AI-Enhanced Stage-Aware Deadline Division for Secure Multi-Cloud Resource Management and Performance Prediction

Wenchong Fang[*], Zhifeng Zhou, Xiqi He, Yingchen Li, Danli Xu
China Southern Power Dispatching & Control Center, Guangzhou 510663, Guangdong, China
E-mail: fangwc851@outlook.com
[*]Corresponding author

Keywords: artificial intelligence, multi-cloud resource security management, performance prediction, SBSAD algorithm, multi-task learning

Received: July 16, 2025

Using the identical traces and capacity controls described in Section 4, SBSAD decreased the end-to-end turnaround time and increases the satisfaction rate of meeting deadlines, while reducing the number of late or dropped tasks and preemptions. These gains stemmed from stage-based deadline bucketing and a single scalar priority that considered urgency, security, dependency, and predicted runtime. This was in contrast to DRL-RSM, where reward shaping and global queues make deadline pressure less explicit. Unlike STGC-SM that relied on static graph features or offline prediction, STGC-SM directly consumed online multitasking prediction (runtime/energy/fault risk) within the scheduler. Therefore, its response to short-term load changes was slower. Moreover, by preempting cost feedback for cross stage promotion, it avoided the convergence overhead observed in the sudden arrival of MARL-RMM. In practical cloud stacks, integration was straightforward: In Kubernetes, a scheduler framework plugin or extender could replace the native scoring with a scalar priority while applying GCN-derived feasibility masks in the filter phase. The predictor was deployed as a telemetry-driven sidecar. Equivalent behavior in OpenStack-like platforms followed from custom host filters and weights. Operational interpretation revealed that improved accuracy reduced queue build-up and deadline violations for batch and ML workflows in shared clusters. It also lowered interference in isolation-critical, multi-tenant SaaS and stabilized edge-to-cloud video and ETL pipelines during short-term surges. Limitations remain: Reliance on predictor quality and stable telemetry can bias prioritization. The CloudSim environment simplifies network and storage contention compared to production systems. Preemption cost and fairness are modeled beyond deadline satisfaction at a coarse level. These factors define low-risk deployment zones and indicate where additional engineering hardening or A/B trials are recommended.

Povzetek: Študija pokaže, da SBSAD algoritem z enotno prioriteto in sprotnimi napovedmi bistveno skrajša čas izvedbe ter zmanjša zamude in prekinitve nalog.

1 Introduction

Due to increased resource elasticity, cost optimization, AI-enabled high-availability services, and other benefits, artificial intelligence (AI)-driven multi-cloud (MC) architecture has emerged as the key infrastructure for enterprise digital transformation as cloud computing technology advances quickly [1]. However, in a multi-cloud environment (MCE), AI needs to deal with the dynamic heterogeneous nature of compute, storage, and network resources. Furthermore, AI also needs to deal with dynamic security threats such as network attacks and malicious traffic. This makes AI-enabled resource scheduling need to simultaneously meet the dual goals of efficiency and network security protection [2-3]. It is important to clarify that "security" in the context of this research encompasses two distinct but related dimensions. The first is cybersecurity, which involves detecting and protecting against external threats, such as malicious traffic and network intrusions. This is addressed through an anomaly detection framework. The second type is policy-driven security. This pertains to enforcing workload isolation, affinity, and anti-affinity rules during the resource allocation process. These rules ensure the integrity and robustness of multi-tenant operations. The proposed framework integrates mechanisms to address both of these dimensions.

Traditional MC resource management approach that relies on manual experience and rules. It is difficult to perceive the multi-dimensional dynamic changes of the cloud environment in real time through AI. It shows significant lag in AI-driven scenarios such as resource demand prediction, cybersecurity threat identification, and cross-cloud intelligent collaborative scheduling [4-5]. First, the non-AI resource allocation model based on static thresholds cannot adapt to the volatility of business loads, leading to an imbalance in resource utilization that is difficult to optimize by AI. Second, the network security protection mechanism and the AI resource scheduling process are severed from each other. It lacks AI-driven linkage response capabilities for security threats and resource anomalies. Third, the single-modal data-driven non-AI prediction model is difficult to capture the complex spatio-temporal correlation features in MCEs through machine learning. This leads to insufficient AI performance prediction accuracy [6-7].

Basha H A et al. proposed an adaptive multi-objective MC resource management algorithm for resource management in MCEs to address the resource optimization challenges in dynamic environments, fluctuating workloads and service objectives. According to experiments, it may adjust to changes in the environment, manage resources in real time, and enhance MC systems' overall performance and resource use [8]. Alonso J et al. focused on the challenge of collaborative MC application development and resource scheduling. The study pointed out through a systematic literature review those traditional architectures were lacking an intelligent sensing mechanism for dynamic resource requirements. AI techniques needed to be introduced to optimize cross-cloud resource allocation strategies [9]. Putri A addressed the challenges faced by organizations in managing MCEs by analyzing the existing literature, exploring key issues such as security, performance, cost, and governance, and proposing strategies and best practices to address them. This was carried out to guarantee smooth operations and peak performance in MCEs. The outcomes gave enterprises useful information for managing MCEs [10]. Anh N H explored the heterogeneous resource management challenges in hybrid cloud migration. The study emphasized the key role of AI in cross-platform load balancing and performance prediction, providing theoretical references for intelligent scheduling in MCEs [11]. Azizi S et al. proposed an intelligent allocation strategy based on heuristic algorithms for the cloud resource optimization problem of IoT device service deployment. The study optimized the service placement by machine learning model. Experiments revealed that its objective function value was reduced by 26.8% compared to the baseline method, validating the effectiveness of AI algorithms in resource scheduling [12].

Khan A A et al. focused on data security and transmission issues of industrial internet of things (IIoT) in eHealth environment. The study proposed a BHIIoT solution based on blockchain distributed ledger technology to address the risks involved in exchanging patient information in a centralized server system. The study reduced the storage burden and enhanced data security by optimizing the lifecycle of healthcare wireless sensor networks, using the NuCypher threshold re-encryption mechanism, and deploying a chain code automation process [13]. De Alwis C et al. constructed a NS security and privacy classification system through literature review for the security and privacy challenges brought by 5G network slicing. The study elaborated the attack scenarios, threat challenges, and solutions. Additionally, it examined the future course of trust privacy research, which served as a guide for achieving secure NS in 5G and other mobile communication networks in the future [14]. Rehan H et al. pointed out that the integration of AI and machine learning was the core direction of MC resource management in the future through industry trend analysis. Its application in scenarios such as dynamic load prediction and anomalous traffic identification could significantly enhance system intelligence [15]. Kazmi S H A et al. studied the security challenges under the convergence of 5G and software-defined networking (SDN). The study introduced AI algorithms to build real-time threat prediction models, which solved the problem of response lag of traditional security protection to novel attacks [16].

