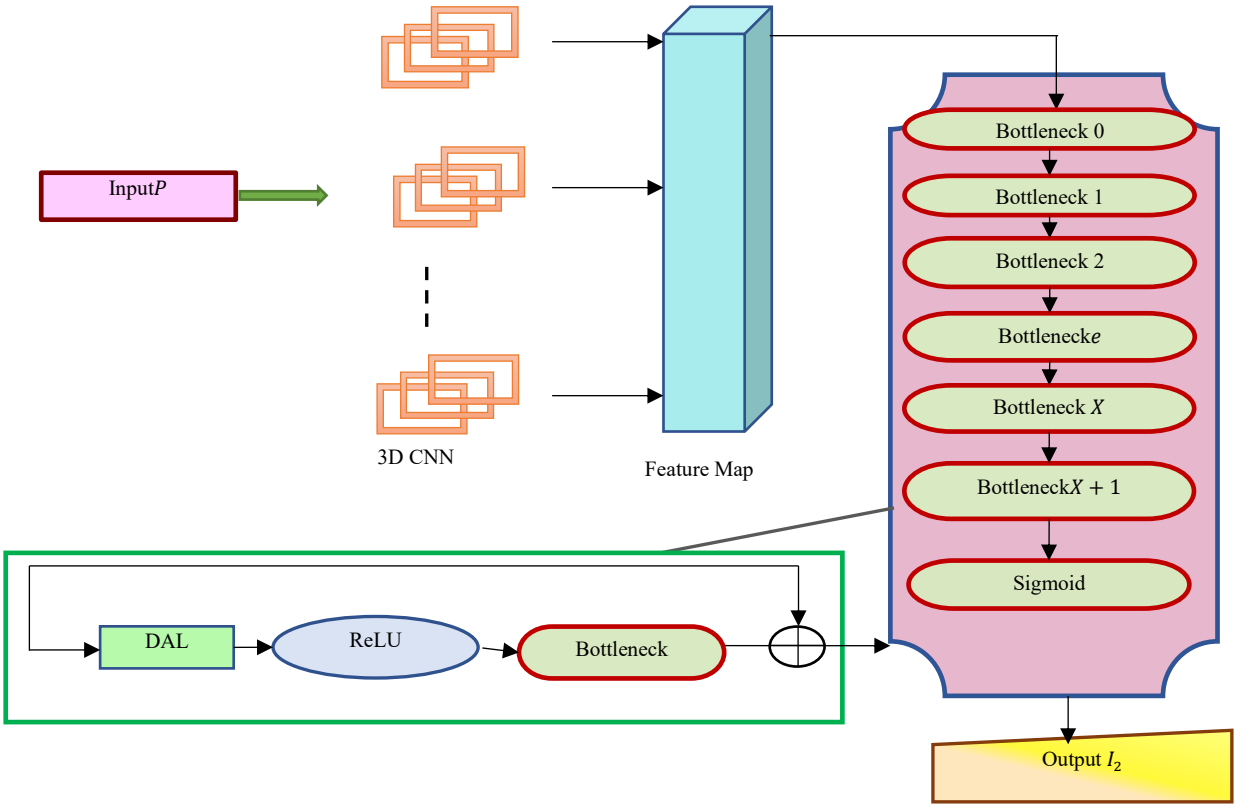

Figure 7: Structure of PDAN

iii) Forward Harmonic Analysis

Harmonic analysis is an arithmetical measure for indicating the irregular functions by the utilization of cosine, and sine values. Moreover, the harmonic analysis is used for minimizing the computational complexity. Here, the outcome of DRNet $(I_1)$and the outcome of PDAN$(I_2)$are used for harmonic estimation. Based on the time series $\Psi_{ts}(ts = 1,2,\ldots,\vartheta)$, the expression for harmonics is expressed as equation (30) to (44),

$$\sigma(\varsigma) = \lambda_0 + \sum_{\varpi=1}^{f}(\lambda_\varpi \cos(2\pi\varsigma\varpi/g) + h_\varpi \sin(2\pi\varsigma\varpi/g)) \quad (30)$$

