Adaptive Virtual Machine Consolidation Based on Autoformer and Enhanced Double Q-Network for Energy-Efficient Cloud Data Center

Kaiqi Zhang, Youbo Lyu, Dequan Zheng*, Yanping Chen, Jianshan Xu
School of Computer and Information Engineering, Harbin University of Commerce, Harbin 150028, China

***Abstract*—As the scale of cloud data centers continues to expand, energy consumption has become a critical issue. Virtual machine (VM) consolidation is a key technology for improving resource utilization and reducing energy consumption, yet it remains challenging to effectively balance energy efficiency with service level agreement violations (SLAV) in dynamic cloud environments. This paper proposes an adaptive VM consolidation strategy based on Autoformer and an enhanced dual Q-Network, referred to as AEDQN-VMC. The approach consists of three integrated components: 1) Autoformer-based load detection, which leverages an autocorrelation mechanism to decompose time-series data into multi-scale trend and periodic components; 2) a VM selection method that integrates the Pearson correlation coefficient and migration time to optimize the selection of VMs for migration; and 3) an enhanced dual Q-Network for VM placement, incorporating the upper confidence bound (UCB) and adaptive learning rate (ALR) to improve the exploration-exploitation trade-off. Extensive experiments on real-world cloud workload traces (PlanetLab, Google Cluster, and Alibaba datasets) demonstrate that the proposed method significantly outperforms state-of-the-art benchmarks such as PABFD, AD-VMC, and AMO-VMC. Specifically, it achieves maximum reductions of 46.5% in energy consumption and 74.2% in SLAV rate. Ablation studies further validate the contribution of each component and confirm the synergistic effect of the overall architecture. The results highlight the potential of AEDQN-VMC as an efficient and reliable solution for sustainable cloud data center operations.**

Keywords*—Cloud computing; virtual machine consolidation; load prediction; energy efficiency; deep reinforcement learning; Autoformer

I. Introduction

With the rapid development of cloud computing, data centers have become the core infrastructure supporting various online services, including big data analysis, artificial intelligence, and Internet of Things (IoT) applications. The exponential growth of user demands has driven a significant expansion in the scale of data centers, which typically consist of thousands of physical servers. However, this expansion gives rise to a critical issue: excessive energy consumption, with a considerable portion of energy wasted due to low resource utilization efficiency [1]. High energy consumption not only increases the operational costs for data center operators but also exacerbates carbon emissions, conflicting with global sustainable development and carbon neutrality goals. Therefore, it has become an urgent challenge to reduce energy consumption while ensuring quality of service (QoS) in the field of data center management.

Virtual Machine (VM) consolidation is a core strategy to address the energy consumption issue of data centers. By dynamically migrating VMs between physical machines (PMs), VM consolidation optimizes resource allocation. Through consolidating VMs onto a smaller number of active PMs, idle or underloaded PMs can be shut down or switched to low-power modes, thereby reducing overall energy consumption and improving resource utilization [2, 3]. However, VM consolidation is a complex decision-making process that requires balancing multiple objectives, such as minimizing energy consumption, reducing Service Level Agreement Violation (SLAV), and lowering migration overhead.

The VM consolidation process is divided into three phases: PM state detection, VM selection, and VM placement. Therefore, accurate PM state detection, reasonable VM selection, and optimized VM placement are the three major challenges that must be addressed to solve the VM consolidation problem. To tackle these challenges, Goyal et al. [4] proposed an adaptive multi-objective VM consolidation strategy (AMO-VMC), which aims to optimize resource management in energy-efficient cloud data centers. This strategy fuses the future resource utilization and the historical utilization for identifying overutilized PMs and selecting VMs to be migrated, and a multi-objective heuristic-based adaptive VM placement algorithm is adopted to select the optimal target host. Zeng et al. [5] proposed an adaptive Deep Reinforcement Learning(DRL)-based VM consolidation framework (AD-VMC), whose core components are an influence coefficient(IC)-based VM Selection algorithm (ICVMS) and a prediction aware DRL-based VM placement method(PADRL). By calculating the similarity between VMs and PMs, VMs that have the greatest impact on PM overload are migrated first to quickly relieve load pressure. Additionally, Long Short-Term Memory (LSTM) Neural Network is used to predict future system states to accelerate the convergence of the DRL model, thereby achieving energy-efficient VM placement.

However, existing researches still have significant limitations and struggle to simultaneously meet the requirements of energy efficiency optimization and QoS assurance in dynamic cloud environments. In the PM state detection phase, although workload prediction-based PM state detection methods are widely adopted, and deep learning-based prediction models can capture the temporal dependencies of

*Corresponding author.

workloads [6, 7], their ability to model the complex multi-scale periodicity (e.g., intra-day and intra-week fluctuations) in cloud workloads remains limited. This may lead to misjudgment of overloaded/underloaded hosts. In the VM selection phase, although some studies consider both resource correlation and migration cost [4, 8], they fail to effectively model the non-linear relationship between correlation and cost. In the VM placement phase, researchers have proposed several VM placement schemes based on DRL [9, 10]. However, these schemes adopt fixed exploration probabilities and learning rates, making it difficult to balance exploring new actions and exploiting known optimal actions, thus resulting in the problem of exploration-exploitation imbalance.

To address the aforementioned limitations, an adaptive VM consolidation scheme based on Autoformer and enhanced double Q-Network (AEDQN-VMC) is proposed, which achieves efficient resource management in dynamic environments through synergistic optimization across three phases:

- Autoformer-based PM state detection: Autoformer is employed for workload prediction, which decomposes workload time series into trend and periodic components via an auto-correlation mechanism.
- VM selection based on Pearson correlation coefficient and migration Time: A VM migration impact factor is constructed by fusing the Pearson correlation coefficient and migration time to balance the effect of load reduction and migration cost.
- VM placement based on enhanced double Q-Network: The upper confidence bound (UCB) and adaptive learning rate (ALR) are introduced into the double Q-Network to optimize the exploration-exploitation balance.

The remainder of this paper is organized as follows. Section II presents a classified review of related research on VM consolidation. Section III describes the system model and overall framework. Section IV details the Autoformer-based PM state detection method. Section V proposes the VM selection strategy based on Pearson correlation coefficient and migration time. Section VI introduces the VM placement algorithm based on the enhanced double Q-Network. Section VII evaluates the effectiveness of the proposed strategy through experiments. Section VIII concludes the paper and outlines future research directions.

II. Related Works

As a core strategy for resource management in energy-efficient cloud data centers, VM consolidation aims to significantly reduce system energy consumption by centrally deploying VMs on a minimized cluster of PMs, thereby maximizing physical resource utilization and minimizing the number of active servers. This strategy runs through the entire lifecycle management of VMs. In the initial VM creation phase, VM consolidation manifests as placement decisions based on PM state detection. During the dynamic migration process in the operation phase, VM consolidation involves a three-stage optimization of PM state detection, VM selection, and VM placement. To systematically analyze the inherent logic of the consolidation mechanism, the following sections will sort out existing studies from three core dimensions: PM state detection, VM selection, and VM placement.