To summarize, although experts have thoroughly studied MC resource management, cloud application development, and hybrid cloud migration, there are still many problems in the existing research. For example, resource optimization methods for dynamically changing and fluctuating loads in MCEs are not yet sufficient. There is less research on the application of hybrid cloud and MC governance strategies in cross-platform quality of service assurance and cross-organizational collaboration. Data security and privacy protection issues lack in-depth analysis and other deficiencies. To address the above challenges, the study suggests a MC resource management and performance prediction framework incorporating AI algorithms. The innovation of the research is to design schedule method based on stage-aware deadline division (SBSAD) based on AI algorithm. The resource parameter estimation model for multi-modal data fusion is constructed through deep reinforcement learning. Second, a security scheduling management module based on graph convolutional network (GCN) is constructed. The firewall rules, real-time traffic monitoring, and resource allocation policies are deeply coupled. Third, the attention mechanism (AM) and Transformer architecture are introduced to build a performance prediction model for multi-task learning. This research expects cross-layer optimization through AI technology. This provides a new technical path for the synergistic improvement of resource utilization efficiency and security protection capability in MCEs.

This study addresses: (RQ1) Whether stage-based deadline bucketing with a single scalar priority can improve deadline satisfaction and end-to-end turnaround under mixed CPU/I/O loads. (RQ2) How to proactively integrate security and isolation constraints into scheduling decisions rather than applying post-hoc filtering. (RQ3) How to fuse online multi-task predictions of runtime, energy, and failure risk to adapt to short-term load shifts. Accordingly, the objectives are: (O1) To specify SBSAD with bucketed deadlines and a scalar priority driven by DQN-based resource estimates and attention-based Transformer forecasts. (O2) To encode security and policy constraints as GCN-derived per-node feasibility masks applied before packing. (O3) To evaluate the design in CloudSim under identical capacity controls against representative baselines. Integration pipeline (ingestion→decision): (1) Collect and normalize host/VM telemetry, traffic, and event logs. (2) Construct fused features from historical traces and online signals. (3) Produce DQN-based VM/resource estimates. (4) Compute GCN-based feasibility masks from security/affinity metadata. (5) Generate online multi-task predictions (runtime/energy/failure risk) via the attention-based Transformer. (6) Compute bucket membership and the scalar priority and pack non-conflicting tasks under capacity and policy constraints. (7) Feed back observed durations, violations, and preemption costs to update models.

Positioning and novelty: Previous studies on cloud scheduling have either combined deep reinforcement learning or graph-based placement without explicit deadline staging or used Transformers solely for offline prediction. The present study advances this line by introducing stage-based deadline bucketing (SBSAD) and ranking tasks with a single scalar priority that fuses urgency, security, dependency, and predicted runtime. Rather than only being profiled offline, multi-task predictions (runtime, energy, and failure risk) are consumed online by the scheduler, forming a closed loop between forecasting and decision-making. A lightweight GCN is used to encode security and policy constraints, yielding per-node feasibility masks. This enables conflict-aware greedy packing under identical capacity limits. A simple cross-stage promotion mechanism accounts for preemption cost and stabilizes performance under load shifts. These design choices work together to provide a compact, reproducible scheduler that surpasses DRL-, GCN-, and Transformer-only approaches. Section 4 reports consistent gains over DRL-RSM, STGCN-SM, MARL-RMM, and ATP-SM under identical traces and capacity controls.

Table RW summarizes representative approaches, their core techniques, the metrics commonly used to evaluate them, and the limitations that SBSAD addresses. These limitations include integrating security with scheduling via feasibility masks applied before packing, fusing multimodal signals into online decisions via DQN-informed estimates and multitask forecasting, and leveraging attention-based learning for fast adaptation, as show in Table 1.

Table 1: Representative approaches, methods, metrics, and limitations addressed by SBSAD

Approach	Core method	Security–scheduling integration	Multi-modal / online fusion
DRL-RSM	Deep RL with global queue and reward shaping	Post-hoc rule filters; violations handled after selection	Limited; predictions often offline or single-metric
STGCN-SM	Spatio-temporal GCN for placement	Static graph features; security not embedded in feasibility	Partial; topology/time only
MARL-RMM	Multi-agent RL for resource management	Local policies; coordination overhead	Per-agent signals; fusion indirect
ATP-SM	Attention-fused Transformer for prediction/scheduling	Not explicitly encoded before packing	Time-series forecasts; often decoupled from scheduler
SBSAD (this study)	Stage-based deadline bucketing + single scalar priority; GCN feasibility masks; DQN-informed estimates; attention-based Transformer for multi-task prediction	Embedded before packing (feasibility masks enforce isolation/affinity)	Online runtime/energy/risk predictions feed the scheduler; capacity updated from telemetry

2 MC resource management and performance prediction modeling

To satisfy reproducibility with minimal overhead, SBSAD is summarized as a single-pass loop. Each task is first mapped to a deadline bucket by comparing its deadline with fixed cutoffs. A shared encoder with task-specific heads outputs three predictions per task: runtime, energy, and failure risk. These predictions are continuously refreshed by recent telemetry. A single priority score ranks tasks within each bucket: $P = a/(d - \text{now}) + bs + cD - d\hat{t}$, where d is the deadline, s is the security level, D is a dependency penalty, and $\hat{t}$ is the predicted runtime. Constants a, b, c, d are fixed on validation data. The scheduler greedily packs the highest-priority non-conflicting tasks under capacity and policy constraints (isolation, affinity/anti-affinity), breaking ties by higher s and smaller $\hat{t}$. When telemetry indicates resource degradation or a risk to meeting a deadline, affected tasks are promoted to higher-priority buckets, and the lowest-priority tasks are preempted. The preemption cost is recorded for the next priority update. After completion, observed durations update the encoder, and the loop repeats. This compact specification explicitly shows how multi-task predictions feed the scheduler, while also making prioritization, conflict handling, and adaptation explicit.

2.1 AI algorithm-based parameter estimation for resource virtual machines

In MCEs, dynamic estimation of virtual machine resource parameters is one of the main challenges to achieve efficient resource management and performance prediction. Considering the multi-dimensional dynamic characteristics of compute, storage, and network resources in MCEs, AI-based parameter estimation models usually take historical resource usage data, real-time monitoring metrics, and business load models as inputs. It constructs a feature space for multi-modal data fusion [17]. The parameter estimation framework of MC resource virtual machine based on deep Q-network is shown in Figure 1.

