AUT Journal of Electrical Engineering
10.22060/EEJ.2026.25455.5936

An Attention-Driven Deep Reinforcement Learning Framework for Energy-Efficient and Service-Level Agreement-Aware Cloud Task Scheduling

Rahul Bhatt[1], Ritika Mehra[2], Kamal Upreti[3*]

[1]Ph.D Scholar, School of Engineering & Computing, Dev Bhoomi Uttarakhand University, Dehradun, Uttarakhand, India
[2]School of Engineering & Computing, Dev Bhoomi Uttarakhand University Dehradun, Uttarakhand, India
[3]Department of Computer Science, Christ University, Delhi NCR Campus, Ghaziabad, India

*Corresponding Mail ID : kamalupreti1989@gmail.com

Abstract:

Dynamic cloud and edge-cloud platforms require task schedulers that can respond to stochastic workloads while minimizing energy use and preserving service-level agreement reliability. This study proposes an attention-driven deep reinforcement learning framework for energy-efficient and service-level agreement-aware cloud task scheduling. The framework combines lightweight convolutional neural network and long short-term memory-based spatial-temporal feature extraction with a multi-head self-attention actor-critic decision module. The convolutional neural network and long short-term memory components capture local virtual machine workload patterns, whereas self-attention models global virtual-machine-to-virtual-machine dependencies for parallel and context-aware scheduling. The scheduling problem is formulated as a Markov decision process using a 242-dimensional virtual-machine-level state representation, probabilistic virtual-machine-to-host assignment actions, and a multi-objective reward function covering makespan, energy consumption, operational resource cost, and service-level agreement penalties. Experiments were conducted in a heterogeneous CloudSim environment with 100 hosts and 100 virtual machines. The proposed framework achieved a normalized makespan of approximately 0.85, a 14.4% reduction in total energy consumption, consistently low service-level agreement violation behavior, and controlled migration activity. Logged analysis further showed a response time of 10.0000 milliseconds per completed task or

virtual machine event, supporting interval-based real-time feasibility. Cost is treated as an operational reward component, not as a standalone billing analysis.

Keywords:

Cloud Computing; Task Scheduling; Deep Reinforcement Learning; Multi-Head Self-Attention; Actor-Critic Learning; Energy Efficiency; Service-Level Agreement-Aware Scheduling; CloudSim

1. Introduction

Cloud computing has become a dominant computing paradigm because of its scalability, elasticity, on-demand resource provisioning, and flexible service delivery. In cloud environments, a large number of tasks must be allocated to available computing resources while maintaining performance efficiency and quality-of-service requirements. Task scheduling is therefore a central resource-management problem, where incoming tasks are mapped to suitable virtual machines (VMs) to optimize makespan, energy consumption, resource utilization, cost-related operational efficiency, and service-level agreement (SLA) compliance. Since the cloud task scheduling problem is multi-objective, dynamic, and NP-complete, designing an efficient scheduler remains challenging, particularly under heterogeneous infrastructure and stochastic workload conditions [1], [2]. Among these objectives, energy efficiency has become increasingly important due to the rising power consumption of cloud data centers and the growing emphasis on green computing. It is discussed in detail that, 67 cloud task scheduling algorithms are available, and the significant issues that are still not resolved are energy-aware scheduling and the need to respond to QoS constraints simultaneously [3], [4], [5]. One of the primary issues that should be considered in cloud computing environment is task scheduling to enhance the system performance and maximize the satisfaction of cloud consumers. Despite numerous available task scheduling algorithms, the current solutions are primarily aimed at reducing the overall completion time with no regard to the workload balancing. The literature has examined a tremendous number of scheduling strategies. The heuristic and meta-heuristic approaches to

10.22060/EEJ.2026.25455.5936

the problem, including HEFT variants, genetic algorithms [6], particle swarm optimization, and its variations [7], [8], and hybrid swarm-intelligence solutions, have shown better makespan and resource use given a controlled environment. Yet, these techniques usually depend on fixed rules, assumptions or large-scale parameter adjustment, so they are less adaptable to real-time changes in workload. Cloud systems based on workflow have also been analyzed in energy conscious and reliability conscious schedule. An energy efficient yet reliability conscious workflow scheduling algorithm was introduced in [9], [10], [11], which combines task ranking, task clustering and slack recovery to minimize energy usage yet ensure system reliability. Though these approaches work well in scientific workflows, they rely greatly on task dependencies that are predetermined and would not as effectively work in complex dynamic cloud environments where arrivals are unpredictable. More recent research has extended the scheduling research to inter-cloud, edge-cloud and IoT-cloud environments. It was highlighted that the current solutions have difficulties that are not generalized in a single-cloud, multi-cloud, and mobile cloud environment, especially when there are several conflicting requirements to be fulfilled at the same time [12], [13]. There are trust-aware [14], [15], [16], SLA-based[17], and deadline-sensitive scheduling strategies [18]. These add constraints, yet complicate and add overhead to the systems. Recently, methods of reinforcement learning (RL) and deep reinforcement learning (DRL) have been explored in attempts to address the shortcomings of non-dynamic scheduling methods. Schedulers based on DRL allow making adaptive decisions based on the feedback of the system [19]. Nevertheless, a significant portion of the solutions to DRL currently available are based on dynamic or locally informed decision making, thus being less able to model large-scale cloud systems with VM-to-VM dependencies.

Therefore, the novelty of this study lies in designing an attention-driven actor–critic scheduling framework that explicitly models global VM-to-VM dependencies rather than relying only on sequential recurrent representations or locally informed scheduling decisions. Unlike conventional actor–critic schedulers, the proposed A2-DRL framework integrates compact spatial–temporal feature extraction, multi-head self-attention, residual learning, and a multi-objective reward design within a CloudSim-based scheduling

environment. This design enables parallel contextual reasoning over heterogeneous virtual machines while jointly considering energy efficiency, SLA reliability, response behavior, and migration stability. The proposed framework is therefore positioned as a global context-aware reinforcement learning scheduler for dynamic cloud task scheduling rather than a simple extension of existing actor–critic models.

2. Research Gap, Motivation and Contribution

2-1- Research Gap

Although cloud, fog, and edge–cloud task scheduling has been widely studied, existing approaches still face important limitations when applied to dynamic, heterogeneous, and SLA-sensitive cloud environments. Heuristic and metaheuristic scheduling methods are computationally efficient and can improve makespan, resource utilization, and energy consumption under specific workload settings [20], [21], [21]. However, these methods commonly depend on fixed rules, repeated population-based search, or manually tuned parameters. As a result, they often show limited adaptability when task arrivals, VM availability, resource contention, and SLA constraints vary over time [22], [23]. Mathematical optimization-based scheduling approaches, including MILP and thermal-aware formulations, can generate structured scheduling decisions and reduce energy consumption under controlled conditions [24], [25]. However, such approaches generally require prior knowledge of task characteristics, resource states, or system constraints, and their computational cost can increase significantly in large-scale cloud environments. Similarly, conventional edge–cloud and fog scheduling methods improve task allocation and resource utilization, but many of them are not designed to learn scheduling policies from continuously changing workload feedback [26], [27]. Recent reinforcement learning and deep reinforcement learning schedulers have improved adaptability compared with static scheduling methods [28]–[29]. Nevertheless, many DRL-based schedulers still rely on recurrent, sequential, or locally informed decision-making. This limits their ability to explicitly model global VM-to-VM dependencies across the complete cloud infrastructure [30]. Recent attention-based, offline DRL, and Transformer-enhanced schedulers have advanced dependency modeling in cloud–edge manufacturing, logistics-involved scheduling, and IoT

edge–cloud applications [31], [32]. However, these studies are mainly oriented toward manufacturing-service scheduling, offline historical scheduling, or IoT application scheduling rather than VM-level SLA-aware cloud task scheduling.

Therefore, a clear research gap remains for a lightweight, reproducible, and globally context-aware DRL scheduler that can jointly model VM-to-VM resource dependencies, support parallel scheduling decisions, reduce energy consumption, maintain SLA reliability, and control migration overhead in dynamic CloudSim-based cloud environments. This gap motivates the development of the proposed A2-DRL framework.

2-2- Motivation of the Study

The motivation of this study arises from the need for adaptive and context-aware scheduling mechanisms in modern cloud and edge–cloud environments. Recent DRL-based schedulers, including IA3C-based and multi-objective RL models, have improved makespan and energy efficiency; however, many of them still depend on sequential recurrent decision-making and have limited system-wide visibility [24], [25]. Hybrid RL-metaheuristic and RL-clustering approaches can improve load distribution, but they often introduce additional computational and architectural complexity, which may reduce scalability in real-time scheduling scenarios [26], [27].

Cloud, fog, and edge scheduling studies further show that workload heterogeneity and spatial–temporal variability require scheduling models capable of parallel reasoning and global dependency modeling [28],[33] . These limitations motivate the integration of attention mechanisms into a DRL-based scheduling framework. Self-attention enables the scheduler to model global VM-to-VM interactions and make context-aware decisions in parallel. Therefore, the proposed A2-DRL framework is designed to improve energy efficiency, SLA reliability, and scheduling stability under dynamic workload conditions.

2-3- Research Objectives and Novel Contributions

10.22060/EEJ.2026.25455.5936

The main objective of this study is to develop an adaptive, energy-efficient, and SLA-aware cloud task scheduling framework that can operate under dynamic and heterogeneous workload conditions. Unlike conventional actor–critic schedulers that mainly depend on sequential recurrent representations or locally observed resource states, this study aims to design a scheduler that reasons over the global VM environment and learns scheduling decisions from system-wide contextual relationships. The proposed A2-DRL framework is not presented as a simple extension of actor–critic learning. Its novelty lies in combining lightweight spatial–temporal feature extraction, multi-head self-attention, residual learning, and multi-objective reinforcement learning into a unified scheduling architecture. This enables the scheduler to capture local workload trends, model global VM-to-VM dependencies, and make parallel VM–host assignment decisions while considering energy consumption, SLA penalty, makespan, and operational resource cost.

The specific objectives of this study are as follows:

1. To formulate cloud task scheduling as a Markov Decision Process using a VM-level state representation, probabilistic VM–host assignment action space, and multi-objective reward function.
2. To design an attention-driven actor–critic scheduler capable of modeling global VM-to-VM dependencies instead of relying only on sequential recurrent decision-making.
3. To improve energy efficiency and SLA reliability under dynamic workloads through attention-guided workload placement and controlled migration behavior.
4. To evaluate the proposed scheduler using a heterogeneous CloudSim-based environment with statistical dispersion, response-time analysis, migration behavior, and comparative benchmarking against heuristic, DRL, attention-based DRL, and Transformer-enhanced scheduling methods.