Where,

$$\lambda_0 = \frac{1}{g}\sum_{\varsigma=1}^{g}\sigma(\varsigma) \quad (31)$$

$$\lambda_\varpi = \frac{2}{g}\sum_{t\varsigma=1}^{g}\sigma(\varsigma)\cos(2\pi\varsigma\varpi/g) \quad (32)$$

$$h_\varpi = \frac{2}{g}\sum_{\varsigma=1}^{g}\sigma(\varsigma)\sin(2\pi\varsigma\varpi/g) \quad (33)$$

Assume$g = 2, and f = 1$. Then Eq. (30), Eq. (31), Eq. (32), and Eq. (33) becomes,

$$\sigma(\varsigma) = \lambda_0 + \lambda_\varpi \cos(\varsigma\pi) + h_\varpi \sin(\varsigma\pi) \quad (34)$$

$$\lambda_0 = \frac{1}{2}\sum_{\varsigma=1}^{2}\sigma(\varsigma) \quad (35)$$

$$\lambda_0 = \frac{1}{2}[\sigma_1 + \sigma_2] \quad (36)$$

$$\lambda_1 = \frac{2}{2}\sum_{\varpi=1}^{2}\sigma(\varsigma)\cos\left(\frac{2\varsigma\pi}{g}\right) \quad (37)$$

$$\lambda_1 = \sigma_1\cos\left(\frac{2\pi}{2}\right) + \sigma_2\cos\left(\frac{2\pi\times 2}{2}\right) \quad (38)$$

$$\lambda_1 = \sigma_1(-1) + \sigma_2(1) \quad (39)$$

$$h_1 = \frac{2}{2}\sum_{\varpi=1}^{2}\sigma(\varpi)\sin\left(\frac{2\pi\varsigma\pi}{g}\right) \quad (40)$$

$$h_1 = \sigma_1\sin\left(\frac{2\pi}{2}\right) + \sigma_2\sin\left(\frac{2\pi\times 2}{2}\right) \quad (41)$$

$$\delta_1 = \Psi_1(0) + \Psi_2(0) \quad (42)$$

Substituting Eq. (35), Eq. (40), and Eq. (42) in Eq. (34).

$$\sigma(\varsigma) = \frac{1}{2}[\sigma_1 + \sigma_2] + [\sigma_1(-1) + \sigma_2(1)]\cos(\pi\varsigma) + 0 \times \sin(\pi\varsigma) \quad (43)$$

$$\sigma(\varsigma) = \frac{1}{2}[\sigma_1 + \sigma_2] + [\omega_1(-1) + \sigma_2(1)]\cos(\pi\varsigma) \quad (44)$$

Forward harmonic analysis serves as a critical fusion mechanism that combines the multi-resolution features from DRNet and the attention-based dependencies from PDAN by modeling their outputs as harmonic time series. This approach helps capture temporal patterns and smooths irregularities, leveraging past learned data to improve scheduling prediction accuracy. By leveraging past learning data within this harmonic framework, the model can reduce overfitting and enhance generalization.

The time series is given in equation (45) to (51),

$$\sigma(1) = \sigma(\varsigma - 1), \quad \sigma(2) = \sigma(\varsigma)\,, and \sigma(\varsigma) = \sigma(\varsigma + 1) \quad (45)$$

Applying Eq. (45) in Eq. (44),

$$\sigma(\varsigma + 1) = \frac{1}{2}[\sigma(\varsigma - 1) + \sigma(\varsigma)] + [\sigma(\varsigma) - \sigma(\varsigma - 1)]\cos(\pi\varsigma) \quad (46)$$

$$\sigma(\varsigma) = I_1 \quad (47)$$

$$\sigma(\varsigma - 1) = I_2 \quad (48)$$

$$\sigma(\varsigma + 1) = K \quad (49)$$

$$K = I_1\left[\frac{1}{2} + \cos(\pi\varsigma)\right] + I_2\left[\frac{1}{2} - \cos(\pi\xi)\right] \quad (50)$$