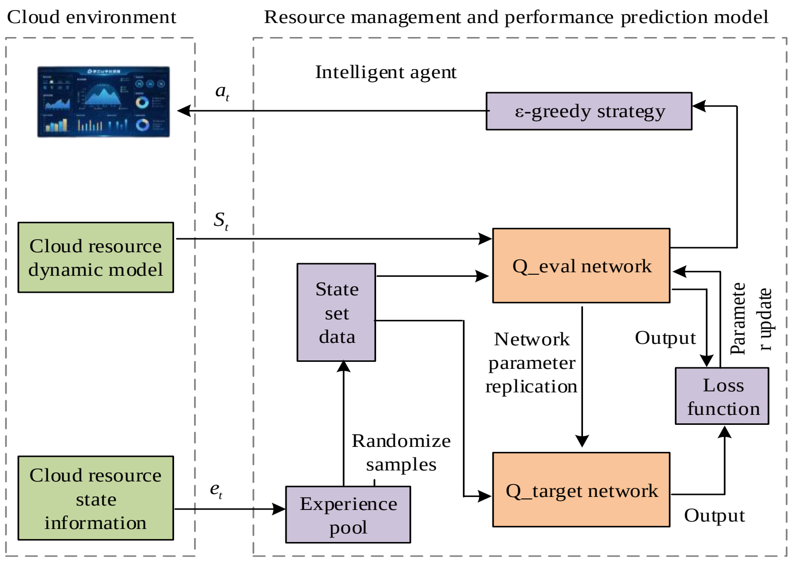


Figure 1: Deep Q-network based parameter estimation framework for MC resource virtual machine

In Figure 1, in the cloud environment section, the cloud resource dynamic model is used to simulate the dynamic changes of cloud resources, while the Cloud Resource State Information provides the current state data of resources such as virtual machines. The core of the resource management and performance prediction model section is the intelligent agent. The intelligent agent decides the actions through an ε-greedy policy to achieve an efficient estimation of the virtual machine parameters. The intelligent agent interacts with the cloud environment, receives state information, and acts based on this information. The model part also contains the Q evaluation network and the Q goal network. These two networks are the key components of the deep Q-network and are used to evaluate the value of different resource management actions. The loss function of the DQN algorithm is defined as the square of the difference between the target Q value and the predicted Q value as shown in Equation (1).

$$L(\theta) = \mathrm{E}_{(s_t, a_t, r_t, s_{t+1}) \sim \mathrm{D}} \left[\left(r_t + \gamma \max_a Q(s_{t+1}, a; \theta^-) - Q(s_t, a_t; \theta) \right)^2 \right] \quad (1)$$

In Equation (1), s_t is the state at time t, a_t the selected action, r_t the immediate reward, and s_{t+1} the next state. Q_θ and Q_{θ^-} denote the current and target Q-networks. Mini-batches (s_t, a_t, r_t, s_{t+1}) are sampled from a replay buffer D. Therefore, the expectation E_{D} is taken over D. To facilitate reproducibility, the training of the DQN agent is configured with the following hyperparameters. The learning rate for the Adam optimizer is set to 1×10^{-4}. A discount factor (γ) of 0.99 is applied to prioritize long-term rewards. The experience replay memory is configured with a capacity of 106 transitions. Mini-batches of 64 samples are drawn from this memory for each training update. The exploration-exploitation strategy is governed by an ϵ-greedy policy, where ϵ is linearly annealed from 1.0 to 0.1 over the first 250,000 steps and subsequently kept constant. The target Q-network is updated every 10,000 training steps to ensure stable learning.

By reducing the discrepancy between the target and anticipated values, the loss function and its related variables in Equation (1) seek to increase the precision of parameter estimation. Therefore, the AI algorithm-based virtual machine parameter estimation process is further demonstrated to clarify the synergistic effect of each key link, as shown in Figure 2.

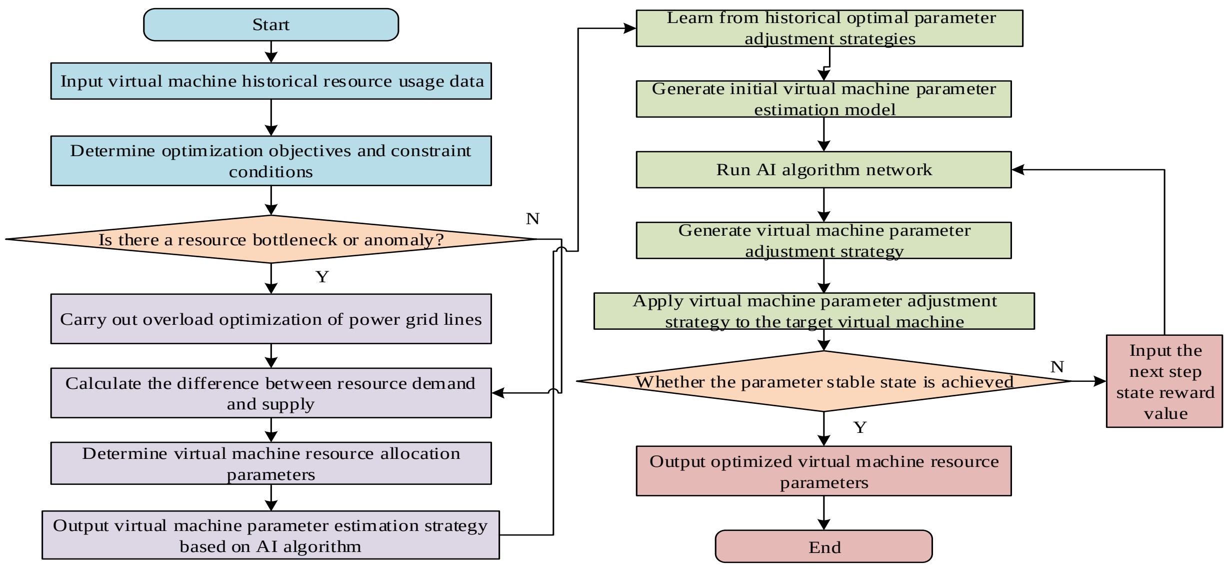

Start
Input virtual machine historical resource usage data
Determine optimization objectives and constraint conditions
Is there a resource bottleneck or anomaly?
N
Y
Carry out overload optimization of power grid lines
Calculate the difference between resource demand and supply
Determine virtual machine resource allocation parameters
Output virtual machine parameter estimation strategy based on AI algorithm
Learn from historical optimal parameter adjustment strategies
Generate initial virtual machine parameter estimation model
Run AI algorithm network
Generate virtual machine parameter adjustment strategy
Apply virtual machine parameter adjustment strategy to the target virtual machine
Whether the parameter stable state is achieved
N
Input the next step state reward value
Y
Output optimized virtual machine resource parameters
End

Figure 2: Flow of virtual machine parameter estimation based on AI algorithm

Figure 2 shows the resource virtual machine parameter estimation process that incorporates cybersecurity considerations. The process starts with the input of historical virtual machine resource usage data. The optimization objective and constraints are determined, followed by determining whether a resource bottleneck or anomaly exists. If so, calculate the difference between resource supply and demand. The virtual machine resource allocation parameters are determined in conjunction with the network security protection requirements. It outputs the virtual machine parameter estimation policy based on AI algorithm to ensure reasonable resource allocation under anomalies. If it does not exist, learn the historical optimal parameter adjustment policy. The initial model is generated and the network of AI algorithms (algorithms incorporating cybersecurity factors) is run to generate an adjustment policy to be applied to the virtual machine. Subsequently, it determines whether the parameter steady state is reached. If it is not reached, it inputs the next state return value to continue adjustment. If it is reached, it outputs the optimized virtual machine resource parameters.