The novel contributions of this study are summarized as follows:

10.22060/EEJ.2026.25455.5936

- **Global VM-to-VM contextual scheduling:** A multi-head self-attention mechanism is embedded within the actor–critic scheduling pipeline to explicitly learn global relationships among heterogeneous VMs. This enables system-wide context-aware decision-making rather than isolated or sequential resource allocation.

- **Hybrid local–global representation learning:** A lightweight CNN-LSTM feature extraction stage is used to capture local spatial–temporal workload behavior, while the attention block captures global resource dependencies. This hybrid design improves representation quality without using a full Transformer encoder.

- **Parallel probabilistic scheduling action design:** The framework represents scheduling actions as a VM–host probability matrix, enabling parallel evaluation of assignment decisions across the cloud infrastructure instead of one-step sequential scheduling.

- **Multi-objective SLA-aware reward formulation:** The reward function jointly considers makespan, energy consumption, operational resource cost, and SLA penalties, allowing the scheduler to reduce energy use without ignoring service reliability.

- **Reproducible CloudSim-based validation:** The study reports simulation configuration, response-time behavior, interval-level statistical dispersion, migration trends, and comparison with recent attention-based and Transformer-enhanced DRL schedulers, directly addressing reproducibility and state-of-the-art comparison concerns.

- **Clear positioning beyond existing DRL schedulers:** Compared with heuristic, recurrent DRL, attention-based cloud manufacturing scheduling, offline Decision Transformer scheduling, and Transformer-enhanced edge–cloud IoT scheduling approaches, the proposed A2-DRL framework specifically targets VM-level, SLA-aware, energy-efficient cloud task scheduling with global context reasoning.

3. Methodology

3-1- Overall System Architecture

The framework of the proposed Attention-Aware Deep Reinforcement Learning (A2-DRL) (Fig. 1) is a distributed agent-environment framework to guarantee scalability, modularity, and realistic cloud simulation. The architecture separates the cloud infrastructure management with the process of making intelligent decisions and both parts of the architecture can work on their own but share information in real-time.

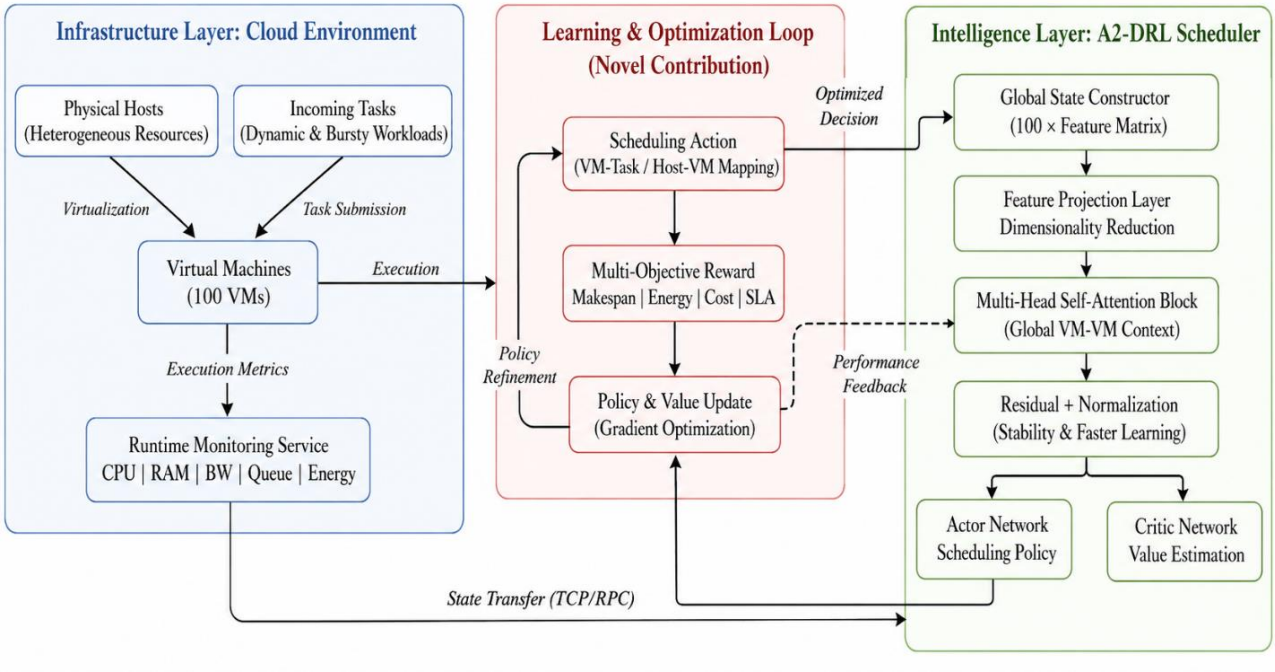

Fig. 1. System Framework of the Proposed A2-DRL Scheduler

The cloud environment is deployed on CloudSim (Java), the responsibility of which is to simulate the physical hosts, virtual machines (VMs), and the dynamics of task execution. The environment in the present configuration comprises of 100 physical hosts and 100 heterogeneous virtual machines each of which will have different CPU, memory, and bandwidth capacities. CloudSim is running on a constant monitoring of system statuses which include resource usage, queues, and execution status.

The intelligent scheduler is a Python application based on PyTorch wherein the proposed A2-DRL agent is located. During the interval of every scheduling, the CloudSim environment sends the global system state to the DRL agent via a minimal TCP/RPC communication bridge. According to this state, the agent transforms the information through its attention-conscious neural architecture and produces an optimal scheduling action in the form of VM task or host VM assignment decisions.

The chosen action will then be returned into the CloudSim environment that will implement the scheduling decision and calculate the output of performance metrics, such as makespan, energy use, cost, and SLA compliance. These measures are then made to produce a reward signal on which the A2-DRL agent learns. This feedback mechanism keeps working during the simulation allowing dynamic workload-based adaptive and data-driven scheduling.

3-2- Problem Definition

The dynamic, stochastic and multi objective nature of resource allocation in heterogeneous cloud environments is modeled by the cloud task scheduling problem as a Markov Decision Process (MDP). Table 1 provides the full formulation of the scheduling problem that includes system features, MDP elements, as well as optimization goals in the proposed A2-DRL model.

Table 1. Unified Problem Definition for Cloud Task Scheduling

Category	Element	Description in A2-DRL Framework
System Characteristics	Task Arrival Pattern	Dynamic and stochastic arrival of independent tasks
	Resource Environment	Cloud infrastructure modeled with 100 physical hosts and 100 heterogeneous virtual machines
	VM Heterogeneity	VMs differ in CPU capacity, memory, bandwidth, and energy profile
MDP Formulation	State (S)	Global system state comprising CPU utilization, memory availability, bandwidth usage, queue length, and workload history of all VMs

10.22060/EEJ.2026.25455.5936

	Action (A)	Scheduling decisions specifying task-to-VM or host–VM assignments
	State Transition (P)	Change in system state after task execution and resource allocation
	Reward (R)	Multi-objective feedback based on makespan, energy consumption, resource cost, and SLA compliance
	Policy (π)	Scheduling strategy learned by the A2-DRL agent to optimize long-term performance
Optimization Objectives	Makespan Minimization	Reduce overall task completion time across the cloud system
	Energy Minimization	Lower total power consumption of physical hosts and VMs
	Cost Minimization	Reduce monetary cost associated with resource usage
Problem Nature	Scheduling Complexity	Multi-objective, non-linear, and time-varying decision process
	Scheduling Requirement	Adaptive, real-time, and scalable scheduling decisions

3-3- State Space Modeling and 242-Dimensional Feature

The effectiveness of a DRL-based cloud scheduler strongly depends on the completeness and consistency of the state representation. In the proposed A2-DRL framework, the cloud state is represented at the VM level so that the scheduler can jointly observe resource availability, workload pressure, historical behavior, and task-allocation context before producing a scheduling decision.

At decision step t, the global cloud state is represented as:

$$s_t \in \mathbb{R}^{N \times F} \quad \text{Eq. (1)}$$

where $N = 100$ denotes the number of virtual machines and $F = 242$ denotes the feature dimension associated with each VM. Therefore, the complete state matrix has a size of 100×242, allowing the scheduler to evaluate all VMs within a single decision cycle.

10.22060/EEJ.2026.25455.5936

The 242-dimensional VM-level feature vector is constructed by combining two complementary feature groups. The first group contains 126 CNN-side spatial/resource features, which describe the current VM and host-side scheduling context, including CPU utilization, memory availability, bandwidth status, VM–host mapping indicators, queue behavior, and resource-availability signals. The second group contains 116 LSTM-side temporal/workload features, which represent workload-history and time-dependent scheduling information, including recent task arrivals, resource-usage trends, migration-sensitive workload variation, and execution-state changes. Thus, the VM-level state vector is expressed as:

$$s_i^t = [\mathrm{x}_{\mathrm{i,CNN}}^{\mathrm{t,126}}; \mathrm{x}_{\mathrm{i,LSTM}}^{\mathrm{t,116}}] \quad \text{Eq. (2)}$$

where $x_{i,CNN}^{t,126}$ is the 126-dimensional spatial/resource feature vector and $x_{i,LSTM}^{t,116}$ is the 116-dimensional temporal/workload feature vector for VM i. The resulting feature dimension is

$$F = 126 + 116 = 242 \quad \text{Eq. (3)}$$

The full global state matrix is then obtained by stacking all VM-level vectors:

$$\mathbf{S_t} = [s_1^t, s_2^t, \ldots, s_N^t]^T \in \mathbb{R}^{100\times 242} \quad \text{Eq. (4)}$$

This formulation ensures that the scheduler does not depend only on isolated VM observations. Instead, it receives a structured representation of the whole cloud environment, which is then projected into a compact embedding space before attention-based decision-making.

3-3-1- Spatial–Temporal Feature Extraction and Rationale for CNN-LSTM

A hybrid CNN-LSTM feature extractor was used before the attention-aware decision module to capture two complementary characteristics of cloud scheduling states: spatial resource correlation and short-term temporal workload evolution. In the proposed environment, each scheduling state consists of VM-level information such as CPU utilization, memory availability, bandwidth usage, queue length, and workload history across 100 heterogeneous virtual machines. The convolutional component is used to identify local

10.22060/EEJ.2026.25455.5936

spatial patterns across VM resource states, such as load imbalance, resource contention, and neighboring VM utilization variation. The LSTM component is then used to model short-term temporal evolution in workload behavior, including bursty task arrivals, recent queue changes, and migration-sensitive utilization trends.