Substitute the values of I_1, and I_2in Eq. (50),

$$K = \left(\left[B \times \ell\left[C\left(\underset{\Phi}{MAX}(P)\right)\right]\right] * \sum J_1 \times P\right)\left[\frac{1}{2} + \cos(\pi ts)\right] +$$
$$\left(Sigmoid\left(G_{R_{X+1}}P\right) \times J_2 \times \left[\left[B \times \ell\left[C\left(\underset{\Phi}{MAX}(P)\right)\right]\right] * \sum J_1 \times P\right]\right)\left[\frac{1}{2} - \cos(\pi ts)\right] \quad (51)$$

The outcome of DRDAFHNet t is denoted by K. The resulting harmonized output Kreflects a balanced fusion of hierarchical multi-scale features and long-range dependencies, leading to more accurate and stable workflow scheduling decisions.

Algorithm 1: Blockchain_DRDAFHNet-based Multi-Objective Workflow Scheduling

Input:

- Task set T = {t1, t2, ..., tn}

- VM set VM = {vm1, vm2, ..., vmk}
- VM parameters: CPU (C), Bandwidth Utilization (BU)
- Task parameters: TP, EST, EFT, TL, ATRT, M, MC, R, RU, Tr, SL, SO, ES, SDS, PE (via EA-LSTM)

Output: Optimal Task-to-VM assignment matrix

Step 1: Blockchain Initialization
- Initialize miners, smart contracts, and TPC.
- Assign anonymized IDs to tasks; distribute as smart contracts.

Step 2: VM Parameter Extraction
- Compute CPU (C) and Bandwidth Utilization (BU) for each VMi. [Eqs. 1–2]
- Construct VM parameter matrix P_VM of size $\alpha \times 5$.

Step 3: Task Parameter Computation

For each task tj, compute:
- TP, EST, EFT, TL, ATRT [Eqs. 3–7]
- Predicted Energy PE via EA-LSTM [Eqs. 8–14]
- Trust (Tr), Memory (M), Makespan Cost (MC), Reliability (R) [Eqs. 15–21]
- RU, SL, SO, ES, SDS [Eqs. 22–25]
- Construct task parameter matrix P_Task of size $\alpha \times m$. [Eq. 26]

Step 4: Input Matrix Construction
- Combine: P = [P_VM || P_Task], size $\alpha \times (m + 5)$

Step 5: DRDAFHNet Scheduling

5a. DRNet: Extract multi-resolution hierarchical features → O_DRNet [Eq. 27]

5b. PDAN: Extract long-range attention features → O_PDAN [Eqs. 28–29]

5c. Forward Harmonic Analysis: Fuse O_DRNet and O_PDAN via harmonic decomposition → O_DRDAFHNet [Eqs. 30–51]

Step 6: Hyperparameter Optimization
- Optimize DRDAFHNet weights using Adam Optimizer.
- Apply dropout (0.5) and L2 regularization ($\lambda = 0.0001$).

Step 7: Schedule Selection & Execution
- Select task-to-VM assignment optimizing:
  → Minimized Makespan and Resource Utilization
  → Maximized Residual Energy
- Deliver schedule to cloud VMs; store results on-chain via smart contracts.

Return: Optimal task-to-VM assignment

Algorithm 1 begins by initializing the blockchain network and anonymizing tasks via smart contracts. VM and task parameters are extracted and combined into a unified input matrix, which is processed through DRNet for multi-resolution feature extraction and PDAN for long-range dependency modeling. The resulting feature streams are fused using forward harmonic analysis to generate the optimal scheduling output. Finally, the task-to-VM assignment that minimizes makespan and resource utilization while maximizing residual energy is delivered to the cloud, with results stored on-chain for auditability.

5 Result and Discussion

The experimentation of Blockchain_DRDAFHNet-based multi-objective workflow scheduling in blockchain-aided cloud. Furthermore, the implementation tool, metrics used, and comparative assessment are explained.