A. PM State Detection

Before selecting VMs and making placement decisions, it is necessary to accurately detect the operational status of PMs. Existing PM detection methods can be categorized into two types based on whether they rely on workload trend prediction:

1) Non-prediction-based: This type of method directly determines the PM status using real-time or historical workload data, without the need to predict future workload changes. In [11], PM state detection primarily relies on real-time resource utilization, with CPU and memory as core dimensions. This involves calculating the available resources of PMs and classifying PM states into overloaded, normal, and underloaded by setting thresholds. The study [12] classifies workloads using the split-and-recombine (SAR) algorithm based on real-time monitored resource utilization and workload characteristics, and further considers thermal cycling effects through a thermal model to determine the host state. The study in [13] calculates power consumption across different utilization intervals using a linear interpolation power model, and then evaluates whether a PM can accommodate additional VMs by combining current and historical CPU utilization. The study [14] relies on real-time temperature monitoring to periodically check server temperatures, which dynamically sets a maximum temperature threshold and determines if the server is overloaded when the temperature exceeds this threshold.

2) Load prediction-based: To overcome the response lag of non-prediction-based methods, researchers have introduced workload prediction technique, which enables proactive state identification by forecasting the future resource utilization of PMs. The core value of such methods lies in providing a buffer time window for migration decisions, thereby avoiding SLAVs and reducing the frequency of emergency migrations.

Statistical methods are traditional workload prediction approaches and remain the mainstream modeling techniques for workload prediction to date. The study [15] uses the ARIMA model to analyze the trend of PM resource utilization, predicting future resource utilization to determine whether a PM is overloaded. Reference [16] predicts the future workload of PMs via the exponential smoothing method, which calculates a workload anomaly function by combining the predicted future workload with the current workload, and then determines whether a PM is in an overloaded state. The study [17] infers the future overload probability by analyzing historical host resource usage with the Local Regression algorithm, thereby identifying whether a PM is in an abnormal state.

However, statistical methods have obvious limitations when dealing with complex dynamic workloads. First, most of these methods rely on the assumptions of linearity and stationarity of data. In reality, however, data center workloads

often exhibit nonlinear and non-stationary characteristics, accompanied by sudden fluctuations, which leads to a decline in prediction accuracy. Second, traditional statistical models have poor adaptability to high-dimensional data and struggle to integrate multi-source information, resulting in limited generalization ability in complex scenarios. Third, for workload sequences with long-term dependency characteristics, simple smoothing or regression methods are unable to capture the deep correlation between historical data and future trends, making them prone to lag errors.

To address the limitations of statistical methods, researchers have increasingly adopted deep learning techniques to construct workload prediction models in recent years. The study [10] combines Temporal Convolutional Networks (TCN) with Median Absolute Deviation (MAD) to predict the overload probability of PMs, which reduces unnecessary VM migrations and SLAVs. The study [5] designs an LSTM-based network to forecast future system states, which provides more reasonable environmental states for the DRL model, assisting in more accurate judgments of whether PMs are overloaded or underloaded. The study [18] proposes a virtualized adaptive scheduling algorithm, which uses LSTM to predict future CPU utilization, memory usage, and network bandwidth requirements of VMs, and employs Deep Q-Network (DQN) to achieve adaptive resource scheduling.

Nevertheless, deep learning-based workload prediction confronts several issues, such as high model complexity, substantial training costs, and sensitivity to small-sample data. To further optimize prediction performance, researchers have combined statistical methods with deep learning to construct hybrid workload prediction models. For instance, the study [4] combines the Dynamic Weighted Moving Average (DWMA) and a neural network model to calculate the expected utilization rate of physical machines. The study [19] combines Bollinger Bands and Neural Prophet techniques to predict the future workload trend of hosts.

B. VM Selection

After identifying the source PMs that require adjustment, it is necessary to further select the VMs to be migrated. Based on the criteria used to select VMs from overloaded PMs, relevant studies on VM selection can be categorized into three types: based on PM-VM correlation, based on VM migration overhead, and based on both correlation and migration overhead.

1) PM-VM correlation: The VM selection criteria in this category primarily focus on the degree of association between VMs and PMs in aspects such as resource usage patterns, workload trends, and energy consumption characteristics. By quantifying this association, VMs that have a significant impact on the PM's state are selected. In study [5], VMs are selected based on the Influence Coefficient, which quantifies a VM's contribution to the overload of a PM by calculating the product of the cosine similarity between the VM's and PM's resource usage. VMs with higher influence coefficient exhibit a stronger correlation with the PM's overload and are thus prioritized for migration. In study [20], a load consolidation strategy based on Q-learning selects VMs by learning the load matching patterns between VMs and PMs. Specifically, it prioritizes the migration of VMs that can balance the PM's load, which reflects the correlation between VM and PM load trends. In study [15], VMs are selected based on contribution of the VM to the overload of the PM. VMs with the largest contribution are prioritized for migration, with the goal of eliminating the overload trend of the PM. In [21], with the goal of minimizing energy consumption, VMs that are compatible with the energy consumption characteristics of PMs are selected.

2) VM migration overhead: The VM selection criteria in this category primarily focus on the costs incurred during the migration process (e.g., time, performance loss, energy consumption, and impacts on QoS). By optimizing the selection of VMs, the negative impacts of migration on the system are reduced. In study [11], a fuzzy meta-heuristic algorithm is employed to select VMs whose migration imposes the least impact on QoS. In study [22], the direct criterion for VM selection is the migration time of VMs. The Minimum Migration Time (MMT) algorithm is adopted to prioritize the selection of VMs that can be migrated from overloaded hosts in the shortest time.

3) Both correlation and migration overhead: The criteria for VM selection in this type of research take into account both the correlation between VMs and PMs and the overhead during the migration process. The core is to balance these two aspects to achieve the optimization goal. For example, in study [8], the objective function aims to simultaneously minimize the number of active PMs and the frequency of VM migrations, thereby balancing resource utilization and migration costs. In study [23], a Deep-Q Network is employed to learn the load matching patterns between VMs and PMs, taking into account the impacts of migration on VM performance. It selects VMs that can balance load optimization and low performance loss. In [24], the correlation between VMs and the overload of PMs is determined to select VMs that can alleviate PM overload, taking into account migration time and performance degradation.

C. VM Placement

VM Placement (VMP) refers to the process of allocating newly created VMs or VMs to be migrated to optimal PMs, under the premise of satisfying multi-dimensional resource constraints such as CPU, memory, storage, and network. The rationality of this process directly affects resource utilization, energy consumption, and service quality. In recent years, researchers have proposed a variety of optimization methods to address this problem, which can be categorized into the following three types based on their technical approaches.

1) Heuristic algorithms: These algorithms generate placement strategies based on intuitive experience or predefined rules, with the goal of quickly obtaining feasible solutions, which is suitable for large scale dynamic scenarios. In study [8], a modified First-Fit strategy combined with a backfilling mechanism is adopted. Based on the runtime and priority of VMs, this approach places each VM onto the first

PM that meets the resource constraints. In study [25], the proposed cut-and-solve algorithm is a tree-search heuristic algorithm, which achieves the optimized placement of VMs by iteratively decomposing the problem and solving the sub-problems. In study [24], the VM allocation problem is modeled as a multi-dimensional bin packing problem, the objective of which is to select the minimum number of energy-efficient PMs, thereby to reduce energy consumption and resource waste. In study [26], a two-stage heuristic algorithm is proposed to address the rack-level hot-spot issue. In the first stage, VMs are selected for migration from overloaded hosts within racks identified as hard hot-spots, aiming to reduce the power consumption of these racks. In the second stage, VMs are selected for migration from underloaded hosts; this step serves to reduce the number of active servers, thereby achieving resource consolidation.