2.2 Secure scheduling management of networked cloud resources

After specifying the parameter estimation method of resource virtual machine based on AI algorithm, the research focuses on the secure scheduling and management mechanism of networked cloud resources. This is to provide further security and efficiency of network cloud resources [18]. In this process, the complete analysis from raw data to anomaly identification is shown in Figure 3.

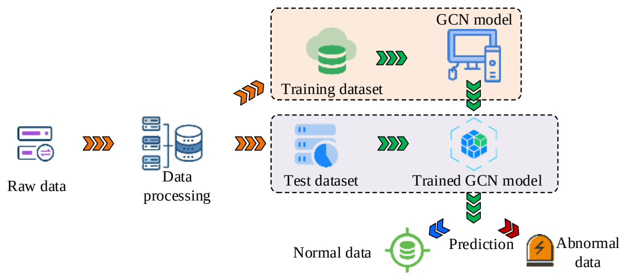


Figure 3: Workflow for training and validating the GCN-based anomaly detection model.

Figure 3 shows the complete analysis from raw data to anomaly identification. The raw data is processed before being partitioned into separate training and testing datasets. The GCN model is trained exclusively on the training dataset to learn the features and patterns of network cloud resources. Crucially, the model's predictive performance and generalization ability are then validated by evaluating its accuracy on an unseen test dataset. This step confirms that the model can distinguish between normal and abnormal data before it is deployed within the secure scheduling system. The raw data first enters the data processing session and after processing is divided into training dataset and test dataset. The training dataset is used to train the GCN model so that it can fully learn the features and patterns of network cloud resources. After the GCN model is trained, the trained model is used to predict and analyze the test dataset to accurately distinguish the normal data from abnormal data. After completing the abnormal identification of network cloud resource data, the secure scheduling mechanism of network cloud resources will be further demonstrated in order to ensure the security and efficient utilization of resources, as shown in Figure 4.

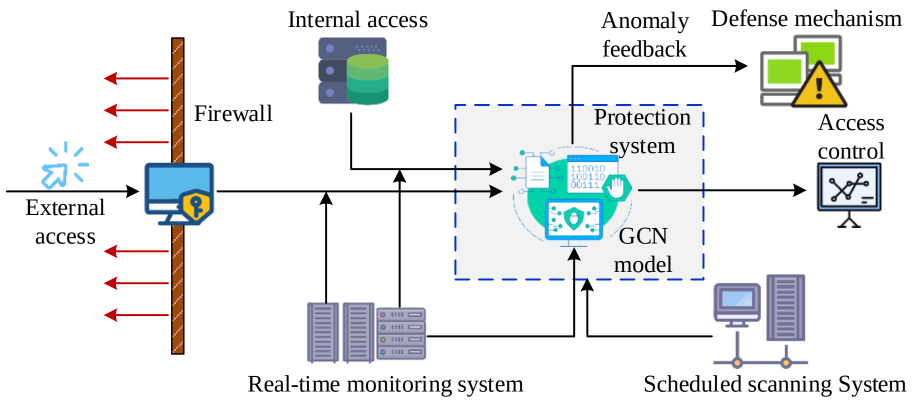


Figure 4: Secure scheduling of network cloud resources

Figure 4 presents the complete process of secure scheduling of network cloud resources. External accesses first pass through the device with security protection and then through the firewall. The firewall screens the access and blocks insecure external access. Internal accesses are then directly connected to the relevant storage and systems. A real-time monitoring system continuously monitors the network cloud resources and a periodic scanning system checks the resources at regular intervals. Both transmit the collected data to the GCN model within the protection system. The GCN model analyzes the data and, once it identifies anomalies, feeds the information to the defense mechanism (which triggers security defenses) and the access control module (which regulates access). The study employs the stochastic gradient descent technique to optimize the detection model's training process in an attempt to maximize the iterative impact of the intrusion detection method on the data content. Equation (2) illustrates the gradient descent method's algorithmic iteration law.

$$\theta_{i+1} = \theta_i - \alpha_i \nabla f_i(\theta_i) \tag{2}$$

θ_{i+1} represents the model's parameters at the $i+1$ th iteration in Equation (2). θ_i stands for the model's current iteration's parameters. The learning rate is shown by α_i. The gradient of the loss function with respect to the parameter θ_i is shown by the symbol $\nabla f_i(\theta_i)$. The stochastic gradient descent approach can be used to lessen the objective function's computing load during the training phase. More efficient data iteration is achieved by transforming the actual descending gradient process into descending estimation through stochastic approximation assignment.

2.3 Network resource security operational performance prediction

After constructing a resource scheduling framework under security constraints, this study further explores how to integrate the AM with the Transformer architecture by fusing it. This is to realize the accurate prediction of the security operation performance of network resources [19]. The effectiveness of secure scheduling decisions relies on accurate prediction of future network states. Therefore, the study establishes the model architecture for predicting the secure operation performance of network resources by fusing the AM, as displayed in Figure 5.

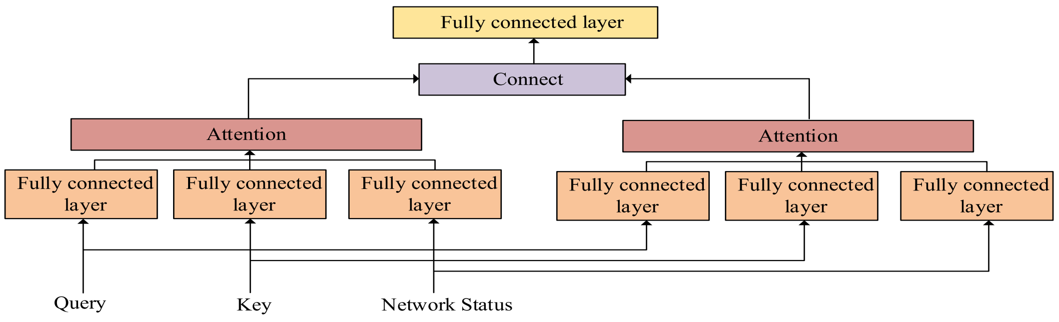


Figure 5: Architecture of a secure operational performance prediction model for network resources incorporating the AM

Figure 5 presents the model structure of the fused AM, where the inputs contain Query, Key, and Network Status that characterizes the network state. These inputs are first processed through the full connectivity layer separately to extract features initially. Subsequently, the AM module focuses on the key information, highlights the features that are important for predicting the secure operational performance of network resources, and weakens the irrelevant information. Next, the processed information is integrated by Connect operation. Finally, the information is input to the full connectivity layer for comprehensive analysis and prediction to realize accurate judgment on the safe operation performance of network resources. The deep interaction between the multi-layer feed-forward network (FFN) and the multi-head (MH) AM further enhances the mining of the intricate temporal relationship between the security risk patterns and the multi-dimensional data of network resources when it is realized that the AM should concentrate on the important characteristics. Figure 6 displays the Transformer-based network resource security operation performance prediction model.