5.1. Experimental Set-up

Table 2: Simulation settings and DL parameters

Parameters	Values
Programming Platform	Python
Simulation Tool	CloudSim
Number of Data Centers	3
Number of Virtual Machines	20, 40, 50
VM MIPS	1000
Bandwidth	500 Mbps
Number of Tasks	100
Task Length	10000 MI
Block Size	1 MB
Block Generation Time	12 sec
Hash Algorithm	SHA-256
Encryption Technique	AES-256
DRDAFHNet Parameters	
DRNet Dilation Rates	1, 2, 4
PDAN Dilation Rates	1, 2, 3, 5
Pyramid Levels	3
Activation Function	ReLU
Optimizer	ReLU
Learning rate	0.0007
Batch Size	32
Epochs	100
Loss Function	MSE
Dropout Rate	0.5
Weight Initialization	He Normal
Regularization	L2 (0.0001)
Fusion Layer Filters	256
Output Layer	Softmax
Hardware Requirements	
OS	Windows 10
RAM	8GB
ROM	More than 100 GB
CPU	2.4 GHZ
GPU	2 GB

The Blockchain_DRDAFHNet-based multi-objective workflow scheduling experiments are conducted using CloudSim, with the simulation configured to model heterogeneous VMs and tasks. The simulations are implemented in Python 3.9.11, and the deep learning components of the proposed Blockchain_DRDAFHNet model are developed using TensorFlow and Keras frameworks. Table 2 shows the simulation parameters used by the Blockchain_DRDAFHNet model.

5.2 Evaluation measures

The metrics like residual energy, makespan, and resource utilization are used to estimate the efficiency of Blockchain_DRDAFHNet.

5.2.1 Residual Energy

Residual energy is the amount of energy remaining in the computing resources after executing a set of scheduled tasks.

5.2.2 Resource utilization

The distribution of resources to the VM for achieving effective scheduling is termed by resource allocation.

5.2.3 Makespan

Makespan is the total time elapsed from the start to the completion of a set of tasks, and it is illustrated in section 4.1.2.

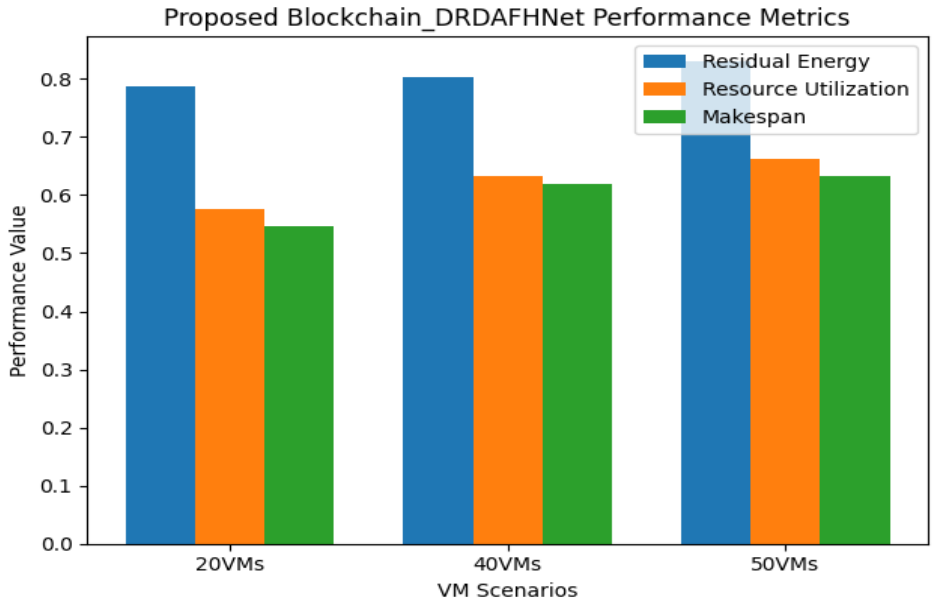

Figure 8: Performance of blockchain_DRDAFHNet across VM scenarios

In figure 8 shows the performance of the proposed Blockchain_DRDAFHNet across 20, 40, and 50 VM scenarios in terms of residual energy, resource utilization, and makespan. Results indicate improved energy efficiency and resource utilization with increasing VM scale, while makespan remains stable, demonstrating the model's scalability and effective scheduling performance.