Although Transformer and Informer architectures are powerful for long-sequence modeling, the present scheduling problem is not formulated as a long-horizon time-series forecasting task. Instead, the scheduler must make low-latency decisions from compact interval-level CloudSim state snapshots. A full Transformer or Informer encoder would introduce a larger parameter space, higher tuning complexity, and additional training requirements, which may not be necessary for the relatively structured 100-VM scheduling state used in this study. Transformer-based models also rely entirely on self-attention and remove recurrence and convolution, whereas Informer was designed mainly to reduce the cost of Transformer-based long-sequence time-series forecasting through ProbSparse attention and sequence distillation.

Therefore, CNN-LSTM was retained as a lightweight spatial–temporal encoder, while the global dependency modeling role was assigned to the proposed multi-head self-attention block. This design avoids using a full Transformer as the complete feature extractor, but still incorporates attention where it is most relevant: modeling VM-to-VM global scheduling interactions after compact feature projection. In this way, the framework combines the stability and efficiency of CNN-LSTM for local spatio-temporal representation with the global contextual reasoning capability of self-attention for scheduling policy generation.

A convolutional operation first extracts local spatial associations among VM states:

$$z_t = \sigma\left(\mathbf{W}_c * s_t + \boldsymbol{b}_c\right) \quad \text{Eq. (5)}$$

10.22060/EEJ.2026.25455.5936

where * denotes convolution, $\sigma(\cdot)$ is a nonlinear activation function, and the convolutional parameters are learned during training. Next, the extracted features are passed through an LSTM network to model short-term workload dynamics:

$$\boldsymbol{h}_t = LSTM\ (\boldsymbol{z}_t, \boldsymbol{h}_{t-1}) \quad \text{Eq. (6)}$$

The resulting encoded representation is then forwarded to the attention-aware decision module. This two-stage design allows the scheduler to preserve local workload trends while still enabling global VM-to-VM reasoning through the self-attention mechanism used in the A2-DRL policy network.

Thus, CNN-LSTM is not used as a replacement for attention; rather, it acts as a compact local spatial–temporal encoder, while multi-head self-attention performs the global VM-context modeling required for scalable scheduling.

3-3-2- Action Space Definition

According to the proposed A2-DRL framework, the action space is a description of the manner in which incoming work or virtual machine (VMs) are mapped to the physical host present in the cloud environment. Every action is a **VM–host** assignment choice, which has a direct impact on system performance with regards to execution time, energy usage, and resource usage.

Let $\mathbf{H} = h_1, h_2, h_3 \ldots \ldots h_{100}$ denote the set of physical hosts and $\mathbf{V} = v_1, v_2, v_3 \ldots \ldots v_{100}$ denote the set of virtual machines. At each decision step *t*, the A2-DRL agent selects an action a_t that assigns VMs to hosts:

$$a_t: \mathbf{V} \rightarrow \mathbf{H} \quad \text{Eq. (7)}$$

In order to facilitate parallel decision-making, the policy network produces a probability distribution of all possible VM–host pairs, which is in the form of a 2-dimensional matrix:

$$\mathbf{P_t} \in \mathbb{R}^{100\times100} \quad \text{Eq. (8)}$$

10.22060/EEJ.2026.25455.5936

where each element p_{ij}^{t} indicates the probability of assigning VM $v_{i,}$ to host $h_{j,}$ at time step *t*.

3-3-3- SoftMax-Based Policy Selection

The probabilities of action are calculated with the help of a softmax policy:

$$p_{ij}^{t} = \frac{\exp(z_{ij}^{t})}{\sum_{k=1}^{100} \exp(z_{ij}^{t})} \quad \text{Eq. (9)}$$

where z_{ij}^{t} is the raw output (logit) of the actor network for the VM–host pair $(v_{i,}, h_{j,})$. This formulation is such that the sum of assignment probabilities of a VM on all the hosts is one. The last scheduling step is acquired by choosing a host that has the best probability of each VM:

$$a_{t}^{(i)} = \arg \max_{j} p_{ij}^{t} \quad \text{Eq. (10)}$$

This action representation as a probabilistic action, can also permit the scheduler to achieve a balance between exploration and exploitation and the agent can learn the best VM placement strategies in the face of dynamic and uncertain workload conditions.

The proposed framework makes the action space a 100 × 100 probability matrix to allow parallel and scalable decision making in the entire cloud infrastructure. Using this representation in tandem with the attention-aware state representation, it is possible to schedule actions worldwide instead of making independent and sequential choices.

3-4- Reward Function Design

The proposed A2-DRL scheduler has the reward functional at the center in shaping the learning behavior. Cloud task scheduling has many, and sometimes conflicting goals, so the reward is formulated as a multi-

10.22060/EEJ.2026.25455.5936

objective, weighted cost function that includes execution efficiency, energy awareness, and economic cost, and rewards Service Level Agreement (SLA) violations.

At each decision step t, the reward is defined as:

$$R_t = -(\alpha E_t + \beta T_t + \gamma C_t) - \lambda(\Phi_t) \quad \text{Eq. (11)}$$

where: E_t denotes the total energy consumption of physical hosts and VMs, T_t represents the task execution time (makespan), C_t corresponds to the resource usage cost, α, β, γ are non-negative weighting coefficients controlling the relative importance of each objective, Φ_t is the SLA violation indicator, and λ is the SLA penalty coefficient.

3-4-1- SLA Violation Modeling

Violations of SLA are introduced as a specific penalty not to encourage the choice to schedule that jeopardizes the quality of service. The term violation is defined to be:

$$\Phi_t = \begin{cases} 1, & if\ SLA\ is\ violated \\ 0, & Otherwise \end{cases} \quad \text{Eq. (12)}$$

The formulation makes sure that a schedule resulting in delays that exceed the deadline, or consume more resources than necessary, or have resources that are contended with will earn a lower reward regardless of whether the schedule has good energy or cost results.

3-4-2- Rationale of the Reward Design

The reward formulation converts the scheduling problem into a multi-objective cost-minimization task suitable for actor–critic learning. By adjusting the weighting parameters *α, β,* and *γ*, the scheduler can be tuned to prioritize energy efficiency, execution performance, or operational cost sensitivity depending on

10.22060/EEJ.2026.25455.5936

system requirements. The SLA penalty term further prevents the learning policy from selecting energy-saving actions that compromise service reliability.

3-5- Baseline A3C-R2N2 Architecture

A3C-R2N2 baseline architecture in Fig. 2 is based on the advanced Asynchronous Advantage Actor Critic (A3C) framework of scheduling suggested in [25]. The model combines residual convolutional and recurrent neurals to improve the learning of temporal features and still asynchronously optimize policy.

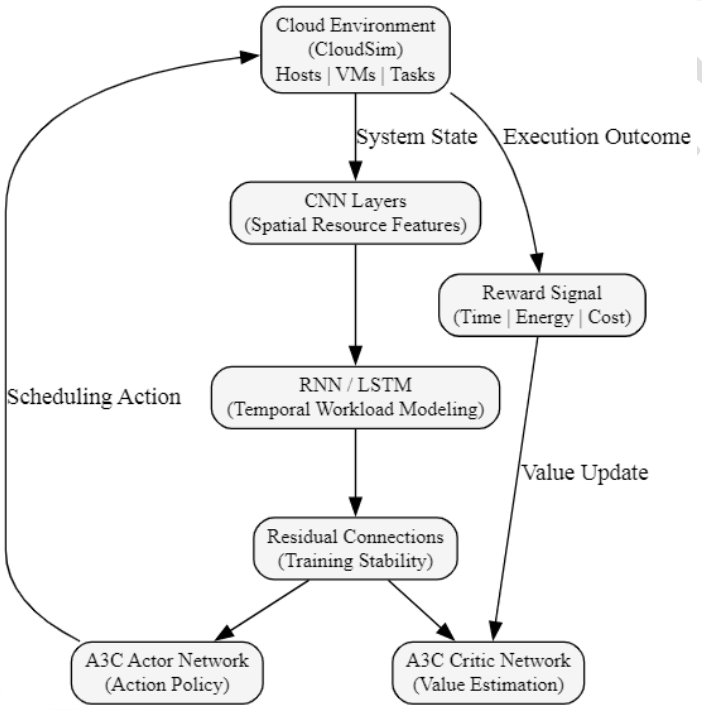

Fig. 2. Baseline A3C-R2N2 Scheduling Architecture [25]

The global cloud state, gathered by the CloudSim environment, in this framework is initially fed into convolutional layers to isolate patterns of virtual machine spatial resource utilization. The characteristics are then transferred to a recurrent neural network (RNN/LSTM) to simulate temporal evolution of workload. Connection with residual between layers of feature extraction are added to enhance training stability and gradient flow.

10.22060/EEJ.2026.25455.5936

This recurrent output is inputted into the A3C actor-critic networks, and the actor produces the scheduling output, and the critic approximates the state value function. Because of the repetitive form, the scheduling choices are generated in a serial manner, which depends on past concealed states, without precise modeling of worldwide VM-to-VM interactions. This architecture is maintained as it is and serves as a point of reference when comparing it to the proposed A2-DRL structure.

3-6- Proposed A2-DRL Framework

The idea behind the proposed Attention-Aware Deep Reinforcement Learning (A2-DRL) framework in Fig. 3 aims to address the shortcomings of recurring A3C-based schedulers with the capability of making global decisions, learning in a stable manner, and making decisions scalable in dynamic clouds. In contrast to the baseline A3C-R2N2 architecture (which uses sequential recurrent representations), A2-DRL proposes an attention-based decision pipeline, so it explicitly describes interactions between all virtual machines (VMs) at each decision step.