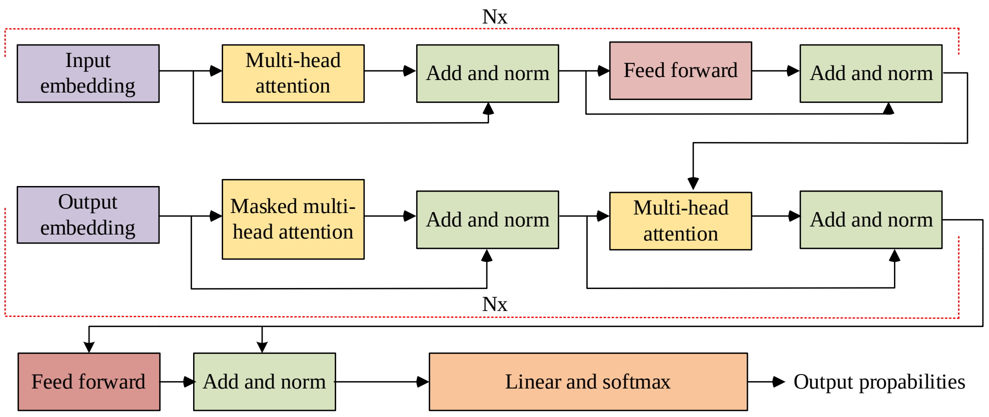


Figure 6: Transformer-based performance prediction model for secure operation of network resources

In Figure 6, the input embedding transforms the multidimensional data of the web resource into an embedded representation that the model can handle. The multi-AM is responsible for capturing the complex associations between different dimensions of data of network resources. It mines the key factors and potential patterns that affect the performance of network security operations. Feedforward network further processes the information processed by the AM. It learns the nonlinear features of network resource data to provide richer feature representation (FR) for performance prediction [20]. Through multiple layers of repeated attention and feedforward computations, the model learns network resource characteristics in depth. The output part is processed by linear transformation (LT) and Softmax to output the predicted probability of the safe operation performance of network resources, thus realizing the prediction of the safe operation state of network resources.

The Transformer architecture's standard self-attention and FFN layers form the foundation of the model. However, its application in this work is specifically tailored for multitask time series forecasting within a multicloud environment. The model utilizes a shared encoder with multiple task-specific heads to concurrently predict three key metrics essential for scheduling: task runtime, energy consumption, and failure risk.

A key architectural modification involves integrating causal convolution layers before the MH self-attention module. This adaptation is critical for effectively processing the time-series data streams from telemetry, such as CPU utilization and network traffic. Causal convolutions are adept at capturing local patterns and short-term temporal dependencies. This is crucial for reacting to sudden load shifts. The model achieves higher prediction accuracy under volatile workloads by combining the local feature extraction capabilities of causal convolutions with the long-range dependency modeling of self-attention. This enhanced predictive capability gives the SBSAD scheduler more reliable and timely information, which improves the quality of its dynamic resource allocation and scheduling decisions.

In the end, the outcomes of the MH AM and completely linked FFN's processing are used to forecast the network resources' safe operating performance. The study names the proposed model as SBSAD.

The SBSAD scheduler operates in a compact single-pass loop. Tasks are mapped to deadline buckets via fixed cutoffs. Online predictions are produced for runtime, energy, and failure risk. A GCN then computes feasibility masks that encode isolation and affinity/anti-affinity before placement. A single priority $P = a/(d - \text{now}) + bs + cD - d\hat{t}$ ranks tasks within each bucket, where d is the deadline, s the security level, D a dependency penalty. $\hat{t}$ the predicted runtime. Constants a, b, c, d are fixed on validation data. The scheduler places the highest-priority feasible tasks under capacity constraints and updates the capacity state. When telemetry indicates degradation or an imminent deadline risk, the scheduler promotes the affected tasks and preempts the lowest-priority ones. Observed durations then feed back to update the predictors.

3 Resource management and performance test results based on AI algorithms

For internal consistency and clarity, all reported results follow a single policy. Unless otherwise stated, percentages denote relative change versus the corresponding baseline and are computed as $\Delta\% = 100 \times (Baseline - Method) / Baseline$. Absolute differences are labeled “Δ abs”. Time-related metrics (response/turnaround) are reported in milliseconds (ms), energy in joules per task (J/task) obtained by integrating host power over time, throughput in tasks/s, and deadline satisfaction as a percentage (%). Values are rounded to one decimal place for percentages, times, and energy, and to the nearest integer for counts. Table totals and ratios are recomputed from unrounded means to avoid rounding drift. Unless noted otherwise, confidence intervals refer to 95% Student-t intervals over 20 seeds, and arrows in tables indicate the preferable direction (↓ latency/energy, ↑ satisfaction/throughput). Baselines use identical traces, capacity limits, and policy constraints to enable consistent comparisons within the same measurement context.

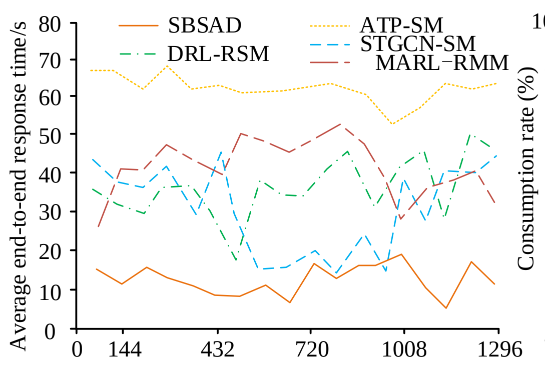
3.1 MC resource management scheduling results

To enhance reproducibility, the experimental setup is summarized concisely in Table 2.