5.3 Comparative Techniques

The effectiveness of the Blockchain_DRDAFHNet-based multi-objective workflow scheduling in blockchain-aided cloud is evaluated by considering the existing methods such as ALPSO (Thekkepuryil et al., 2021), Actor-Critic architecture (Dong et al., 2021), CGA (Mohammadzadeh et al., 2021), Two-Stage Deep Reinforcement Learning with Graph AttentionNetwork (2SD-GAT) (Chandrasiri & Meedeniya, 2026), HGALO-GOA (Mohammadzadeh et al., 2021), Deep Neural Network model with Improved Grey Wolf–Horse Herd Optimization (DNN-IGW-HHO) (Kumar et al., 2026), Reinforcement Learning-Based Multi-Objective Task Scheduling (RL-MOTS) (Sardaraz & Tahir, 2019).

In table 3 shows the comparative discussion of Blockchain_DRDAFHNet, where the performance estimation is done by considering different counts of VMs and PMs. Here, the superior values of residual energy, resource utilization, and makespan are attained by the Blockchain_DRDAFHNet using 10 PM with 20 VMs. Here, the Blockchain_DRDAFHNet attains the residual energy of 0.831 J, whereas the residual energy attained by the existing methods is 0.650J, 0.689J, 0.713J, 0.739J, 0.759J, 0.772J, and 0.796J. The resource allocation of 0.576 is attained by the Blockchain_DRDAFHNet, and the values 0.769, 0.737, 0.713, 0.709, 0.706, 0.693, and 0.603 are obtained by the existing methods. The makespan of 0.546 is achieved by the Blockchain_DRDAFHNet, in which the values are 0.761, 0.726, 0.703, 0.682, 0.663, 0.594, and 0.559. Moreover, the superior values attained by the Blockchain_DRDAFHNet for residual energy, resource utilization, and makespan are 0.831J, 0.576, and 0.546. The optimal outcomes are obtained by the proposed Blockchain_DRDAFHNet, because the developed model is more robust, and offers a secure solution in dynamic, and large-scale scheduling.

Table 3: Comparation for blockchain_DRDAFHNet-based workflow scheduling

Variations	Metrics	ALPSO	Actor-Critic architecture	CGA	2SD-GAT	HGALO-GOA	DNN-IGW-HHO	RL-MOTS	Proposed Blockchain_DRDAFHNet
10 PM with 20 VMs	Residual Energy (J)	0.656	0.673	0.700	0.713	0.715	0.716	0.736	0.788
	Resource utilization	0.769	0.737	0.713	0.709	0.706	0.693	0.603	**0.576**
	Makespan	0.761	0.726	0.703	0.682	0.663	0.594	0.559	**0.546**
10 PM with 40 VMs	Residual Energy (J)	0.650	0.675	0.722	0.728	0.731	0.740	0.765	0.803
	Resource utilization	0.787	0.765	0.735	0.708	0.704	0.676	0.653	0.632
	Makespan	0.769	0.759	0.734	0.704	0.700	0.682	0.645	0.620
15 PM with 50VMs	Residual Energy (J)	0.650	0.689	0.713	0.739	0.759	0.772	0.796	**0.831**
	Resource utilization	0.808	0.793	0.763		0.739		0.685	0.662
	Makespan	0.783	0.768	0.747	0.740	0.721	0.698	0.664	0.633

5.5.1 Ablation Study

Table 4: Ablation study results

Model Variant	Residual Energy (J)	Resource Utilization	Makespan
DRNet only	0.784	0.721	0.698
PDAN only	0.768	0.709	0.684
DRNet + PDAN (concat)	0.811	0.685	0.651
Full DRDAFHNet (proposed)	**0.831**	**0.662**	**0.633**

In table 4 shows that the proposed DRDAFHNet achieves the best performance across all metrics, with the highest residual energy and the lowest resource utilization and makespan, confirming its effectiveness.