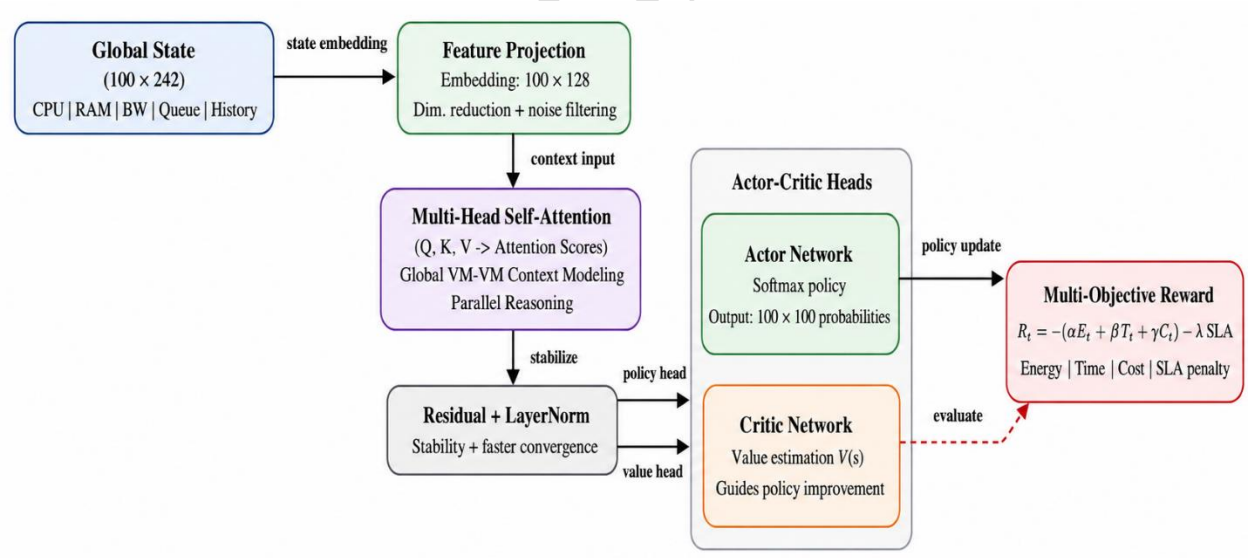

Fig. 3. Internal Architecture of the Proposed A2-DRL Framework

3-6-1- Feature Projection Layer

10.22060/EEJ.2026.25455.5936

The raw VM-level state representation has a feature size of 242, which is informative but computationally expensive for direct attention processing. Therefore, a feature projection layer is introduced to map each VM-level state vector into a compact latent embedding:

$$z_i^t = \phi(\mathbf{W}_p s_i^t + \mathbf{b}_p) \quad \text{Eq. (13)}$$

where $s_i^t \in \mathbb{R}^{242}$, $\mathbf{W}_p$ and $\mathbf{b}_p$ are learnable projection parameters, and $\phi(\cdot)$ denotes the ReLU activation function. In the implemented A2-DRL model, the 242-dimensional input is projected into a 128-dimensional embedding. This projection reduces computational overhead while preserving scheduling-relevant information for the attention block. The projected global state is represented as:

$$Z_t \in \mathbb{R}^{100\times128} \quad \text{Eq. (14)}$$

where 100 represents the number of VMs and 128 is the projected embedding dimension. This compact representation is used as the input to the multi-head self-attention module.

3-6-2- Multi-Head Self-Attention Mechanism

The central innovation of the proposed A2-DRL framework is the use of multi-head self-attention to model global VM-to-VM dependencies. Unlike recurrent schedulers, which process resource states sequentially, self-attention evaluates interactions among all VMs within the same decision step. This allows the scheduler to identify overloaded VMs, underutilized hosts, and migration-sensitive workload patterns using global context.

In the implemented model, the projected VM state $Z_t \in \mathbb{R}^{100\times128}$is processed using **four attention heads**. For each head, query, key, and value matrices are computed as:

$$\mathbf{Q} = \mathbf{Z_t W_Q}, \mathbf{K} = \mathbf{Z_t W_K}, \mathbf{V} = \mathbf{Z_t W_V} \quad \text{Eq. (15)}$$

The attention output is calculated as:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}}\right)\mathbf{V} \quad \text{Eq. (16)}$$

The outputs from the four heads are concatenated and passed through a residual connection followed by layer normalization. This design enables the model to learn multiple interaction perspectives, such as resource contention, workload imbalance, migration opportunity, and host utilization pressure. The attention-enhanced representation is then forwarded to the scheduling output layer to generate VM–host assignment probabilities.

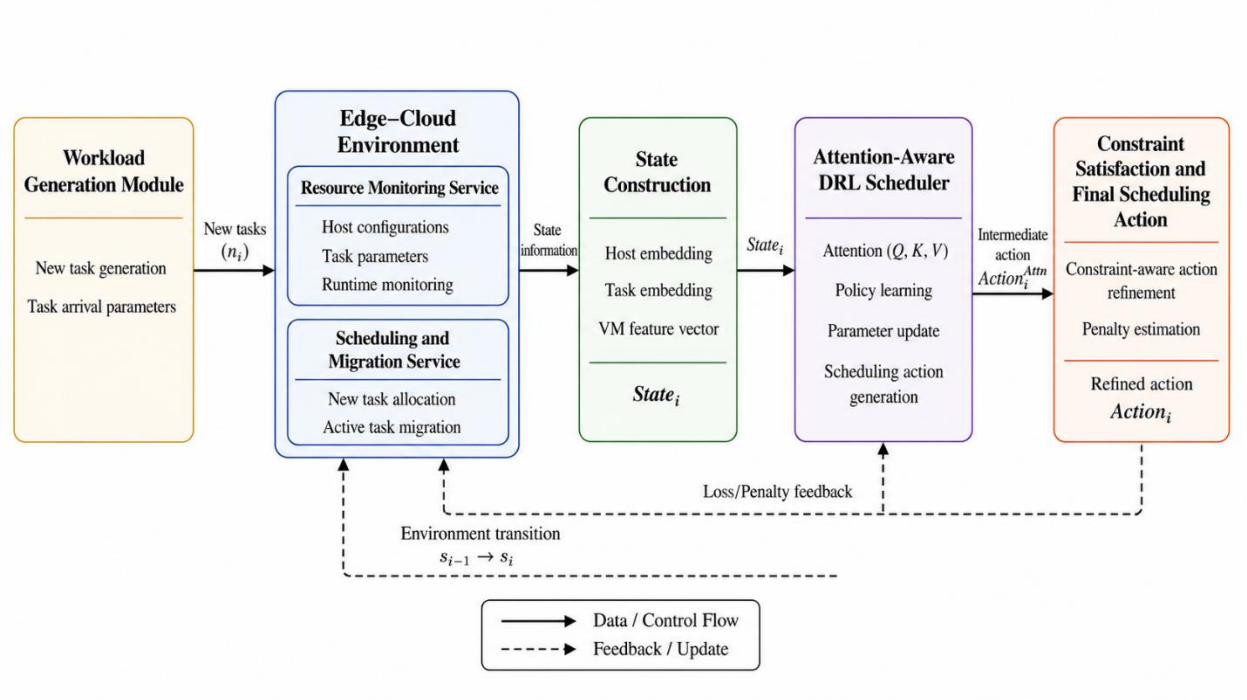

Fig. 4. System-level workflow of the proposed attention-aware deep reinforcement learning (A2-DRL) framework for task scheduling and migration in an edge–cloud environment.

Fig. 4 shows the system-level workflow from workload generation to edge-cloud monitoring, state construction, attention-aware scheduling, and final constraint-based action refinement.

3-6-3- Residual Connections and Normalization

10.22060/EEJ.2026.25455.5936

In a way that guarantees the stability of the training and rapid convergence, residual (skip) connections are used around the attention block:

$$h_t = LayerNorm(e_t + Attention\ (e_t)) \quad \text{Eq. (17)}$$

The residual learning allows computing the vanishing gradients to be eliminated and the performance degradation is reduced with the network depth. The further stabilisation of learning by layer normalisation is the reduction of internal covariate shift. This design directly helps solve the instability of convergence that is seen in recurrent A3C-based schedulers.

3-6-4- Policy and Value Networks

The refined contextual representation is common to two parallel networks:

- Actor Network: returns a probabilistic schedule policy on VM host assignments.
- Critic Network: approximates the value to be had with the state.

The scheduling probabilities are generated by the actor with the help of a softmax policy, whereas the critic assesses the long-term performance with the multi-objective reward function previously described. Both networks are co-trained based on actor-critic optimization, and simultaneous learning of performance-effective and energy-conscious scheduling policies is made possible. Table 2 gives an overview of the major architectural elements of the proposed A2-DRL framework, including the design decision they make and the rationale behind this design decision in the context of complexity reduction, global contextual reasoning, learning stability, and efficient policy optimization.

Table 2. Design Rationale of the Proposed A2-DRL Framework

Component	Design Choice	Motivation

Feature Projection	242 → 128 embedding	Reduce complexity while preserving decision-relevant features
Self-Attention	Multi-head attention	Enable global VM–VM interaction modeling
Residual Learning	Skip connections + normalization	Improve convergence stability and gradient flow
Actor–Critic	Shared representation	Joint optimization of scheduling policy and value estimation
Parallel Processing	Attention-based reasoning	Replace sequential recurrent decision-making

3-7- Training Strategy, Computational Optimization, and Experimental Configuration

The proposed A2-DRL scheduler was implemented in Python 3.12.9 using PyTorch 2.9.1+cpu and integrated with the CloudSim simulation environment through a lightweight TCP/RPC communication bridge. CloudSim was used to simulate task execution, virtual machine behavior, host utilization, migration activity, energy consumption, and SLA-related performance logs. The Python-based A2-DRL agent receives the global scheduling state from CloudSim, processes the state through the attention-aware actor–critic network, and returns VM–host assignment decisions to the simulation environment.

The training process follows an actor–critic learning design with attention-based contextual representation. At each scheduling interval, the VM-level state matrix is first passed through a feature projection layer, where the 242-dimensional input representation is compressed into a 128-dimensional embedding.

The projected state is further processed through a four-head self-attention block to capture global VM-to-VM dependencies across the scheduling environment. Residual connections and layer normalization techniques are employed in order to stabilize the training process and minimize the drop in performance that may arise due to updates in policy. The actor predicts the probability distribution for VM–host

10.22060/EEJ.2026.25455.5936

assignments, while the critic evaluates the state value according to the multi-objective reward function described above. The reward function is defined based on makespan, power consumption, resource cost, and SLA violations. In the present study, however, cost is considered part of the reward structure, corresponding to the operational expense of resources, and it is not used in economic calculations.

In order to promote reproducibility, the most important implementation, simulation, learning, and evaluation parameters are listed in Table 3.