Heuristic algorithms offer the advantage of strong real-time performance. However, they struggle to balance multi-objective optimization (e.g., simultaneously minimizing energy consumption and achieving load balancing) and tend to get trapped in local optima.

2) Meta-heuristic algorithms: To address the limitations of heuristic algorithms, meta-heuristic algorithms perform global optimization by simulating natural phenomena (e.g., biological evolution, swarm behavior), which are suitable for complex multi-objective scenarios. The study [11] proposes a hybrid algorithm named IVPTS, which is designed for the collaborative optimization of task scheduling and VM placement in cloud computing environments. This algorithm integrates the Improved Particle Swarm Optimization algorithm with the Fuzzy Resource Management framework to achieve the goals of load balancing and energy conservation. To simultaneously optimize QoS, server energy consumption, and cooling energy consumption, The study [12] proposes a VM Placement strategy named BTVMP, which is based on the Enhanced Simulated Annealing algorithm and integrates a thermal cycling effect model. In study [14], the VM placement involves both heuristic and meta-heuristic methods. In the initial placement phase, a hybrid algorithm combining genetic algorithm and simulated annealing algorithm is adopted to achieve global optimization of the initial VM allocation. In the dynamic migration phase, a dynamic migration strategy based on the greedy algorithm is employed, which selects the server with the highest energy efficiency as the migration target.

3) Reinforcement learning approaches: With the development of machine learning, Reinforcement Learning (RL) and Deep Reinforcement Learning (DRL) have been applied to address the dynamic optimization problem of VM placement. The core of this method lies in modeling the placement process as a Markov Decision Process (MDP), where an agent learns the optimal strategy through interaction with the environment. In study [5], VM placement is implemented based on DRL. An LSTM-based state prediction network provides future environmental states, which accelerates the convergence of the DRL model and enables it to adaptively select the optimal target host for migrated VMs. To address the issue that existing methods optimize sub-steps (such as selecting target PMs for migration and choosing services to be migrated) independently while ignoring the correlations between decisions, Reference [9] proposes a micro-service migration framework based on Multi-Agent RL, which jointly optimizes the three sub-steps of the migration process: selecting source PMs, selecting micro-services, and selecting target PMs. In study [27], a VM migration management strategy based on RL is proposed, which adopts a decentralized operation mode through parallel learning and state/action space aggregation. In study [28], an energy-efficient dynamic consolidation method named MAGNETIC based on the multi-agent Q-Learning algorithm is proposed, in which the agent dynamically selects the optimal power mode to balance energy consumption and QoS.

RL approaches offer the advantage of strong adaptability, enabling them to dynamically adjust strategies to accommodate the uncertainties of cloud environments. However, they face challenges such as high training costs and insufficient convergence stability.

III. System Model and Framework

A. System Model

Suppose a data center consists of *N* PMs denoted as $H = \{H_1, H_2, \dots, H_N\}$, where each PM H_i runs n_i VMs denoted as $V = \{V_1, V_2, \dots, V_{n_i}\}$. VM consolidation refers to the process of reasonably allocating a large number of VMs to PMs, with the core objective of optimizing resource utilization and reducing energy consumption while meeting QoS requirements and resource constraints.

1) Energy Consumption: The energy consumption of a PM is represented using a linear model as follows:

$$E_i = E_{idle} + (E_{max} - E_{idle}) \cdot u_i \quad (1)$$

where E_i denotes the total energy consumption of the PM H_i, E_{idle} represents the base energy consumption in the idle state, E_{max} is the maximum energy consumption under full load, and $u_i \in [0, 1]$ stands for the CPU utilization rate.

The total energy consumption of the entire data center is the sum of the energy consumption of all active PMs:

$$EC = \sum_{i=1}^{N} E_i \cdot \delta_i \quad (2)$$

where $\delta_i \in \{0, 1\}$ indicates the activation state of the PM H_i.

2) Service Level Agreement Violation: Service Level Agreement Violation (SLAV), a key metric for measuring whether VM consolidation affects QoS, refers to the number or proportion of times the QoS specified in the SLA fails to be met. In general, SLAV depends on two metrics: one is the SLAV time of each active host (SLATAH), and the other is the performance degradation caused by migration (PDM). SLAV can be expressed as:

$$SLAV = SLATAH \times PDM \quad (3)$$

where SLATAH represents the proportion of the total operating time that is occupied by SLA violation time in the service agreements of each active host, which is expressed as:

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{H_i}^{OVER}}{T_{H_i}^{TOTAL}} \quad (4)$$

where N is the total number of all PMs. For any PM H_i, the total duration during which it violates the SLA due to overload is recorded as $T_{H_i}^{OVER}$, and the total duration during which this host is in an active state is $T_{H_i}^{TOTAL}$.

In study (3), PDM is used to represent the proportion of performance degradation caused by VM migration to its total performance requirements, which is expressed as:

$$PDM = \frac{1}{n}\sum_{j=1}^{n}\frac{S_{d,j}^{OVER}}{R_{d,j}^{TOTAL}} \quad (5)$$

where n represents the number of VMs, d denotes the resource type. For any VM V_j, the amount of performance degradation caused by migration is marked as $S_{d,j}^{OVER}$, and the total resource demand of this VM during its lifecycle is recorded as $R_{d,j}^{TOTAL}$.

B. System Framework

The framework of the VM consolidation system is illustrated in Fig. 1. Adopting a modular design, this framework decomposes the entire VM consolidation process into three key phases, forming a complete closed-loop control system.

- Workload Detection Module: Monitoring the resource utilization of all PMs in real time and outputs three types of PM states.
  - Overloaded PMs: PMs predicted to exceed the safety threshold.
  - Underloaded PMs: Idle PMs whose resource utilization remains below the minimum threshold.
  - Normal PMs: PMs operating within the ideal load range.
- VM Selection Module: For overloaded PMs, VMs are selected to be migrated out. For underloaded PMs, all VMs running on them are marked as to-be-migrated.
- VM Placement Module: The optimal target PM is selected from normal PMs to place the VMs selected in the VM selection module. After executing the migration, idle PMs are shut down to save energy.