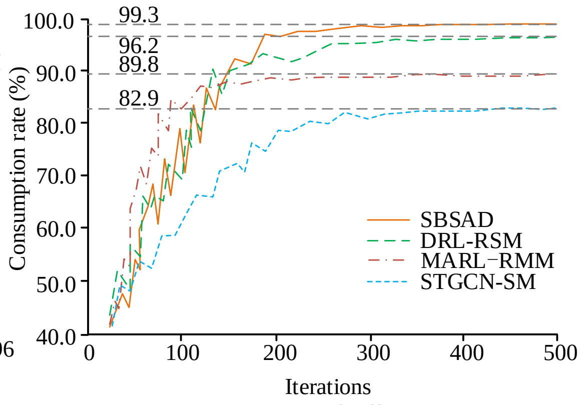
Table 2: CloudSim environment, workload, variability, and evaluation (concise summary)

Category	Item	Setting
Simulator	Platform	CloudSim (datacenter, VM, cloudlet, network modules)
Topology	Datacenters & Hosts	3 datacenters; 100 hosts each
Hosts	Type S / Type L	S: 16 vCPUs @2.5 GHz, 64 GB RAM, 1 Gbps; L: 32 vCPUs @2.5 GHz, 128 GB RAM, 10 Gbps
Scheduling	Host / Cloudlet	TimeShared host VM scheduler; TimeShared cloudlet scheduler
VMs	t1 / t2 / t3	t1: 2 vCPUs, 4 GB, 2,000 MIPS; t2: 4 vCPUs, 8 GB, 4,000 MIPS; t3: 8 vCPUs, 16 GB, 8,000 MIPS
Placement	Capacity & Policy	Capacity-limited with isolation and affinity/anti-affinity rules
Workload	Mix & Lengths	60% CPU-bound, 20% I/O-bound, 20% mixed; log-normal lengths (median 3×1093\times10^9 MI, σ=0.6\sigma=0.6)
I/O	Input/Output Size	50–500 MB
Arrivals	Process	Poisson arrivals at 6 tasks/s
Deadlines	Setting	Per-task deadline =1.2× predicted runtime (non-trivial but feasible pressure)
Security	Levels	$s \in \{1,2,3\}$ used for isolation/affinity constraints
Telemetry	Updates	Online metrics refresh predictions for runtime, energy, failure risk
Variability	Load & Network	Aggregate load $\{0.5\times, 1.0\times, 1.5\times\}$ of host capacity; latency mean 3 ms (SD 1 ms)
Failures	Injection	Disabled (avoid confounding); predicted risk still consumed by scheduler
Repetitions	Seeds & runs	20 random seeds per load → 60 runs total
Metrics	Primary	Turnaround time, deadline satisfaction rate, energy per task, preemption count
Baselines	Fairness Controls	Identical capacity limits, policies, and workload traces across all policies

The study selects deep reinforcement learning-driven resource scheduling model (DRL-RSM), spatio-temporal graph convolutional network-based scheduling model (STGCN-SM), multi-agent reinforcement learning resource management model (MARL-RMM), attention-fused transformer for prediction and scheduling model (ATP-SM), and the proposed SBSAD method for comparison. The experiment relies on a high-performance server configured with Intel Xeon Gold 6248 processor, 128GB RAM, and NVIDIA Tesla V100 GPU. The operating system is Ubuntu 18.04. A MCE is constructed with the help of the CloudSim simulation platform, which accurately simulates the dynamic changes of the cloud resources and task loads. The performance comparison of different models in MC resource management scheduling is shown in Figure 7.



(a) Average end-to-end response time of different models changes with test time



(b) Consumption rate of different models changes with the number of iterations

Figure 7: Performance comparison of different models in MC resource management scheduling

In Figure 7(a), the average end-to-end response time (AETERT) of the research-proposed SBSAD method is relatively low at different test time points. This indicates that it is able to respond to tasks more quickly in MC resource management scheduling. It effectively improves the scheduling delay and has a significant advantage over other models. Figure 7(b) presents the variation of different model consumption rates with the number of iterations.

To provide a robust evaluation, the performance of the proposed SBSAD model is compared against four baseline models under identical experimental conditions. All reported results are mean values averaged over 20 independent runs. The standard deviation (SD) is included to indicate the consistency and reliability of the measurements. This statistical approach guarantees that the differences observed between models are significant and not the result of random variation. The key performance metrics are summarized in Table 3.

Table 3: Comparative performance metrics of scheduling models (Mean±SD over 20 runs)

Metric	SBSAD (Proposed)	DRL-RSM	STGCN-SM	MARL-RMM	ATP-SM
Turnaround time (ms) ↓	118.4±5.2	145.7±9.8	162.1±11.5	155.3±13.1	151.8±10.4
Deadline satisfaction (%) ↑	96.2±1.8	88.5±3.5	85.4±4.1	87.1±4.8	89.3±3.2
Energy per task (J/task) ↓	25.6±1.1	31.2±2.4	34.8±2.9	33.5±3.2	30.7±2.1
Preemption count ↓	15.3±2.5	45.8±6.7	51.2±7.2	60.1±8.5	41.5±6.1

Note: Arrows (↓↑) indicate the preferable direction for each metric.

The aggregated results in Table 3 clearly demonstrate the superior performance of the SBSAD model across all evaluated metrics. With an average turnaround time of 118.4 ms, SBSAD significantly outperforms the next best baseline, DRL-RSM, which has an average turnaround time of 145.7 ms. Furthermore, its deadline satisfaction rate is the highest at 96.2%, indicating its effectiveness in meeting task constraints. The model is also more efficient. It has the lowest energy consumption per task (25.6 J/task) and a drastically reduced preemption count (15.3), compared to all other methods. The small standard deviations associated with SBSAD's metrics across the 20 runs underscore the consistency and stability of its performance, thereby reinforcing the statistical significance of these improvements.

3.2 Results of predicting the operational performance of network resource security management

For clarity, model names are abbreviated as follows: DRL-RSM (deep-reinforcement-learning resource scheduling model), STGCN-SM (spatio-temporal graph convolutional scheduling model), MARL-RMM (multi-agent reinforcement-learning resource management model), and ATP-SM (attention-based Transformer scheduling model). Representative implementations and citations are summarized in related work.

The study tested the above models on two datasets, CloudSim Simulation Dataset (CloudSim Dataset) and BigDataBench: Big Data Benchmark (BigDataBench). A comparison of the results of the predicted operational performance of security management of network resources for different models is shown in Figure 8.

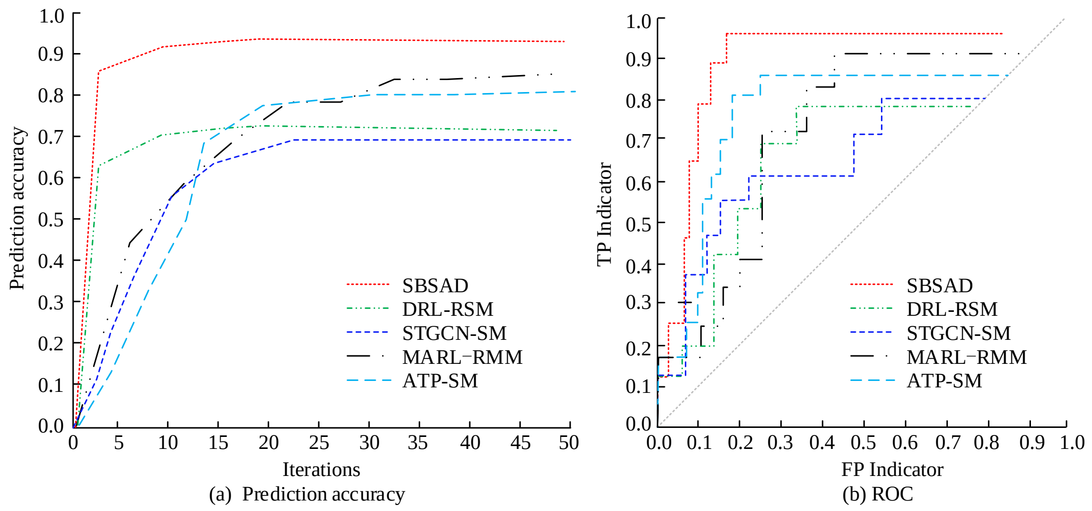


Figure 8: Comparison of operational performance prediction results of different models for network resource security management Abbreviations: DRL-RSM=deep-reinforcement-learning resource scheduling model; STGCN-SM=spatio-temporal GCN scheduling model; MARL-RMM=multi-agent RL resource management model; ATP-SM=attention-based Transformer scheduling model.