Table 3. Experimental and Training Configuration of the Proposed A2-DRL Framework

Category	Parameter	Value / Description
Implementation environment	Programming language	Python 3.12.9
Implementation environment	Deep learning library	PyTorch 2.9.1+cpu
Hardware execution mode	GPU usage	CPU-based execution in the verified terminal environment
Simulation environment	Cloud simulator	CloudSim
Communication mode	Python–CloudSim bridge	Lightweight TCP/RPC communication
Cloud infrastructure	Physical hosts	100
Cloud infrastructure	Virtual machines	100

10.22060/EEJ.2026.25455.5936

Category	Parameter	Value / Description
State representation	CNN-side spatial/resource features	126
State representation	LSTM-side temporal/workload features	116
State representation	Total VM-level feature dimension	242
State representation	Global state matrix	100 × 242
Feature projection	Input-to-embedding projection	242 → 128
Attention module	Attention type	Multi-head self-attention
Attention module	Number of attention heads	4
Attention module	Residual connection	Used
Attention module	Layer normalization	Used
Action representation	Per-VM host output dimension	100
Action representation	Final scheduling action matrix	100 × 100 VM–host probability matrix
Policy selection	Assignment decision	SoftMax-based VM–host probability ranking
Optimizer	Optimization algorithm	Adam

Category	Parameter	Value / Description
Learning setup	Learning rate	0.00001
Active A2-DRL setting	Batch size	1
Baseline CNN-LSTM setting	Batch size	5
CUDA baseline script setting	Batch size	12
Reward objective	Reward components	Makespan, energy consumption, operational resource cost, and SLA penalty
Cost handling	Cost interpretation	Operational reward component; not standalone billing analysis
Response-time metric	Calculation	Total logged response time / completed task/VM events
Statistical reporting	Dispersion level	Interval-level mean, standard deviation, and 95% confidence interval

This configuration improves reproducibility by specifying the software environment, state-space dimension, feature projection size, number of attention heads, action representation, optimization scheme, and statistical reporting procedure. The use of the state matrix of size 100 × 242 allows all resource and workload-related features on a per-VM basis to be observed within one decision-making cycle, while the action matrix of size 100 × 100 is used to make scheduling decisions for multiple VMs simultaneously. The proposed four-head self-attention block allows learning various patterns of interactions between VMs, such as resource conflicts, underutilization, migration fitness, and workload imbalance.

10.22060/EEJ.2026.25455.5936

3-8- Performance Metrics

For the purpose of assessing the success of the proposed A2-DRL scheduling scheme, some metrics need to be considered, including efficiency, cost, reliability, and learning behavior. The formulae for each performance metric are illustrated in Table 4 below.

- **Makespan** measures the total completion time of all submitted tasks and reflects overall scheduling efficiency.
- **Energy consumption** quantifies the total power usage of physical hosts and virtual machines during task execution, capturing energy efficiency.
- **SLA violation rate** indicates the proportion of tasks that fail to meet predefined service-level constraints, reflecting quality-of-service compliance.
- **Load variance across VMs** measures workload distribution balance and indicates the effectiveness of load balancing decisions.
- **Convergence speed** evaluates how quickly the learning algorithm stabilizes, reflecting training efficiency and scalability.

Collectively, these measures give an overall evaluation of the operational performance as well as learning stability.

Table 4. Performance Metrics and Mathematical Formulation

Metric	Symbol	Mathematical Formulation	Description
Makespan	MS	$MS = max(Ci), i = 1, 2, \ldots, n$	Maximum completion time among all tasks

Energy Consumption	E	$E = \sum_{h=1}^{H} (P_h \times T_h)$	Total energy consumed by all hosts
Service-Level Agreement Violation Rate	SLA_v	$SLA_v = \frac{N_{viol}}{N_{Total}}$	Ratio of tasks violating service-level agreement constraints
Load Variance Across VMs	σ_{load}^2	$\sigma_{load}^2 = \frac{1}{N}\sum_{j=1}^{N}(L_j - \bar{L})^2$	Measures workload imbalance among VMs
Convergence Speed	T_{conv}	$T_{conv} = min\{t \mid \|\Delta R_t\| < \epsilon\}$	Time/episodes required for reward stabilization

The discussed A2-DRL framework is compared to a series of conventional baseline approaches, such as DLSF, DDQN, REINFORCE, and A3C-R2N2 architecture. These baselines include heuristic, value-based, policy-gradient and actor-critic reinforcement learning paradigms, which makes the comparison of all categories fair and complete. To ensure consistency of the experiments, all the methods are tested with the same workload traces, cloud configurations, task arrivals distributions and simulation durations. The variation in performance can only be explained by the architectural and algorithmic design choice but not the environmental disparity. The suggested methodology combines the attention-based global reasoning process, stable actor-critic learning, and multi-objective optimization to overcome the constraints of sequential and locally aware schedulers. All design decisions, such as feature projection, self-attention, residual learning and multi-objective reward formulation are clearly aimed at minimizing execution time, energy consumption, operational resource cost and SLA violations and maximizing load balance and convergence behavior

4. Results and Discussion

This section presents the experimental evaluation of the proposed A2-DRL scheduling framework in a CloudSim-based heterogeneous cloud environment. The evaluation examines the effectiveness, efficiency,

and stability of the proposed scheduler under dynamic workload conditions. The proposed A2-DRL scheduler was compared with representative baseline approaches, including DLSF, REINFORCE, DDQN, and A3C-R2N2. This way, both heuristic, policy gradient, value-based, and actor-critic reinforcement learning methodologies could be considered for a comprehensive comparison between various scheduling techniques. The experiments involved the same cloud environment, workloads, task arrival distribution, and simulation period The evaluation metrics included makespan, energy consumption, SLA violation behavior, load variance, convergence behavior, response time, and migration activity. Furthermore, the recorded simulation output data were subjected to descriptive statistical analysis at the interval level through the computation of mean, standard deviation, and 95% confidence intervals. For simulations where the independent seed folders were not available, the dispersion presented is taken as an interval-level variance rather than 10-20 independent seed validation.

4-1- Statistical Dispersion and Reliability Analysis

Descriptive statistical measures were also determined for each available interval-based measure, which includes mean, standard deviation, and 95% confidence interval. This will give us better insight into the variation of energy consumption, behavior of SLA violation, operation cost, response time, completion behavior, and migration behavior.

For the primary logged run, 287 simulation intervals were available. The total logged energy value was 91,615,299.3574, with mean interval energy of 319,217.0709 ± 97,607.4147 and a 95% confidence interval half-width of 11,292.7037. The cumulative SLA violation index was 0.04509377, while the mean interval SLA violation was 0.00015712 ± 0.00005688, with a 95% confidence interval half-width of 0.00000658. The run recorded 827 completed task/VM events, total logged response time of 8.2700 s, and an observed logged response time of 10.0000 ms per completed task/VM. In addition, 4,071 VM migrations were recorded, with a total migration time of 526.0800 s and mean interval migration count of 14.1847 ± 8.1010.

10.22060/EEJ.2026.25455.5936

Table 5. Statistical dispersion of logged A2-DRL scheduling results

Metric	Result
Number of logged intervals	287
Total logged energy	91,615,299.3574
Mean interval energy	319,217.0709 ± 97,607.4147
95% CI half-width for interval energy	11,292.7037
Total operational cost	6,331.4194
Mean interval cost	22.0607 ± 0.1069
Total SLA violation index	0.04509377
Mean interval SLA violation	0.00015712 ± 0.00005688
95% CI half-width for interval SLA	0.00000658
Completed task/VM events	827
Total logged response time	8.2700 s
Logged response per completed task/VM	10.0000 ms
Total VM migrations	4,071
Total migration time	526.0800 s
Mean interval migrations	14.1847 ± 8.1010

10.22060/EEJ.2026.25455.5936

Table 5 summarizes the dispersion statistics obtained from the available logged simulation intervals. The results show that although interval energy consumption fluctuates due to dynamic workload variation, SLA violation remains consistently low. The logged response time of 10.0000 ms per completed task/VM indicates low scheduling-response overhead in the available simulation records. The migration statistics also show that the scheduler performs controlled VM relocation rather than excessive migration.

The reported values represent dispersion from available logged simulation intervals and should not be interpreted as independent multi-seed statistics unless additional independent seed-level runs are executed and included.

4-2- Decision Response Time and Real-Time Feasibility

The decision-response behavior of the proposed A2-DRL scheduler was evaluated using the logged response-time records generated during the CloudSim-based simulation. The response-time metric was computed as the ratio between the total logged response time and the number of completed task/VM events. The simulation recorded a total response time of 8.2700 s for 827 completed task/VM events. Therefore, the average logged response time per completed task/VM is calculated as:

$$T_{response} = \frac{T_{total-response}}{N_{completed}} \times 1000 \quad \text{Eq. (18)}$$

$$T_{response} = \frac{8.2700}{827} \times 1000 = 10.0000 \text{ ms} \quad \text{Eq. (19)}$$

This finding shows that the proposed A2-DRL algorithm has a low response overhead even during interval-based decision-making. The measured response time of 10.0000 ms per executed task/VM, along with the very low index of SLA violations and well-managed VM migration strategy, suggests that the real-time implementation is feasible for the proposed model in the tested dynamic workload scenario. Due to the nature of scheduling being done at certain simulation intervals, and not on a per-microsecond control loop

10.22060/EEJ.2026.25455.5936

basis, the response time measured is appropriate for the studied cloud scheduling environment with CloudSim tool support.

4-3- Energy–SLA Stability Analysis

Results from the experiment regarding overall performance indicate that the proposed A2-DRL framework manages to attain a normalized makespan of about 0.85. This demonstrates successful task execution on account of the evaluated workloads. Fig. 5 gives the analysis of the combined trend for energy consumption and violations of service level agreement over a 1400 minutes time duration. There is a variance in the trend for energy consumption, ranging between approximately 2.0×10^5 and 4.5×10^5 power units recorded. In spite of this, the SLA violation rate is still kept lower than 2.6×10^{-4}.