5.6 Limitations

The proposed model has a few notable limitations. Evaluation is conducted solely in CloudSim, and real-world deployment on platforms such as AWS or Azure may reveal additional complexities. The workload is static and uniform, unlike dynamic real-world workflows. Scalability beyond 50 VMs and 15 PMs remains untested, and blockchain overhead has not been independently quantified. Generalization to other workflow types and hybrid cloud architectures, and the impact of hardware-level energy factors not captured by EA-LSTM, are left for future investigation.

6 Conclusion

Workflow scheduling in blockchain-assisted cloud systems must simultaneously optimize energy efficiency, resource utilization, and task makespan while ensuring security and reliability in dynamic, heterogeneous environments. To address these challenges, this paper proposed the Blockchain_DRDAFHNet a hybrid deep learning model integrating a Dense Resolution Network (DRNet) and a Pyramid Dilated Attention Network (PDAN), unified through forward harmonic analysis. Workflow scheduling was guided by a comprehensive set of VM and task parameters, including CPU utilization, bandwidth, trust, security level, security overhead, encryption speed, and predicted energy estimated via an Evolutionary Attention-based LSTM (EA-LSTM). Experiments conducted in CloudSim across three VM/PM configurations confirm the superiority of the proposed model. Under the 15 PM/50 VM setting the most resource-intensive configuration Blockchain_DRDAFHNet achieved a residual energy of 0.831 J, resource utilization of 0.576, and makespan of 0.546, outperforming all seven comparative methods including ALPSO, Actor-Critic, CGA, 2SD-GAT, HGALO-GOA, DNN -IGW-HHO, and RL-MOTS. These results confirm that the integration of blockchain-based security with multi-resolution and attention-based deep learning yields a more robust and generalizable scheduling solution. Future work will explore federated learning strategies to train scheduling models collaboratively across geographically distributed cloud nodes without centralizing sensitive data, thereby enhancing both model generalization and privacy preservation. Additionally, extending the framework to edge-cloud and fog-cloud hybrid environments, incorporating real-world workload traces, and evaluating scalability under dynamically arriving tasks are identified as important directions for subsequent research.

References

[1] Adhikari, M., Amgoth, T., & Srirama, S. N. (2019). A survey on scheduling strategies for workflows in cloud environment and emerging trends. *ACM Computing Surveys (CSUR)*, *52*(4), 1-36. https://doi.org/10.1145/3325097

[2] Ali, N. A. M., & Ali, F. A. M. (2023). Optimizing cloud-fog-edge job scheduling using catastrophic genetic algorithm and block chain-based trust: A collaborative approach. *Journal of Applied Engineering and Technological Science, 5*(1), 569–580. https://doi.org/10.37385/jaets.v5i1.3125

[3] Badri, S., Alghazzawi, D. M., Hasan, S. H., Alfayez, F., Hasan, S. H., Rahman, M., & Bhatia, S. (2023). An efficient and secure model using adaptive optimal deep learning for task scheduling in cloud computing. *Electronics*, *12*(6), 1441. https://doi.org/10.3390/electronics12061441

[4] Bal, P. K., Mohapatra, S. K., Das, T. K., Srinivasan, K., & Hu, Y. C. (2022). A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques. *Sensors*, *22*(3), 1242. https://doi.org/10.3390/s22031242

[5] Chandrasiri, S., & Meedeniya, D. (2025). Energy-efficient dynamic workflow scheduling in cloud environments using deep learning. *Sensors*, *25*(5), 1428. https://doi.org/10.3390/s25051428