The prediction accuracy for various models varies with the quantity of iterations, as shown in Figure 8(a). The SBSAD method proposed in the study shows a rapid increase in prediction accuracy and stabilizes at a high level during the iteration process. Compared with other models, it can learn the patterns of network resources' secure operation performance more quickly and precisely, and significantly improve the prediction accuracy. The ROC curve analysis in Figure 8(b) shows that the curve of SBSAD is located at the top. It means that SBSAD has a higher true positive indicator with the same false positive indicator (FP Indicator). It can more effectively identify the real state of network resource security operation and reduce the misjudgment situation. The variation of error and loss of each model with training rounds under different datasets is shown in Figure 9.

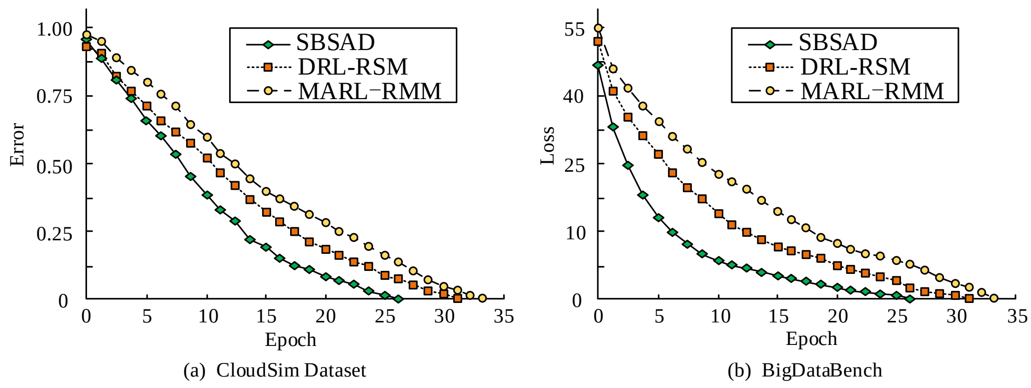


Figure 9: Variation of error and loss with training rounds for each model under different datasets Abbreviations: DRL-RSM=deep-reinforcement-learning resource scheduling model; STGCN-SM=spatio-temporal GCN scheduling model; MARL-RMM=multi-agent RL resource management model; ATP-SM=attention-based Transformer scheduling model.

Figure 9(a) demonstrates the variation of error with training rounds for different models on CloudSim Dataset. Figure 9(b) demonstrates the variation of loss with Epoch for different models on BigDataBench. The error of SBSAD method proposed in the study decreases rapidly with increasing Epoch on CloudSim dataset. Compared to the rest of the models, it has a lower final error value and more accurate prediction. On the BigDataBench dataset, the loss value of SBSAD decreases rapidly. Moreover, under the same Epoch, the loss value is lower than the other models. Its training effect is better and it can learn the characteristics of network resource security management operation performance more efficiently. Table 4 displays the performance forecast results for the security scheduling operation and MC resource management.

Table 4: Predicted performance of MC resource management and secure scheduling operations

Test scenario	Scheduling model	Network bandwidth utilization (%)	Task scheduling rate improvement (%)	Task quantity reduction	Prediction accuracy improvement (%)	Predicted response time (ms)	System stability (%)
Scenario 1	SBSAD and multi-task learning model	85.27	6.52	18	14.8	110.65	97.89
Scenario 2	Traditional scheduling method	81.58	0.00	0	0	124.32	94.35
Scenario 3	SBSAD and multi-task learning model	89.12	6.52	18	14.8	105.21	98.03
Scenario 4	Traditional scheduling method	77.63	0.00	0	0	131.54	93.12

Interpreting Table 4, performance gains are linked to where SBSAD changes the decision process rather than to aggregate scores alone: The DQN-informed estimates reduce queueing and late starts. This contributes to the 6.52% increase in deadline satisfaction despite identical capacity limits. GCN-derived feasibility masks eliminate security and affinity conflicts before packing. This explains why the number of required scheduling tasks is reduced by 18 and why stability is higher. The attention-based, multi-task predictor reduces runtime prediction error by 14.8%. This results in shorter predicted response times and fewer preemptions. Improvements in bandwidth utilization (e.g., 85.27%–89.12% in Scenarios 1–3) reflect conflict-aware placement rather than mere over-commitment. Note that some VM-level entries in Table 2 favor DRL-RSM for isolated metrics, such as single-VM delay. However, end-to-end turnaround time and deadline satisfaction favor SBSAD because it handles conflicts and risk prior to packing.

4 Discussion

Using the identical traces and capacity controls described in Section 4, SBSAD decreased the end-to-end turnaround time and increases the satisfaction rate of meeting deadlines, while reducing the number of late or dropped tasks and preemptions. These gains stemmed from stage-based deadline bucketing and a single scalar priority that considered urgency, security, dependency, and predicted runtime. This was in contrast to DRL-RSM, where reward shaping and global queues make deadline pressure less explicit. Unlike STGC-SM that relied on static graph features or offline prediction, STGC-SM
directly consumed online multitasking prediction (runtime/energy/fault risk) within the scheduler.

Therefore, its response to short-term load changes was slower. Moreover, by preempting cost feedback for cross stage promotion, it avoided the convergence overhead observed in the sudden arrival of MARL-RMM. In practical cloud stacks, integration was straightforward: In Kubernetes, a scheduler framework plugin or extender could replace the native scoring with a scalar priority while applying GCN-derived feasibility masks in the filter phase. The predictor was deployed as a telemetry-driven sidecar. Equivalent behavior in OpenStack-like platforms followed from custom host filters and weights. Operational interpretation revealed that improved accuracy reduced queue build-up and deadline violations for batch and ML workflows in shared clusters. It also lowered interference in isolation-critical, multi-tenant SaaS and stabilized edge-to-cloud video and ETL pipelines during short-term surges. Limitations remain: Reliance on predictor quality and stable telemetry can bias prioritization. The CloudSim environment simplifies network and storage contention compared to production systems. Preemption cost and fairness are modeled beyond deadline satisfaction at a coarse level. These factors define low-risk deployment zones and indicate where additional engineering hardening or A/B trials are recommended.