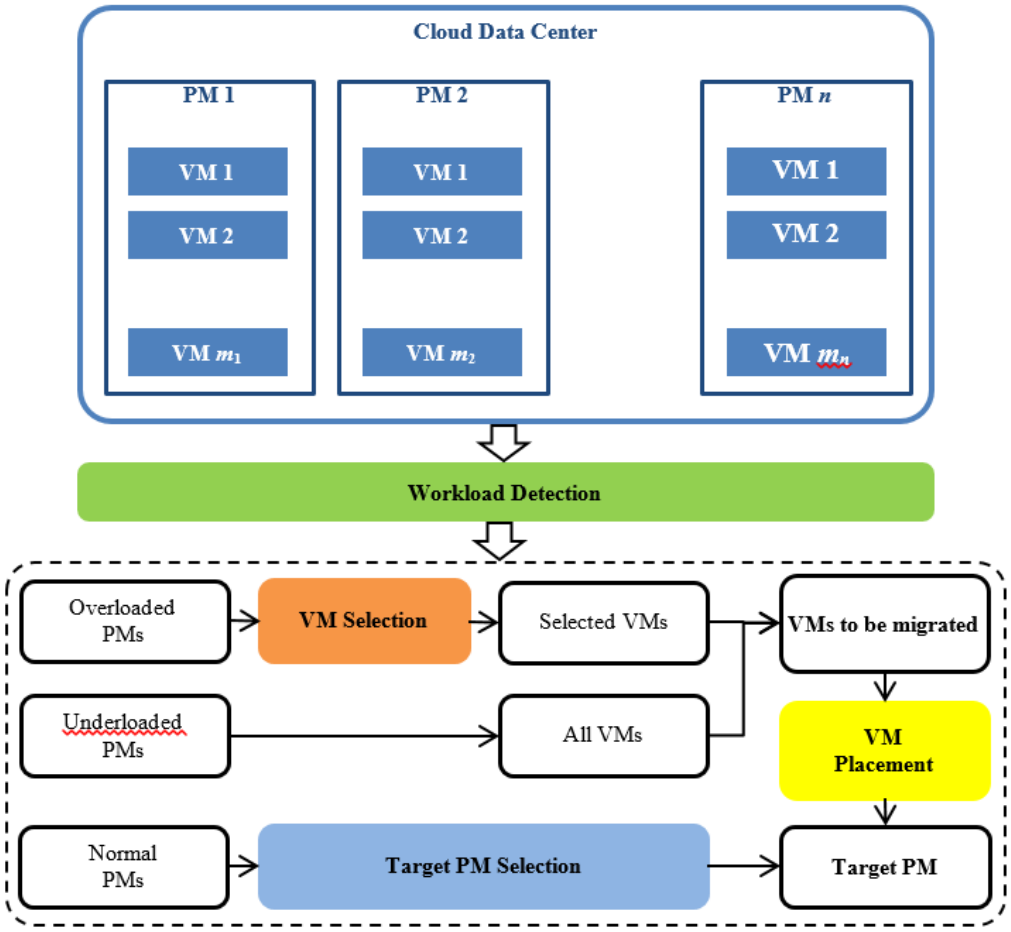


Fig. 1. Framework of VM consolidation.

IV. Workload Detection Based on Autoformer

Cloud workload time-series data typically contain both linear and non-linear components, exhibiting complex time-varying characteristics and significant temporal dependence. In response to this feature, a prediction architecture based on Autoformer is adopted [29], whose core components consist of an encoder and a decoder, as shown in Fig. 2.

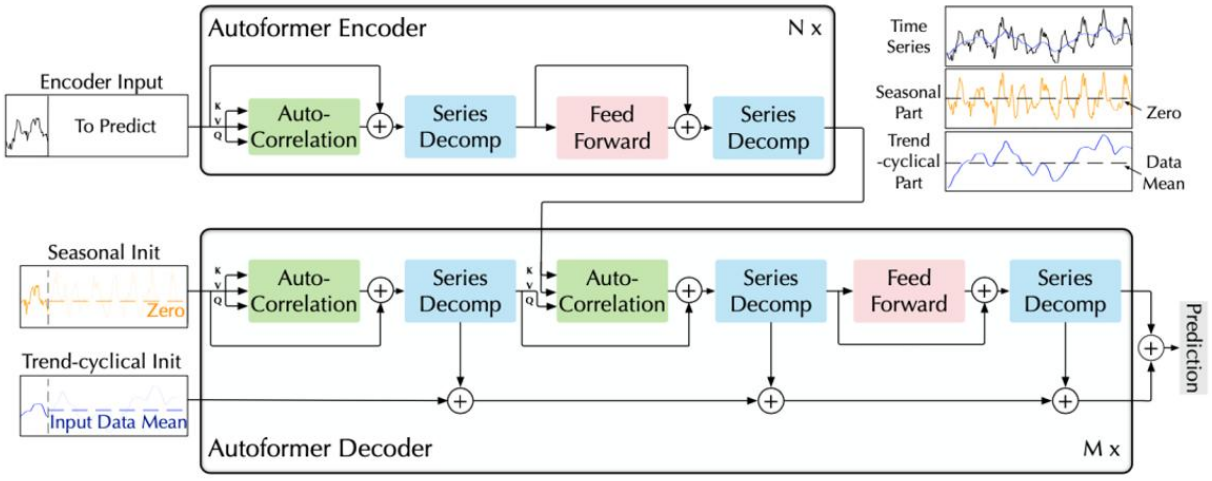


Fig. 2. Cloud workload prediction architecture based on Autoformer [29].

The encoder module takes historical workload time-series as input and performs multi-scale decomposition on the original time-series data through the Auto-Correlation Mechanism. This decomposition decouples the data into two orthogonal components:

- workload trend component which characterizes the long-term evolution law of resources;
- workload periodic component which depicts the inherent periodic fluctuation pattern of the system.

The decoder module adopts a dual-branch structure to process these two components separately. For the periodic component, progressive time-series feature extraction is achieved through cascaded auto-correlation blocks. For the trend component, a cumulative summation operation is used for multi-order trend information aggregation. Each layer of the decoder receives complete time-series segments rather than discrete sampling points. This global perception mechanism significantly enhances the model's ability to model complex workload patterns.

The workload detection process based on Autoformer is shown in Algorithm 1. In the initialization phase, the sets of PMs in overloaded, underloaded, and normal states are initialized as empty sets respectively, and the trained Autoformer model is loaded simultaneously. Next, the algorithm processes each PM in sequence. Line 3 obtains the current utilization rate of the PM by calling the getCurrentUtilization function, while Line 4 uses the getHistoryUtilization function to acquire the historical utilization sequence of the PM according to the window step size. This sequence contains resource usage data within a certain time span and serves as a key input for model prediction. Line 5 inputs the historical utilization sequence into the Autoformer model. Through its encoder, the historical workload time series undergoes multi-scale decomposition to separate the workload trend component and workload periodic component. These two components are then processed separately by the dual-branch structure of the decoder, ultimately outputting the predicted utilization rate. Lines 6–12 achieve accurate classification of PM states by comparing the current utilization rate and predicted utilization rate with the overload threshold and underload threshold. If the current utilization rate or the predicted utilization rate exceeds the overload threshold, the PM is determined to be in an overloaded state. If the predicted utilization rate is lower than the underload threshold and all data in the historical utilization sequence are below the underload threshold, it indicates that the host's resource utilization has been continuously low in the past and will remain low for a period in the future, so the PM is judged to be in an underloaded state. In all other cases, the host is classified as being in a normal state.