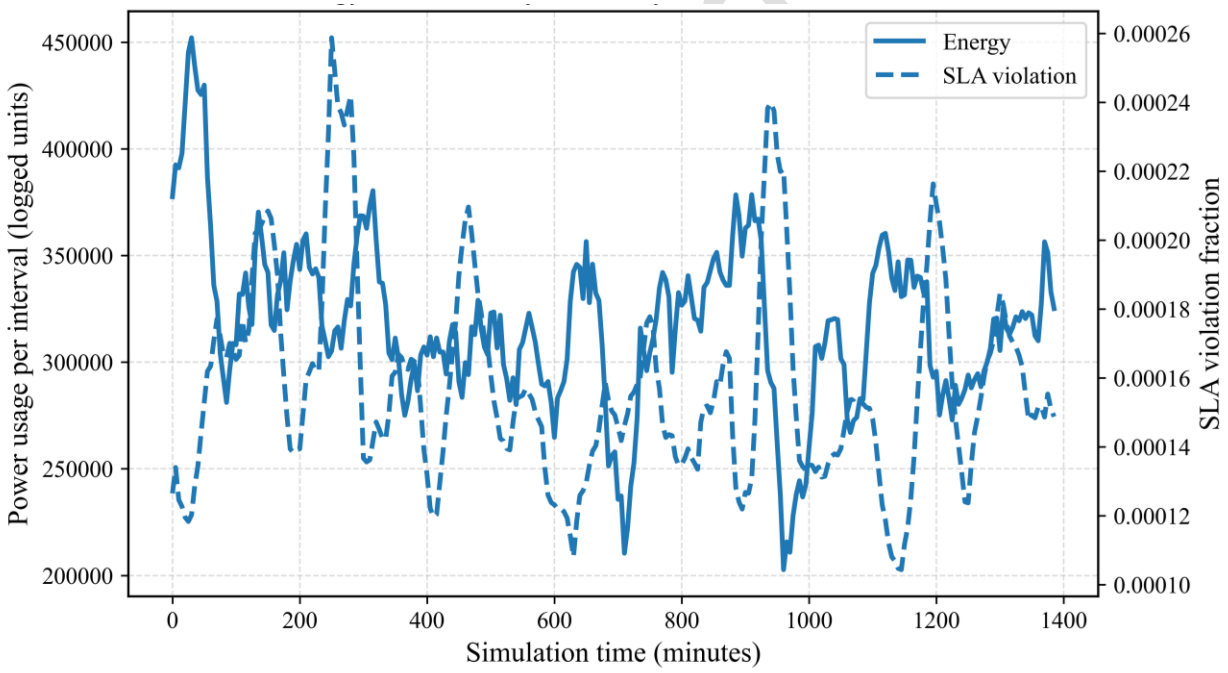

Fig. 5. Energy–SLA stability analysis of the proposed A2-DRL scheduler, illustrating energy consumption fluctuations and consistently low SLA violation rates over the simulation duration.

10.22060/EEJ.2026.25455.5936

The results show that the A2-DRL algorithm retains the quality of services despite varying levels of energy during the workload of the system. The decision-making process through attention allows for preemptive migration of tasks to avoid violating the SLA during high workload periods. This demonstrates the validity of the multi-objective reward function since the SLA penalty ensures that the scheduler avoids excessive energy-saving decisions that could compromise service reliability.

4-4- Task Completion Efficiency

Fig. 6 illustrates the task completion process for the proposed A2-DRL scheduler. The number of tasks completed was about 800 tasks, whereas the percentage of exceeding expected time stayed very small. It means that the proposed scheduling policy has the capability to guarantee reliable task completion without affecting performance.

Fig. 6. Task completion performance of the proposed A2-DRL scheduler, showing a high number of completed tasks with a minimal exceeded expected time percentage.

10.22060/EEJ.2026.25455.5936

This is an indication of the reliability of the proposed model during its implementation. The use of global context-based task allocation along with global VM level contextual data makes sure that the waiting queue pressures are reduced without violating deadlines significantly.

4-5- Robustness under Hyperparameter Variations

Fig. 7 studies the scheduling mechanism robustness via studying the number of migrations in the tasks for varying hyperparameter setups (α, β, γ, δ, ϵ). While the number of migrations changes, their behaviour remains bounded and constant.

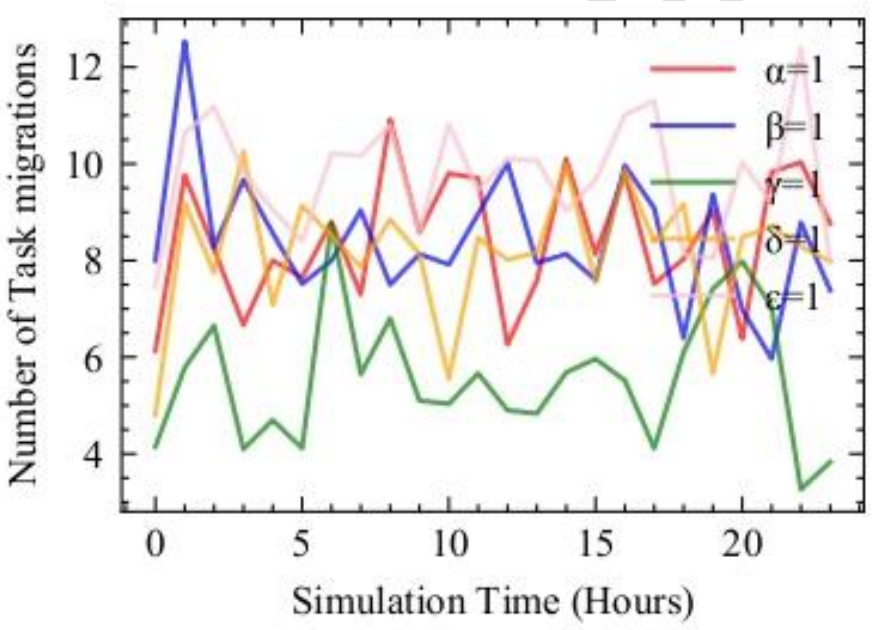

Fig. 7. Task migration behavior under different hyperparameter weight configurations, demonstrating stable and bounded migration patterns across varying reward settings.

This result indicates that the feature extraction and state representation components of A2-DRL remain robust to changes in reward-weight settings. Scheduler adjusts its behavior without showing any erratic or unstable migration behavior which is vital when applying the system in real life where parameters may change according to workloads.

10.22060/EEJ.2026.25455.5936

4-6- Migration Time and Scheduling Overhead

Fig. 8 presents the interval-level migration time under different hyperparameter settings. Migration mostly occurs within a small-time frame without any sudden or unstable spikes being noted. This implies that the proposed A2-DRL scheduler is able to keep migration times at a stable level.

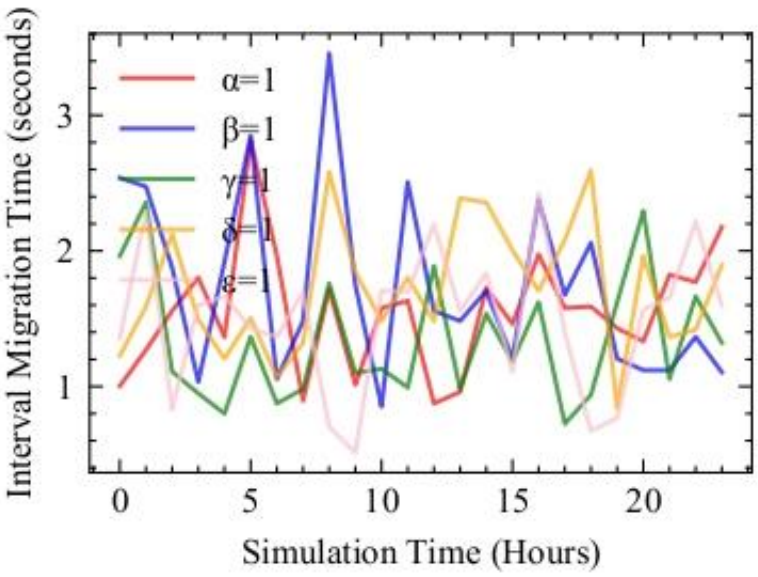

Fig. 8. Migration time during intervals with different settings of hyperparameters, and this demonstrates consistent and effective relocation of tasks throughout the time period of simulation.

It is also shown that the efficiency in terms of reduced scheduling overhead is primarily due to the ability to make parallel decisions by means of the self-attention algorithm. While the conventional approach makes use of sequential processing, the new algorithm allows for parallel decisions to be made.

4-7- Migration Dynamics of the Proposed Model

Fig. 9 shows the migration-count variation of the proposed A2-DRL scheduler during the simulation period. The migrations performed tend to be relatively high at the start of the simulation due to exploration of the optimal placement of the virtual machines on different hosts amid varying workload scenarios.

10.22060/EEJ.2026.25455.5936

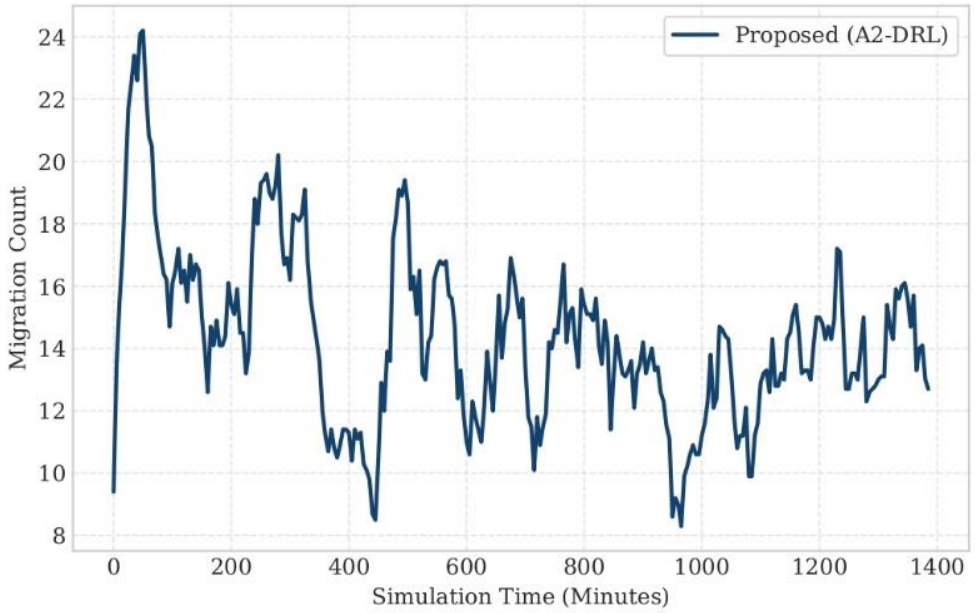

Fig. 9. Migration count variation of the proposed A2-DRL scheduler over the simulation period

This indicates that there is a consistent pattern of policy convergence wherein inappropriate migrations decrease once the scheduler achieves a balanced state. This is necessary since too much VM migration may cause increased communication costs, energy consumption, and instability of services. Thus, the detected migration trend proves the assertion that A2-DRL achieves scheduling stability amid varying workloads.

4-8- Interval Energy Consumption Analysis

Fig. 10(a) presents the interval-wise energy consumption trend of the proposed A2-DRL scheduler, while Fig. 10(b) shows the corresponding SLA violation stability under dynamic workload conditions. Despite the fluctuations of energy consumption over time, there is an observation that the trend of SLA violations stays low through simulation iterations.