Future work will entail evaluating SBSAD on production-like clusters with realistic network and storage contention. Additionally, the predictor will be augmented with calibrated uncertainty to de-bias the priority score under telemetry drift. Furthermore, fairness and preemption cost will be incorporated into a multi-objective function that goes beyond deadline satisfaction. Finally, cross-cluster portability and autoscaling interactions in Kubernetes and OpenStack deployments will be studied to broaden applicability.

5 Conclusion

This study introduced the SBSAD model, an AI-driven framework that improved MC resource management. The model integrated a DQN for parameter estimation, a GCN for security policy enforcement, and a Transformer for multi-task performance prediction. The experimental results showed that the SBSAD method significantly improves scheduling performance. It reduced the AETERT by 15%-20% and increased the task scheduling rate by 6.52%, compared to baseline models. Furthermore, the model improved prediction accuracy by 14.8% while maintaining system stability above 97.89%.

Although the proposed approach shows promise, future work should address its current limitations, such as its adaptability to heterogeneous environments and its ability to generalize against novel threats. Future optimizations will focus on lightweight neural network architectures and introducing federated learning to improve the practical, real-world applicability of secure MC systems.

References

[1] Singla D, Verma N. Performance Analysis of Authentication System: A Systematic Literature Review. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 2024, 17(7): 47-67. https://doi.org/10.2174/0126662558246531231121115514

[2] Globa L, Kartashov A. Optimizing distributed data storage in multi - cloud environments: algorithmic approach. Information and Telecommunication Sciences, 2024, (2):4 - 12. https://doi.org/10.20535/2411-2976.22024.4-12

[3] Mohammad O K J, Seno M E, Dhannoon B N. Detailed cloud linear regression services in cloud computing environment[J]. Informatica, 2024, 48(2): 185-194. https://doi.org/10.31449/inf.v48i12.6771

[4] Umoga U J, Sodiya E O, Ugwuanyi E D, Jacks B S, Lottu O A, Daraojimba O D. Exploring the potential of AI - driven optimization in enhancing network performance and efficiency. Magna Scientia Advanced Research and Reviews,2024,10(1):368 - 378. https://doi.org/10.30574/msarr.2024.10.1.0028

[5] Saif M A N, Niranjan S K, Murshed B A H, Al - Ariki H D E, Abdulwahab H M. Multi - agent QoS - aware autonomic resource provisioning framework for elastic BPM in containerized multi - cloud environment. Journal of Ambient Intelligence and Humanized Computing,2023,14(9):12895 - 12920. https://doi.org/10.1007/s12652-022-04120-4

[6] Ahmadi S. Security implications of edge computing in cloud networks. Journal of Computer and Communications,2024,12(02):26 - 46. https://doi.org/10.4236/jcc.2024.122003

[7] Jeyaraj R, Balasubramaniam A, MA A K, Guizani N, Paul A. Resource management in cloud and cloud - influenced technologies for internet of things applications. ACM Computing Surveys,2023,55(12):1 - 37. https://doi.org/10.1145/3571729

[8] Basha H A, Anilkumar B H, Swetha G, Reddy R, Manoli S. Real - time challenges and opportunities for an effective resource management in multi - cloud environment. SN Computer Science,2024,5(2):238 - 257. https://doi.org/10.1007/s42979-023-02578-3

[9] Alonso J, Orue - Echevarria L, Casola V, Torre A I, Huarte M, Osaba E. Understanding the challenges and novel architectural models of multi - cloud native applications - a systematic literature review. Journal of Cloud Computing,2023,12(1):6 - 40. https://doi.org/10.1186/s13677-022-00367-6

[10] Putri A. Multi - Cloud Strategies for Managing Big Data Workflows and AI Applications in Decentralized Government Systems. Journal of Computational Intelligence for Hybrid Cloud and Edge Computing Networks,2025,9(1):1 - 11.

[11] Anh N H. Hybrid Cloud Migration Strategies: Balancing Flexibility, Security, and Cost in a Multi - Cloud Environment. Transactions on Machine Learning, Artificial Intelligence, and Advanced Intelligent Systems,2024,14(10):14 - 26.

[12] Azizi S, Farzin P, Shojafar M, Rana O. A scalable and flexible platform for service placement in multi - fog and multi - cloud environments. The Journal of supercomputing,2024,80(1):1109 - 1136. https://doi.org/10.1007/s11227-023-05520-9

[13] Khan A A, Bourouis S, Kamruzzaman M M, Hadjouni M, Shaikh Z A, Laghari A A. Data security in healthcare industrial internet of things with blockchain. IEEE Sensors Journal,2023,23(20):25144 - 25151. https://doi.org/10.1109/JSEN.2023.3273851

[14] De Alwis C, Porambage P, Dev K, Gadekallu T R, Liyanage M. A survey on network slicing security: Attacks, challenges, solutions and research directions. IEEE Communications Surveys & Tutorials,2023,26(1):534 - 570. https://doi.org/10.1109/COMST.2023.3312349

[15] Ma J, Zhu C, Fu Y, Zhang H, Xiong W. Reliable task scheduling in cloud computing using optimization techniques for fault tolerance[J]. Informatica, 2024, 48(2): 159-170. https://doi.org/10.31449/inf.v48i23.6901

[16] Kazmi S H A, Qamar F, Hassan R, Nisar K, Chowdhry B S. Survey on joint paradigm of 5G and SDN emerging mobile technologies:

Architecture, security, challenges and research directions. Wireless Personal Communications,2023,130(4):2753 - 2800. https://doi.org/10.1007/s11277-023-10402-7

[17] Li S, Yang Y, Zhang L. Knowledge Representation Learning Method Based on Semantic Enhancement of External Information. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 2024, 17(7): 68-84. https://doi.org/10.2174/0126662558271024231122045127

[18] Patel A D, Jhaveri R H, Shah K A, Patel A D, Rathore R S, Paliwal M, et al. Security Trends in Internet-of-things for Ambient Assistive Living: A Review. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 2024, 17(7): 18-46. https://doi.org/10.2174/0126662558270314231129051456

[19] Ullah A, Muhammad F, Mehmood K, et al. Improving load balancing efficiency in cloud data centers through hybrid metaheuristic algorithms[J]. Informatica, 2025, 49(2): 121-136. https://doi.org/10.31449/inf.v49i28.8860

[20] Wang X, Mei J, Cui S, Wang C X, Shen X S. Realizing 6G: The operational goals, enabling technologies of future networks, and value - oriented intelligent multi - dimensional multiple access. IEEE Network,2023,37(1):10 - 17. https://doi.org/10.1109/MNET.001.2200429