Algorithm 1: DetectWorkload

Input: physical machines $H = \{H_1, H_2, \dots, H_N\}$
window steps I
overload and underload threshold th_{over}, th_{under}

Output: overloaded PMs, underloaded PMs, normal PMs

Initialize:

1 $overloaded \leftarrow \emptyset$, $underloaded \leftarrow \emptyset$, $normal \leftarrow \emptyset$
Autoformer ← **load pretrained Autoformer model**

2 **For** each host H_i in H **Do**

3 $current\_util \leftarrow$ getCurrentUtilization(H_i)

4 $util\_sequence \leftarrow$ getHistoryUtilization(H_i, I)

5 $predicted\_util \leftarrow$ ***Autoformer*****.predict(** ***util_sequence*** **)**

6 **If** $current\_util > th_{over}$ OR $predicted\_util > th_{over}$ **Then:**

7 $overloaded \leftarrow overloaded \cup \{H_i\}$

Else If *predicted_util* < th_{under} AND all(*util* < th_{under} for *util* in *util_sequence*) **Then:**

    underloaded ← *underloaded* ∪ { H_i}

Else:

    normal ← *normal* ∪ { H_i}

End If

End For

return *overloaded*, *underloaded*, *normal*

V. VM Selection Based on Pearson Correlation Coefficient and Migration Time

In cloud computing environments dynamically changing, VM selection strategy is a key link that affects consolidation efficiency. An unreasonable VM selection strategy may lead to low migration efficiency or the risk of secondary overload. Therefore, this section proposes a comprehensive selection strategy that combines the Pearson correlation coefficient and migration time.

The Pearson correlation coefficient is a core metric for measuring the linear correlation between two variables. In the context of VM selection, it can accurately characterize the degree of correlation between the resource utilization of a VM and that of a PM. The equation is as follows:

$$\rho(V,H) = \frac{\sum_{t=1}^{T}(U_V^t-\mu_V)(U_H^t-\mu_H)}{\sqrt{\sum_{t=1}^{T}(U_V^t-\mu_V)^2}\sqrt{\sum_{t=1}^{T}(U_H^t-\mu_H)^2}} \quad (6)$$

where V denotes a VM, H represents a PM, and T stands for the length of the historical time window. U_V^t is the CPU utilization of the VM at time t, and U_H^t is the CPU utilization of the PM at time t. $\mu_V = \frac{1}{T}\sum_{t=1}^{T} U_V^t$ is the mean value of the CPU utilization time series of the VM, while $\mu_H = \frac{1}{T}\sum_{t=1}^{T} U_H^t$ is the mean value of the CPU utilization time series of the PM.

The time overhead for migrating virtual machine V from host H is modeled as:

$$MigTime(V,H) = Mem(V)/BW_{avail}(H) \quad (7)$$

where $\mathrm{Mem(V)}$ is the memory usage of the virtual machine, and $\mathrm{BW_{avail}(H)}$ represents the available network bandwidth of the host.

The VM migration impact factor based on the Pearson correlation coefficient and migration time is defined as follows:

$$IF(V,H) = \rho^2(V,H)/MigTime(V,H) \quad (8)$$

where the square operation strengthens the impact of VMs with high correlation, and the denominator term penalizes the migration of VMs with large memory.

Based on the definition of the VM migration impact factor, a VM selection algorithm that combines the Pearson correlation coefficient and migration time is proposed, as shown in Algorithm 2. This algorithm adopts a greedy strategy, selecting the VM with the largest current IF value for migration until the resource utilization of the overloaded host drops below the safety threshold.

Algorithm 2: SelectVMS

Input: overloaded hosts set $H_{overload}$

Output: migration list M

Initialize: $M \leftarrow \emptyset$

For each host H_i in $H_{overload}$ **Do**

    $V^* \leftarrow \underset{V \in H_i}{\mathrm{argmax}}\, IF(V,H_i)$

    $M \leftarrow M \cup \{V^*\}$

    $H_i \leftarrow H_i \setminus \{V^*\}$

End For

return M

VI. VM Placement Based on Enhanced Dual Q Network

VM placement refers to the reasonable allocation of VMs to PMs, aiming to optimize resource utilization and reduce energy consumption.

A. Problem Model

The VM placement framework based on the improved double Q neural network is illustrated in Fig. 3. The double Q neural network consists of two Q networks with identical structures but different parameter update methods: the online Q network QNet(θ) and the target Q network TargetQNet(θ^-). The online Q network is used to select action $\mathrm{a_t}$ based on the current environmental state $\mathrm{s_t}$, while the target Q network is employed to calculate the target Q-value. This design avoids the overestimation problem that may occur in traditional Q-learning, making the estimation of Q-values more accurate.

1) State representation: The environmental state s_t includes the CPU utilization information of each PM and the CPU demand information of the VM, which is defined as $s_t = [u_1^t, u_2^t, \ldots, u_i^t, \ldots, u_N^t, vm_{cpu}]$. Here, u_i^t denotes the CPU utilization of the i-th PM, and vm_{cpu} represents the CPU request information of the VM to be placed.

2) Action selection: Action a_t refers to placing the selected VM onto an appropriate PM. To balance exploration and exploitation, the upper confidence bound (UCB) strategy is adopted to improve action selection. For a given environmental state s_t, the optimal action is selected according to Eq. (9):

$$a_t^* = \underset{a \in A}{argmax}\, UCB(s_t,a) \quad (9)$$

where $\mathrm{UCB(s_t,a)}$ is defined as:

$$UCB(s_t,a) = Q_\theta(s_t,a) + c \cdot \sqrt{\ln N(s_t)/N(s_t,a)} \quad (10)$$

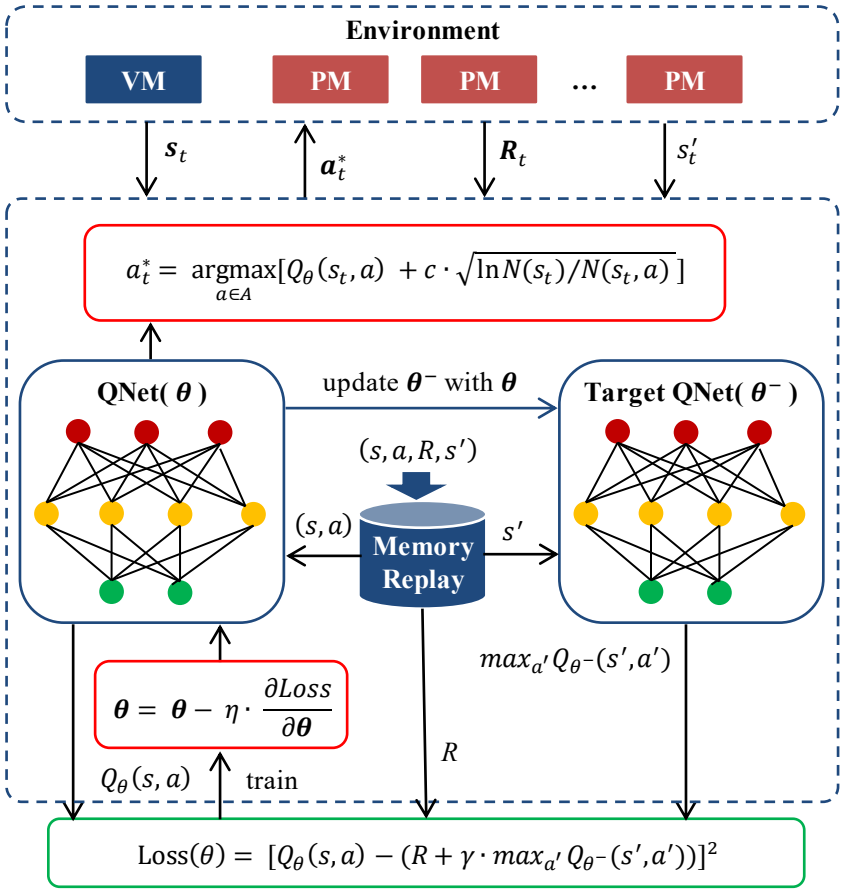


Fig. 3. VM placement framework based on enhanced deep Q-Network.