AUT Journal of Electrical Engineering
10.22060/EEJ.2026.25455.5936

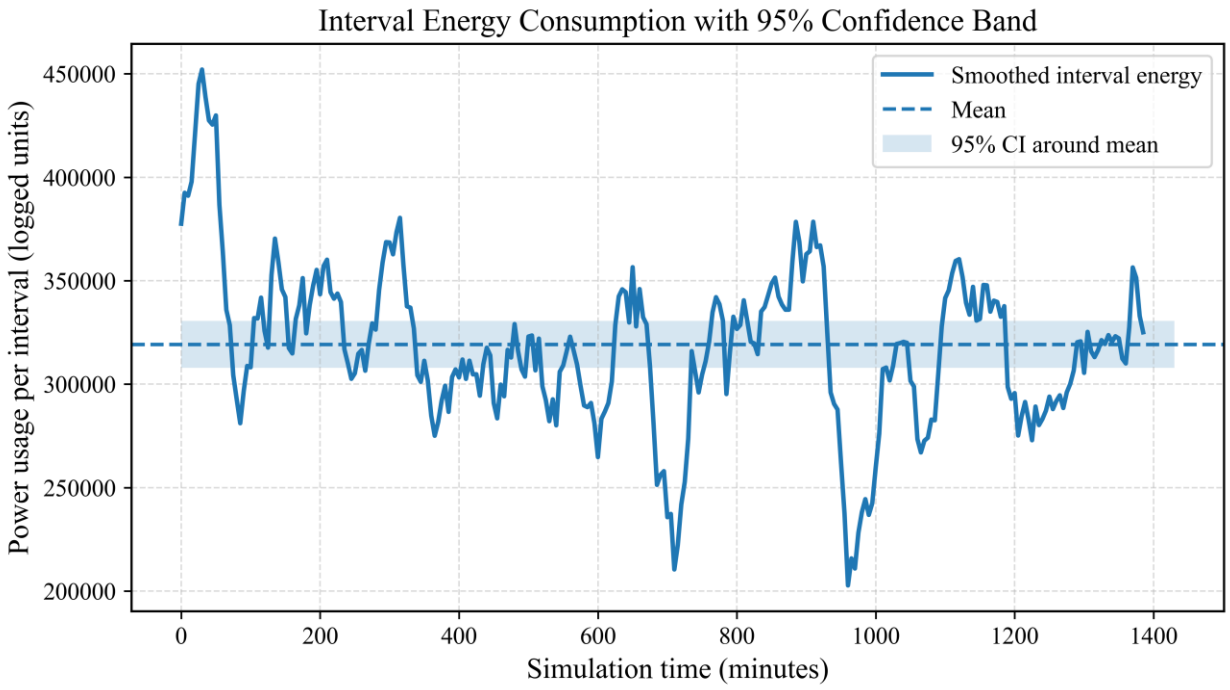

Fig. 10(a). Interval energy consumption trend of the proposed A2-DRL scheduler with 95% confidence band.

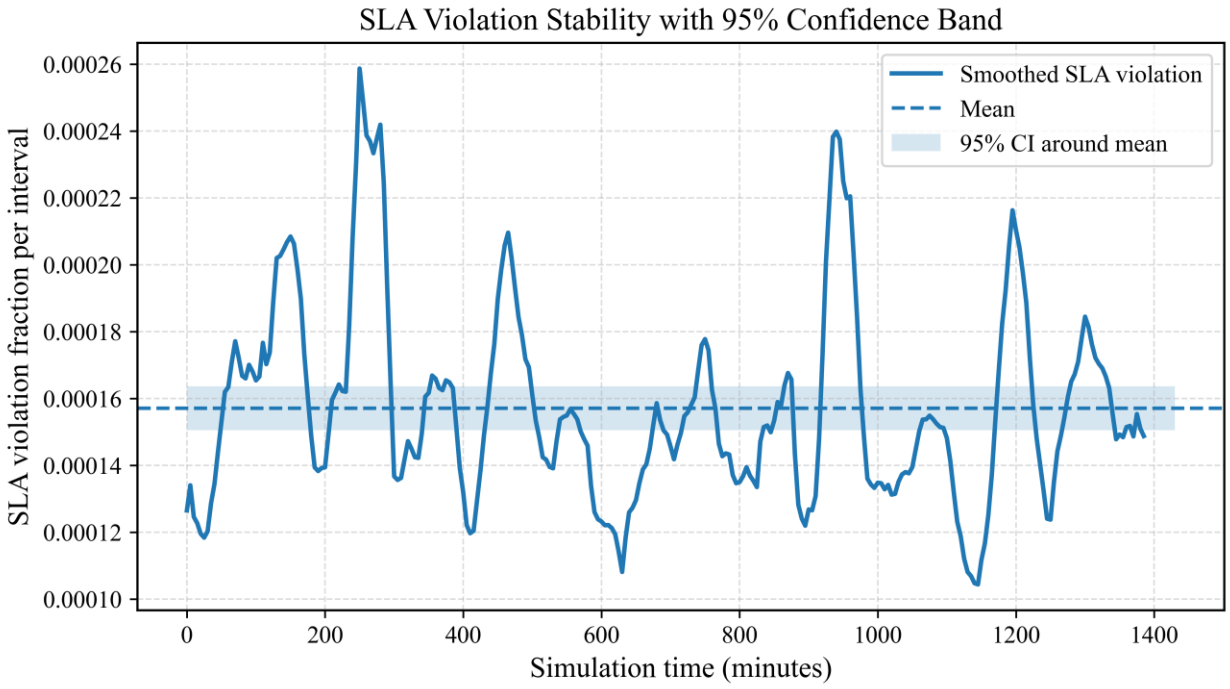

Fig. 10(b). SLA violation stability of the proposed A2-DRL scheduler with 95% confidence band under dynamic workload conditions.

Despite these fluctuations, the proposed scheduler achieves a 14.4% decrease in overall energy expenditure compared to the baseline policy [25]. It is primarily due to workload consolidation and migration guided by attention, which minimize redundant host activations without compromising application performance.

Overall, the experiments reveal that the presented approach provides benefits for energy savings, SLA guarantee, and stable scheduling performance in highly dynamic cloud and edge-cloud environments. The ability to consider global context between all VMs via the multi-head attention makes it possible to ensure an appropriate workload allocation and migration. Moreover, residual learning along with normalization helps in ensuring stable convergence. As shown in Table 6, recent scheduling studies have increasingly adopted attention mechanisms, Decision Transformer-based offline reinforcement learning, and Transformer-enhanced distributed DRL to improve adaptability and convergence in cloud–edge scheduling environments. However, these methods mainly focus on cloud manufacturing, logistics-involved scheduling, or IoT edge–cloud application scheduling. In contrast, the proposed A2-DRL framework focuses on global VM-to-VM contextual reasoning, SLA-aware energy-efficient scheduling, and actor–critic decision learning in a CloudSim-based cloud task scheduling environment.

Table 6. Comprehensive Comparative Analysis of Cloud Task Scheduling Frameworks

Author & Year	Scheduling Framework	Core Techniques	Key Objectives	Time & Energy Performance	Load Balancing Strategy	Key Limitations
[20]	Heuristic Algorithms for Task Scheduling	Comparative study of heuristic algorithms, e.g., Min-Min, Max-Min	Makespan, resource utilization, cost, degree of imbalance, throughput	Heuristics provide faster solutions but are often suboptimal compared to advanced methods	Various heuristic-driven assignments	Limited optimality; fixed rules; less adaptable to workload changes

Author & Year	Scheduling Framework	Core Techniques	Key Objectives	Time & Energy Performance	Load Balancing Strategy	Key Limitations
[21]	Advancements in heuristic task scheduling for IoT applications	Review of heuristic advancements for fog-cloud computing	Latency, energy, resource utilization	Heuristics show promise but face challenges with scalability and real-time demands	Various heuristic strategies	Difficulty optimizing multiple objectives simultaneously; not learning-based
[22]	Heuristic Scheduling Approach for Fog-Cloud	Heuristic algorithms for stationary IoT devices	Makespan, latency, energy consumption	Optimizes these metrics but often struggles with dynamic scenarios and complex interdependencies	Resource allocation for stationary devices	Limited adaptability to non-stationary or highly dynamic IoT environments
[23]	Dynamic energy-aware scheduling	Bi-objective cost function for makespan and energy efficiency	Energy consumption, execution time	Provides trade-off between energy consumption and execution time	Not explicitly detailed	Trade-off approach may not optimize all objectives simultaneously
[24]	MILP formulations for spatio-temporal thermal-aware scheduling	Mixed Integer Linear Programming	Energy, makespan	Optimized energy value is 39% lower; makespan optimization yields 30% lower energy; mean energy difference 6.7%	Spatio-temporal allocation	Computationally expensive for large-scale problems
[29]	Application Scheduling on Edge, Fog, and Cloud	Resource-aware algorithms and evaluated scheduling strategies	Cost and performance, including latency and throughput	Performance varies by strategy; trade-offs exist between cost and performance	Implied resource mapping	Does not leverage advanced learning; may lack adaptability to dynamic workloads
[31]	Optimization model for QoS-based	Combined parameter-based	Load balancing, response time,	Approx. 3 s for 40 independent tasks and 2.25 s for 7 dependent tasks	Max-Min outperforms Min-Min for larger tasks	Focuses on specific task types; heuristic-based

Author & Year	Scheduling Framework	Core Techniques	Key Objectives	Time & Energy Performance	Load Balancing Strategy	Key Limitations
	task scheduling	scheduling strategy	resource utilization, memory storage			
[32]	Evaluation of Task Scheduling Algorithms	Independent task scheduling algorithms	Task failure reduction and resource utilization	More than 25% of tasks fail for large task sets in heterogeneous edge-cloud environments	Not specified	Significant degradation in heterogeneous edge-cloud environments
[33]	Enhanced Osprey Optimization Algorithm	Modified OOA	Makespan, energy consumption, SLA violations	Reduces makespan by 27%, energy by 36%, and SLA violations by 50%	Implicitly handled by optimization	Metaheuristic; may face convergence challenges
[34]	Cloud–edge collaborative task scheduling	Attention-based DRL with Gated Transformer-XL and V-MPO	Customer satisfaction, production balance, and cloud–edge task scheduling	Validated effectiveness, training stability, generalizability, scalability, and robustness in cloud–edge manufacturing scheduling	Attention-guided scheduling using cloud service and edge production information	Focused on cloud manufacturing; limited emphasis on VM-to-VM data-center scheduling and SLA-aware energy optimization
[35]	Logistics-involved task scheduling in cloud manufacturing	Offline DRL with Decision Transformer and attention-based policy	Makespan, cost, offline scheduling, and historical-data utilization	Reported competitive scheduling performance against DDQN, DRQN, PPO, and behavior cloning under multiple scheduling scenarios	Sequential MDP modeling and attention-based task allocation	Designed for cloud manufacturing logistics; not focused on real-time VM-level cloud scheduling
[36]	TF-DDRL for IoT application	Transformer-enhanced distributed	Response time, energy consumption,	Reported reductions in response time,	Distributed scheduling across edge	Focused on IoT application scheduling; not

Author & Year	Scheduling Framework	Core Techniques	Key Objectives	Time & Energy Performance	Load Balancing Strategy	Key Limitations
	scheduling in edge–cloud systems	DRL with actor–critic learning, prioritized experience replay, and off-policy correction	monetary cost, and weighted cost optimization	energy consumption, monetary cost, and weighted cost	and cloud servers	primarily designed for global VM-to-VM SLA-aware cloud task scheduling
Proposed (A2-DRL)	Attention-Augmented Actor–Critic	Self-Attention + Residual Learning + Actor–Critic DRL	Energy, SLA reliability, and low response overhead	10.0 ms logged response per completed task/VM, 14.4% energy reduction, stable SLA, normalized makespan ≈0.85	Global VM-to-VM context-aware scheduling	Full economic cost analysis is not explicitly modeled; cost is treated as an operational reward component

Note: The response-time value represents the logged response time per completed task/VM event, calculated from the simulation records as total response time divided by completed task/VM events.