[6] Chandrasiri, S., & Meedeniya, D. (2026). Multi-Objective Cloud Workflow Scheduling with Two-Stage Deep Reinforcement Learning and Graph Neural Networks. *IEEE Access*.14, 34802-34829. https://doi.org/10.1109/access.2026.3669772

[7] Chaudhary, D., & Kumar, B. (2018). Cloudy GSA for load scheduling in cloud computing. *Applied Soft Computing*, *71*, 861-871. https://doi.org/10.1016/j.asoc.2018.07.046

[8] Choudhary, A., Gupta, I., Singh, V., & Jana, P. K. (2018). A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing. *Future Generation Computer Systems*, *83*, 14-26. https://doi.org/10.1016/j.future.2018.01.005

[9] Dai, R., Das, S., Minciullo, L., Garattoni, L., Francesca, G., & Bremond, F. (2021). Pdan: Pyramid dilated attention network for action detection. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 2970-2979). https://doi.org/10.1109/WACV48630.2021.00301

[10] Dasgupta, D., Shrein, J. M., & Gupta, K. D. (2019). A survey of blockchain from security perspective. *Journal of Banking and Financial Technology*, *3*(1), 1-17. https://doi.org/10.1007/s42786-018-00002-6

[11] Dong, T., Xue, F., Xiao, C., & Zhang, J. (2021). Workflow scheduling based on deep reinforcement learning in the cloud environment. *Journal of Ambient Intelligence and Humanized Computing*, *12*(12), 10823-10835. https://doi.org/10.1007/s12652-020-02884-1

[12] Ge, J., He, Q., & Fang, Y. (2017, April). Cloud computing task scheduling strategy based on improved differential evolution algorithm. In *AIP conference proceedings* (Vol. 1834, No. 1, p. 040038). AIP Publishing LLC. https://doi.org/10.1063/1.4981634

[13] Guo, H., & Yu, X. (2022). A survey on blockchain technology and its security. *Blockchain: research and applications*, *3*(2), 100067. https://doi.org/10.1016/j.bcra.2022.100067

[14] Huang, B., Li, Z., Tang, P., Wang, S., Zhao, J., Hu, H., ... & Chang, V. (2019). Security modeling and efficient computation offloading for service workflow in mobile edge computing. *Future Generation Computer Systems*, *97*, 755-774. https://doi.org/10.1016/j.future.2019.03.011

[15] Kakkottakath Valappil Thekkepuryil, J., Suseelan, D. P., & Keerikkattil, P. M. (2021). An effective meta-heuristic based multi-objective hybrid optimization method for workflow scheduling in cloud computing environment. *Cluster Computing*, *24*(3), 2367-2384. https://doi.org/10.1007/s10586-021-03269-5

[16] Kumar, M., Kant, R., Gupta, B. K., Shadab, A., Kumar, A., & Kant, K. (2026). IDN-MOTSCC: Integration of Deep Neural Network with Hybrid Meta-Heuristic Model for Multi-Objective Task Scheduling in Cloud Computing. *Computers*, *15*(1), 57. https://doi.org/10.3390/computers15010057

[17] Li, Y., Zhu, Z., Kong, D., Han, H., & Zhao, Y. (2019). EA-LSTM: Evolutionary attention-based LSTM for time series prediction. *Knowledge-Based Systems*, *181*, 104785. https://doi.org/10.1016/j.knosys.2019.05.028

[18] Mazrekaj, A., Sheholli, A., Minarolli, D., & Freisleben, B. (2019, May). The Experiential Heterogeneous Earliest Finish Time Algorithm for Task Scheduling in Clouds. In *CLOSER* (pp. 371-379). https://doi.org/10.5220/0007722203710379

[19] Meena, J., Kumar, M., & Vardhan, M. (2016). Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint. *IEEE Access*, *4*, 5065-5082. https://doi.org/10.1109/access.2016.2593903

[20] Mikram, H., El Kafhali, S., & Saadi, Y. (2024). HEPGA: A new effective hybrid algorithm for scientific workflow scheduling in cloud computing environment. *Simulation modelling practice and theory*, *130*, 102864. https://doi.org/10.1016/j.simpat.2023.102864