where $Q_\theta(s_t, a)$ is the value estimation of action a under state s_t from the online Q-network; c is the exploration coefficient; $N(s_t)$ represents the total number of times the state s_t has been visited; and $N(s_t, a)$ is the number of times the action a has been selected under state s_t. In this way, more diverse VM placement actions can be explored in the early stage. As experience accumulates, the focus gradually shifts to exploiting the optimal placement strategies that have been identified.

3) Reward: Based on the multi-objective optimization problem in the system model, the reward function is expressed as a function of energy consumption, SLAV rate, and migration count, which is defined as follows:

$$R_t = -(\alpha \cdot \frac{EC - EC_{min}}{EC_{max} - EC_{min}} + \beta \cdot \frac{SLAV - SLAV_{min}}{SLAV_{max} - SLAV_{min}} + \gamma \cdot \frac{M - M_{min}}{M_{max} - M_{min}}) \quad (11)$$

where $\alpha, \beta, \gamma \in [0, 1]$ are weight coefficients that satisfy $\alpha + \beta + \gamma = 1$; EC denotes total energy consumption, EC_{max} and EC_{min} represent the theoretical maximum and minimum energy consumption respectively; $SLAV_{max}$ and $SLAV_{min}$ are the theoretical maximum and minimum SLAV respectively; M indicates the migration count, M_{max} and M_{min} are the theoretical maximum and minimum migration counts respectively.

4) Loss Function: The online Q-network uses a mean squared error (MSE) loss function to calculate parameter gradients, which is defined by the following formula:

$$Loss(\theta) = E[(Q_\theta(s, a) - (R + \gamma \cdot max_{a'} Q_{\theta^-}(s', a')))^2] \quad (12)$$

where $Q_\theta(s, a)$ is the value estimation of the online Q-network for performing action a under state s; R denotes the obtained reward; $\gamma \in [0,1)$ is the discount factor; $Q_{\theta^-}(s', a')$ represents the value estimation of the target Q-network for action a′ under the next state s′; and $max_{a'} Q_{\theta^-}(s', a')$ refers to selecting the optimal action for the next state through the target Q-network.

5) Adaptive learning rate: An adaptive learning rate (ALR) is used for the training of the online Q-network, which is defined as follows:

$$\eta = \frac{1}{N(s,a)+1} \quad (13)$$

where $N(s, a)$ denotes the number of times action a has been selected under state s.

When the number of visits to a state-action pair is extremely small, $N(s, a) \approx 0$, and the learning rate $\eta \approx 1$ accordingly. A larger step size allows the parameters of the online Q-network to update rapidly, accelerating the learning process for value estimation of new decisions. As the same state-action pair is visited repeatedly, $N(s, a)$ increases, and the

learning rate η gradually approaches 0. A smaller step size prevents parameter oscillations caused by excessive updates, enabling the Q-value estimation to converge to a stable value and ensuring the accurate exploitation of known high-quality strategies.

B. VM Consolidation Algorithm

The VM consolidation algorithm is shown in Algorithm 3. The consolidation process consists of three parts: workload detection, VM detection, and VM placement. For each time window, the DetectWorkload function is executed first to detect the state of each PM. For overloaded PM, the SelectVMS function is executed to select the VMs that need to be migrated out. For underloaded PM, all VMs are migrated out. The VM placement phase is the core decision-making part of the algorithm, integrating deep reinforcement learning and the exploration-exploitation balance strategy.

The action selection adopts an improved variant of the ε-greedy strategy. On the basis of Q-network valuation, it introduces a confidence interval adjustment term composed of the exploration coefficient c, state-action counters N(s, a) and N(s). This design not only enables the exploitation of known high-quality placement schemes through the Q-network, but also encourages the exploration of low-frequency visited actions via $\sqrt{\ln N(s_t)/N(s_t,a)}$, thus preventing the algorithm from falling into local optimal solutions.

Algorithm 3. Virtual Machine Consolidation

Initialize:
the size of time window
the physical machines $H = \{H_1, H_2, \dots, H_N\}$
online net QNet(θ) with random weights θ
target net TargetQNet(θ^-) with weights $\theta^- \leftarrow \theta$
experience replay buffer D with capacity C
exploration coefficient c
discount factor γ

For each time step t **Do**
  Overload,Underload,Normal ← **DetectWorkload**(H)
  MigrationList ← **SelectVMS**(Overload)
  MigrationList ←*MigrationList*+VMs of underload PMs
  For each virtual machine V in *MigrationList* **Do**
    s_t = getCurrentState(*Normal*, V)
    $a_t^* = \underset{a \in A}{\operatorname{argmax}} [Q_\theta(s_t, a) + c \cdot \sqrt{\ln N(s_t)/N(s_t,a)}]$
    Place V on the host determined by the action a_t^*
    $R_t = -(\alpha \frac{EC - EC_{min}}{EC_{max} - EC_{min}} + \beta \frac{SLAV - SLAV_{min}}{SLAV_{max} - SLAV_{min}} + \gamma \frac{M - M_{min}}{M_{max} - M_{min}})$
    D ← D + (s, a, R, s')
  End For
  If D.size()>=batchsize AND (t%traininterval)==0 **Then**
    Batch = D.sample(batch size)
    For each sample (s, a, R, s') in Batch **Do**
      $\text{Loss}(\theta) = [Q_\theta(s,a) - (R + \gamma \cdot max_{a'} Q_{\theta^-}(s', a'))]^2$
      $\eta = \frac{1}{N(s,a)+1}$
      $\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \cdot \frac{\partial Loss}{\partial \boldsymbol{\theta}}$
    End For
  End If
  If (t % target update interval) ==0 **Then**
    $\theta^- \leftarrow \theta$
  End If
End For

The experience replay mechanism caches historical interaction data and performs random sampling for training when the batch condition is met. This breaks the temporal correlation between samples and improves the stability of Q-network training. The loss function adopts the squared temporal difference error, taking the deviation between the current Q-value and the target Q-value as the optimization objective. The learning rate η is linked to N(s, a), enabling state-action pairs that are less frequently visited to obtain a larger parameter update magnitude, thus accelerating the learning of sparse sample regions.

The periodic update of the target network further prevents oscillations in Q-value estimation. By copying the online network parameters θ to the target network parameters θ^- at fixed intervals, the calculation benchmark for target Q-values remains stable within a certain period, providing a reliable reference for the gradient descent of the online network.