Moreover, the multi-objective reward formulation controls the trade-off among execution time, energy consumption, operational resource cost, and SLA violation. The consistently low SLA violation behavior indicates that the proposed scheduler avoids excessive energy-saving decisions that could compromise service reliability. Unlike heuristic schedulers and many conventional DRL-based methods that rely on sequential or locally informed decisions, A2-DRL maintains stable performance under varying workload conditions, hyperparameter settings, and fluctuating energy behavior. These findings support the novelty of attention-based parallel reasoning for cloud task scheduling and demonstrate the practical relevance of A2-DRL for large-scale cloud and edge–cloud environments where adaptability, efficiency, and reliability are essential.

5. Conclusion

This study presented A2-DRL, an attention-driven actor–critic framework for energy-efficient and SLA-aware cloud task scheduling. The proposed model advances beyond conventional recurrent and locally informed schedulers by combining CNN-LSTM-based spatial–temporal feature extraction with multi-head self-attention for global VM-to-VM contextual reasoning. The framework formulates scheduling as a multi-objective reinforcement learning problem using VM-level state representation, probabilistic VM–host assignment, and reward feedback based on makespan, energy consumption, operational resource cost, and SLA penalties. CloudSim-based evaluation demonstrated that A2-DRL achieves a normalized makespan of approximately 0.85, reduces total energy consumption by 14.4%, maintains consistently low SLA violation behavior, and supports controlled migration activity. The logged response time of 10.0000 ms per completed task/VM event further supports interval-based scheduling feasibility. Overall, the results confirm that attention-guided global reasoning improves scheduling stability, energy efficiency, and SLA reliability under dynamic cloud and edge–cloud workloads. Operational cost is included as a reward component, while detailed economic billing analysis, multi-tenant fairness, and real-world cloud deployment remain important future extensions.

References

[1] S. Gupta *et al.*, "Efficient prioritization and processor selection schemes for heft algorithm: A makespan optimizer for task scheduling in cloud environment," *Electronics*, vol. 11, no. 16, p. 2557, 2022.

[2] T. Bezdan, M. Zivkovic, N. Bacanin, I. Strumberger, E. Tuba, and M. Tuba, "Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm," *Journal of Intelligent & Fuzzy Systems*, vol. 42, no. 1, pp. 411–423, 2021.

[3] R. Ghafari, F. H. Kabutarkhani, and N. Mansouri, "Task scheduling algorithms for energy optimization in cloud environment: a comprehensive review," *Cluster Computing*, vol. 25, no. 2, pp. 1035–1093, 2022.

[4] P. Tank and R. Jain, "Big Data in the Internet of Things IoT Sensor Data Analysis and Edge Computing," in *Advancements in Cloud-Based Intelligent Informative Engineering*: IGI Global Scientific Publishing, 2025, pp. 79–94.

[5] L. H. Al-Farhani, Y. Alqahtani, H. A. Alshehri, R. J. Martin, S. Lalar, and R. Jain, "[Retracted] IOT and Blockchain-Based Cloud Model for Secure Data Transmission for Smart City," *Security and Communication Networks*, vol. 2023, no. 1, p. 3171334, 2023.

[6] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, "An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1531–1541, 2020.

[7] S. E. Shukri, R. Al-Sayyed, A. Hudaib, and S. Mirjalili, "Enhanced multi-verse optimizer for task scheduling in cloud computing environments," *Expert Systems with Applications*, vol. 168, p. 114230, 2021.

[8] P. Pirozmand, H. Jalalinejad, A. A. R. Hosseinabadi, S. Mirkamali, and Y. Li, "An improved particle swarm optimization algorithm for task scheduling in cloud computing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, pp. 4313–4327, 2023.

[9] S. K. Balam, R. Jain, J. S. Alaric, B. Pattanaik, and T. B. Ayele, "Renewable energy integration of IoT systems for smart grid applications," in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2023: IEEE, pp. 374–379.

[10] R. Jain, Y. Bekuma, B. Pattanaik, A. Assebe, and T. Bayisa, "Design of a smart wireless home automation system using fusion of iot and machine learning over cloud environment," in *2022 3rd international conference on intelligent engineering and management (ICIEM)*, 2022: IEEE, pp. 840–847.

[11] R. Medara and R. S. Singh, "Energy efficient and reliability aware workflow task scheduling in cloud environment," *Wireless Personal Communications*, vol. 119, no. 2, pp. 1301–1320, 2021.

[12] A. Amini Motlagh, A. Movaghar, and A. M. Rahmani, "Task scheduling mechanisms in cloud computing: A systematic review," *International Journal of Communication Systems*, vol. 33, no. 6, p. e4302, 2020.

[13] M. Masdari and M. Zangakani, "Efficient task and workflow scheduling in inter-cloud environments: challenges and opportunities: M. Masdari, M. Zangakani," *The Journal of Supercomputing*, vol. 76, no. 1, pp. 499–535, 2020.

[14] R. Jain and M. Varshney, "Group Key Management Protocols for Non-Network: A survey," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 20, no. 1, pp. 214–225, 2024.

[15] R. Jain and M. Varshney, "A Trust-Based Group key management protocol for Non-Networks," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 11, no. 6s, pp. 483–489, 2023.

[16] G. Rjoub, J. Bentahar, and O. A. Wahab, "BigTrustScheduling: Trust-aware big data task scheduling approach in cloud computing environments," *Future Generation Computer Systems*, vol. 110, pp. 1079–1097, 2020.

[17] M. Lavanya, B. Shanthi, and S. Saravanan, "Multi objective task scheduling algorithm based on SLA and processing time suitable for cloud environment," *Computer Communications*, vol. 151, pp. 183–195, 2020.

[18] S. C. Nayak, S. Parida, C. Tripathy, and P. K. Pattnaik, "An enhanced deadline constraint based task scheduling mechanism for cloud environment," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 2, pp. 282–294, 2022.

[19] S. Swarup, E. M. Shakshuki, and A. Yasar, "Task scheduling in cloud using deep reinforcement learning," *Procedia Computer Science*, vol. 184, pp. 42–51, 2021.

[20] S. H. H. Madni, M. S. Abd Latiff, M. Abdullahi, S. i. M. Abdulhamid, and M. J. Usman, "Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment," *PloS one*, vol. 12, no. 5, p. e0176321, 2017.

[21] D. Alsadie, "Advancements in heuristic task scheduling for IoT applications in fog-cloud computing: challenges and prospects," *PeerJ Computer Science*, vol. 10, p. e2128, 2024.

[22] F. Juarez Pérez, J. Ejarque, and R. M. Badia Sala, "Dynamic energy-aware scheduling for parallel task-based application in cloud computing."

[23] J.-M. Pierson, P. Stolf, H. Sun, and H. Casanova, "MILP formulations for spatio-temporal thermal-aware scheduling in Cloud and HPC datacenters," *Cluster Computing*, vol. 23, no. 2, pp. 421–439, 2020.

[24] S. Mangalampalli *et al.*, "Efficient deep reinforcement learning based task scheduler in multi cloud environment," *Scientific Reports*, vol. 14, no. 1, p. 21850, 2024.

[25] X. Yu, J. Mi, L. Tang, L. Long, and X. Qin, "Dynamic multi objective task scheduling in cloud computing using reinforcement learning for energy and cost optimization," *Scientific Reports*, 2025.

[26] B. Kruekaew and W. Kimpan, "Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning," *Ieee Access*, vol. 10, pp. 17803–17818, 2022.

[27] A. R. Khan, "Dynamic load balancing in cloud computing: optimized RL-based clustering with multi-objective optimized task scheduling," *Processes*, vol. 12, no. 3, p. 519, 2024.

[28] P. Varshney and Y. Simmhan, "Characterizing application scheduling on edge, fog, and cloud computing resources," *Software: Practice and Experience*, vol. 50, no. 5, pp. 558–595, 2020.

[29] Y. Ye and P. Ruan, "Enhanced Osprey Optimization Algorithm for task scheduling in cloud computing environment," *Journal of Engineering and Applied Science*, vol. 72, no. 1, p. 141, 2025.

[30] Z. Chen, L. Zhang, X. Wang, and K. Wang, "Cloud–edge collaboration task scheduling in cloud manufacturing: An attention-based deep reinforcement learning approach," *Computers & Industrial Engineering*, vol. 177, p. 109053, 2023.

[31] X. Wang, L. Zhang, Y. Liu, and C. Zhao, "Logistics-involved task scheduling in cloud manufacturing with offline deep reinforcement learning," *Journal of Industrial Information Integration*, vol. 34, p. 100471, 2023.

[32] Z. Wang, M. Goudarzi, and R. Buyya, "TF-DDRL: A transformer-enhanced distributed DRL technique for scheduling IoT applications in edge and cloud computing environments," *IEEE Transactions on Services Computing*, vol. 18, no. 2, pp. 1039–1053, 2025.

[33] K. Li, "Design and analysis of heuristic algorithms for energy-constrained task scheduling with device-edge-cloud fusion," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 2, pp. 208–221, 2022.