[21] Mohammadzadeh, A., Masdari, M., & Gharehchopogh, F. S. (2021). Energy and cost-aware workflow scheduling in cloud computing data centers using a multi-objective optimization algorithm. *Journal of Network and Systems Management*, *29*(3), 31. https://doi.org/10.1007/s10922-021-09599-4

[22] Murthy, C. V. B., Shri, M. L., Kadry, S., & Lim, S. (2020). Blockchain based cloud computing: Architecture and research challenges. *IEEE access*, *8*, 205190-205205. https://doi.org/10.1109/access.2020.3036812

[23] Naik, B. B., Singh, D., & Samaddar, A. B. (2021). Multi-objective virtual machine selection in cloud data centers using optimized scheduling. *Wireless Personal Communications*, *116*(3), 2501-2524. https://doi.org/10.1007/s11277-020-07807-z

[24] Qiu, S., Anwar, S., & Barnes, N. (2021). Dense-resolution network for point cloud classification and segmentation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 3813-3822). https://doi.org/10.1109/wacv48630.2021.00386

[25] Saeedi, S., Khorsand, R., Bidgoli, S. G., & Ramezanpour, M. (2020). Improved many-objective particle swarm optimization algorithm for scientific workflow scheduling in cloud computing. *Computers & Industrial Engineering*, *147*, 106649. https://doi.org/10.1016/j.cie.2020.106649

[26] Sardaraz, M., & Tahir, M. (2019). A hybrid algorithm for scheduling scientific workflows in cloud computing. *IEEE Access*, *7*, 186137-186146. https://doi.org/10.1109/access.2019.2961106

[27] Shakkeera, L. (2025). Efficient task scheduling and computational offloading optimization with federated learning and blockchain in mobile cloud computing. *Results in Control and Optimization*, *18*, 100524. https://doi.org/10.1016/j.rico.2025.100524

[28] Swarup, S., Shakshuki, E. M., & Yasar, A. (2021). Task scheduling in cloud using deep reinforcement learning. *Procedia Computer Science*, *184*, 42-51. https://doi.org/10.1016/j.procs.2021.03.016

[29] Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on blockchain technology?—a systematic review. *PloS one*, *11*(10), e0163477. https://doi.org/10.1371/journal.pone.0163477

[30] Zhang, Y., & Wang, J. (2024). Enhanced whale optimization algorithm for task scheduling in cloud computing environments. *Journal of engineering and applied science*, *71*(1), 121. https://doi.org/10.1186/s44147-024-00445-3

Authors Biography

G. Madhumala is an Assistant Professor in the Department of Computer Science and Engineering at Sri Siddhartha Institute of Technology , Tumakuru, Karnataka ,India. She obtained her Bachelor's degree in Computer Science and Engineering and Master's degree in Computer Science and Engineering from Visvesvaraya Technological University(VTU) ,Belagavi. She has an extensive teaching experience and is actively involved in research and academic activities. Her research interests include Cloud Computing , Blockchain Technology, Machine Learning, Deep Learning. She has published research papers in many of the reputed national and international journals and conferences. She has guided several undergraduare projects in emerging areas of computer science.

Dr.R. Suma is Professor in the Department of Information Science and Engineering at Sri Siddhartha Institute of Technology (SSIT), Tumakuru, Karnataka, India. She holds an M.Tech. degree. and a Ph.D. from Visvesvaraya Technological University (VTU), awarded in 2018. With over 19 years of teaching and research experience, her areas of specialization include Data Analytics, Network Security, and Image Processing. She has published several research papers in reputed national and international journals and conferences, including Scopus- and Web of Science-indexed publications, and has contributed book chapters in the areas of MANETs and VANETs. She is a life member of ISTE and actively participates in academic, research, and administrative activities. Her research interests focus on secure routing protocols, machine learning applications in network security, data analytics, and intelligent communication systems.