Overall, through the closed-loop mechanism of perception-decision-learning, this algorithm achieves dynamic optimization of VM consolidation. Workload detection accurately identifies resource bottlenecks; VM selection focuses on key adjustment targets; VM placement integrates reinforcement learning to realize intelligent decision-making; and the design of experience replay and target network ensures the convergence and robustness of the algorithm in dynamic cloud environments. Ultimately, it achieves the comprehensive goals of improving resource utilization, reducing data center energy consumption, and ensuring service quality.

VII. Experimental Evaluation

A. Experimental Setting

To verify the effectiveness of the adaptive VM consolidation strategy based on Autoformer and Enhanced Double Q-Network (AEDQN-VMC) proposed in this paper, the experimental simulations were conducted by building a test environment on the CloudSim 4.0 simulation platform [30]. As a mainstream simulation tool in the cloud computing field, this platform supports the simulation of resource scheduling for PMs and VMs, as well as the quantitative calculation of key metrics such as energy consumption and SLA violation rate, and has been widely adopted in relevant research. To cover the load characteristics under different business scenarios and

verify the generalization ability of the strategy, three types of real cloud load datasets were used in the experiments: the PlanetLab dataset [31], the Google Cluster Trace dataset [32], and the Alibaba dataset [33].

1) Configuration of PMs and VMs: To ensure consistency with the experimental conditions of existing mainstream research, the hardware parameters of PMs and VMs are all set with reference to the experimental parameters in [5]. The specific configurations are as follows:

a) PMs: Two types of heterogeneous hosts (HP ProLiant G4 and HP ProLiant G5) are adopted. Each host is configured with 2 CPU cores and a unified memory capacity of 4GB. The core difference lies in processor performance, with specific parameters shown in Table I. A total of 100 physical machines are deployed in the data center in the experiment, and the quantity ratio of the two types of hosts is 1:1, so as to simulate the heterogeneous hardware environment of real data centers.

TABLE I. PM CONFIGURATION

PM Type	Processor (MIPS)	Num. of Cores	Memory (GB)
HP ProLiant G4	1860	2	4
HP ProLiant G5	2660	2	4

b) VMs: Four types of VM instances are designed to cover scenarios from lightweight to high-performance. The MIPS (Million Instructions per Second) of processor ranges from 500 to 2500, and the memory capacity matches the CPU requirements, with specific parameters shown in Table II. There are 200 VMs in total in the experiment, with 50 VMs of each type.

TABLE II. VM CONFIGURATION

VM Type	Processor(MIPS)	Memory(GB)
Micro	500	0.85
Small	1000	1.7
Extra Large	2000	3.75
High-CPU Medium	2500	0.85

2) Parameters of the energy consumption model: The energy consumption calculation of PMs refers to the SPECpower benchmark test data [34]. As an industry standard for data center energy consumption evaluation, this benchmark provides measured power consumption values under different CPU utilization rates. In the experiment, the power consumption data of the two types of hosts within the CPU utilization range of 0%–100% is shown in Table III. For intermediate utilization rates not covered in the table (e.g., 15%, 25%), the linear interpolation method is used to calculate continuous power consumption values, ensuring the accuracy of the energy consumption model.

3) Key parameter settings: Autoformer Load Prediction Parameters: The length of the historical time window is 24 time steps; the number of encoder/decoder layers is 2 each; the number of heads in the autocorrelation block of each layer is 4; and the hidden layer dimension is 64. The batch size is 128, the number of epochs is 100, the Adam optimizer is used, the initial learning rate is 0.01, and the loss function is mean squared error (MSE). The overload threshold of physical machines is 0.9, and the underload threshold is 0.2.

Enhanced Double Q-Network Parameters: A fully connected neural network is adopted as the network model for the double Q-network, and the online Q-network and target Q-network share an identical network structure. This fully connected neural network contains 3 hidden layers, with the number of nodes in each hidden layer being 256, 128, and 64 respectively. The exploration coefficient c in (10) is 0.5; α, β, γ in (11) are each 1/3; and the discount factor γ in (12) is 0.9. The training interval is 10 time steps, and the target network update interval is 100 time steps.

TABLE III. POWER CONSUMPTION AT DIFFERENT CPU UTIL. (WATTS)

CPU Util.	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
HP G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP G4	93.7	97	101	105	110	116	121	125	129	133	135

4) Comparison benchmarks and evaluation metrics: To verify the superiority of the AEDQN-VMC strategy, three VM consolidation strategies were selected as comparison benchmarks in the experiment:

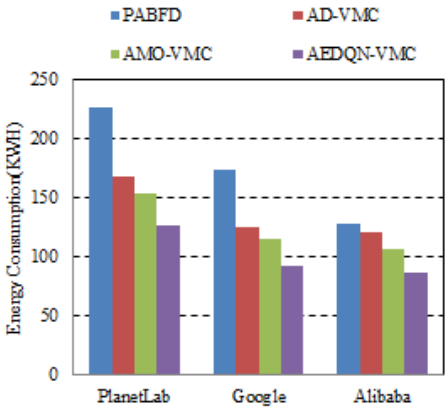
- PABFD (Power-Aware Best Fit Decreasing): An energy-aware strategy built into CloudSim [35], which selects the host with the minimum energy consumption increment after VM placement.
- ADVMC (Adaptive DRL-based VM Consolidation): A VM consolidation strategy proposed by [5], which is based on LSTM prediction and DQN placement.
- AMOVMC (Adaptive Multi-Objective Virtual Machine Consolidation): An VM consolidation strategy proposed by [4], which uses a neural network to predict future resource utilization, and adopts a multi-objective heuristic adaptive VM placement algorithm to select the optimal target host.

Two core evaluation metrics were used in the experiment: total energy consumption and SLAV rate. The calculation of total energy consumption refers to Eq. (2), and the calculation of SLAV rate refers to Eq. (3)–(5). The CPU performance loss caused by migration is set to 10% by default.

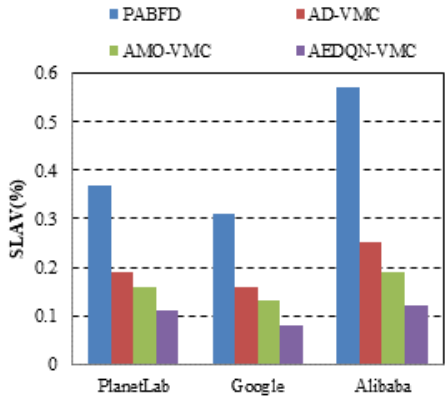
B. Performance Evaluation

To verify the effectiveness and advancement of the VM consolidation method AEDQN-VMC proposed in this paper, we conducted a comparative analysis between the proposed method and the benchmark methods. The comparison results are shown in Fig. 4. It can be seen that the energy consumption of the method proposed in this paper is significantly lower than that of the comparison methods on the three datasets (PlanetLab, Google, and Alibaba). Specifically, compared with the PABFD method, the energy consumption is reduced by 44.5%, 46.5%, and 32.6% respectively; compared with the ADVMC method, the energy consumption is reduced by

25.2%, 25.7%, and 28.3% respectively; and compared with the AMOVMC method, the energy consumption is reduced by 18%, 19.7%, and 18.8% respectively.



(a) Energy



(b) SLAV

Fig. 4. Comparison of performance among consolidation methods.

The performance improvement in energy consumption stems from the following aspects. The Autoformer-based workload detection module can accurately predict the load trend of PMs, identify overloaded and underloaded states in advance, and avoid redundant energy consumption caused by unbalanced resource allocation. The VM placement module achieves efficient resource utilization through the enhanced double Q-network. On the premise of meeting the resource requirements of VMs, it minimizes the number of active PMs, thereby reducing overall energy consumption. The timely shutdown strategy for underloaded hosts further reduces idle energy consumption and improves energy utilization efficiency.

The SLAV of the method proposed in this paper remains the lowest across all three datasets. Specifically, on the Google dataset, its SLAV is 0.08%, a reduction of 50% compared to the ADVMC method, a reduction of 38.5% compared to the AMOVMC method, and a reduction of 74.2% compared to the PABFD method. The accurate load prediction of the Autoformer model can avoid the risk of PM overload in advance and reduce service interruptions caused by resource saturation. The VM selection module screens migration targets using the Pearson correlation coefficient, prioritizing the migration of VMs that have less impact on host load. Meanwhile, by integrating migration time optimization, it reduces performance loss during the migration process, thereby lowering the overall SLAV.

C. Ablation Experiments

1) Impact of PM detection on consolidation: To verify the impact of the Autoformer-based host state detection (Autoformer-HSD) proposed in this paper on VM consolidation performance, we kept the VM selection algorithm and VM placement algorithm unchanged, and replaced the host state detection algorithm proposed in this paper with other host state detection algorithms. In this way, we obtained the VM consolidation performance based on different host state detection algorithms. The benchmark methods used to verify Autoformer-HSD are as follows:

- Current state-based host state detection without prediction (CS-HSD)
- Local regression-based host state detection (LR-HSD)
- LSTM-based host state detection (LSTM-HSD)

The performance comparison results between Autoformer-HSD and other methods are shown in Fig. 5. As can be seen from Fig. 5, Autoformer-HSD is significantly superior to other comparison methods in terms of the two core metrics—energy consumption and SLAV, verifying its role in improving VM consolidation performance.

a) Energy consumption: Compared with CS-HSD, the energy consumption of Autoformer-HSD is reduced by 30.8% (PlanetLab), 34.5% (Google), and 26.4% (Alibaba) across the three datasets. This is because CS-HSD only relies on the current load state to make decisions and cannot identify overload risks in advance. It often triggers migration only after the host is already overloaded, leading to redundant resource allocation and additional energy consumption from frequent migrations. In contrast, Autoformer-HSD can adjust resource allocation in advance through accurate prediction of future loads, reducing ineffective energy consumption at the source.

Compared with LR-HSD, the energy consumption of Autoformer-HSD is reduced by 25.6% (PlanetLab), 19.0% (Google), and 22.5% (Alibaba). LR-HSD can only capture local linear trends and has limited ability to model complex periodic fluctuations and long-term trends in cloud loads, resulting in high prediction errors that further affect the accuracy of host state judgment. However, the autocorrelation mechanism of Autoformer can effectively decompose the periodic and trend components of the load, improving prediction accuracy and reducing energy waste caused by misjudgment.

Compared with LSTM-HSD, the energy consumption of Autoformer-HSD is still reduced by 5.3% (PlanetLab), 7.1% (Google), and 7.7% (Alibaba). This benefit comes from the global perception mechanism of the Autoformer decoder, enabling it to more accurately capture the long-term dependencies of the load. This optimizes the timeliness and accuracy of host state detection, further reducing energy consumption.

b) SLAV: The SLAV of Autoformer-HSD is the lowest across all datasets. Compared with CS-HSD, its SLAV is reduced by 57.7% (PlanetLab), 61.9% (Google), and 36.8% (Alibaba). This is because CS-HSD lacks prediction capability and cannot avoid overload in advance, causing hosts to frequently be in a state of resource saturation and triggering SLA violations. In contrast, Autoformer-HSD can complete VM migration before the arrival of load peaks by predicting overload risks in advance, significantly reducing service interruption time.

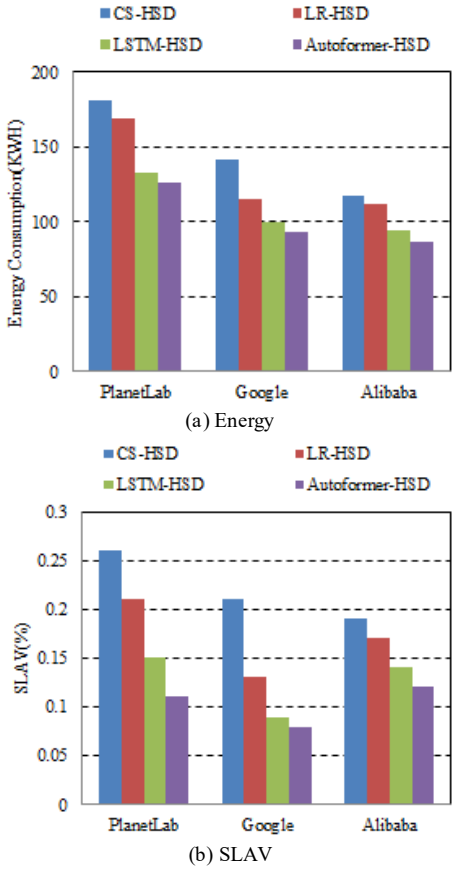


Fig. 5. Performance comparison between Autoformer-HSD and other host detection methods.

Compared with LR-HSD and LSTM-HSD, the SLAV advantage of Autoformer-HSD is slightly narrowed but still maintains a leading position. For example, on the Google dataset, its SLAV is 38.5% lower than that of LR-HSD and 11.1% lower than that of LSTM-HSD. This stems from Autoformer's accurate modeling of load fluctuations, which enables more precise judgment on whether a host will enter an overloaded state, reducing the increase in SLAV caused by prediction errors.

This ablation experiment shows that Autoformer-HSD can more precisely identify the overloaded and underloaded states of hosts by improving the accuracy of load prediction and the ability to model complex time-series features. It provides a reliable decision-making basis for subsequent VM selection and placement, ultimately reducing energy consumption while effectively ensuring service quality. This verifies its core role in the VM consolidation system.

2) Impact of VM selection on consolidation: To evaluate the impact of the PCM-VMS proposed in this paper on the overall performance of VM consolidation, we kept all other components of the proposed VM consolidation algorithm unchanged, while replacing PCM-VMS with other benchmark VM selection algorithms. In this way, we obtained performance data of VM consolidation based on different VM selection algorithms. The other benchmark VM selection algorithms used to verify the performance of PCM-VMS are as follows:

- Minimum migration time-based VM selection (MMT-VMS)
- Maximum correlation-based VM selection (MC-VMS)
- Influence coefficient-based VM selection (IC-VMS)

The performance comparison results between PCM-VMS and other VM selection algorithms are shown in Fig. 6. It can be indicated that the proposed PCM-VMS performs the best in terms of both energy consumption and SLAV metrics, verifying its role in improving VM consolidation performance.

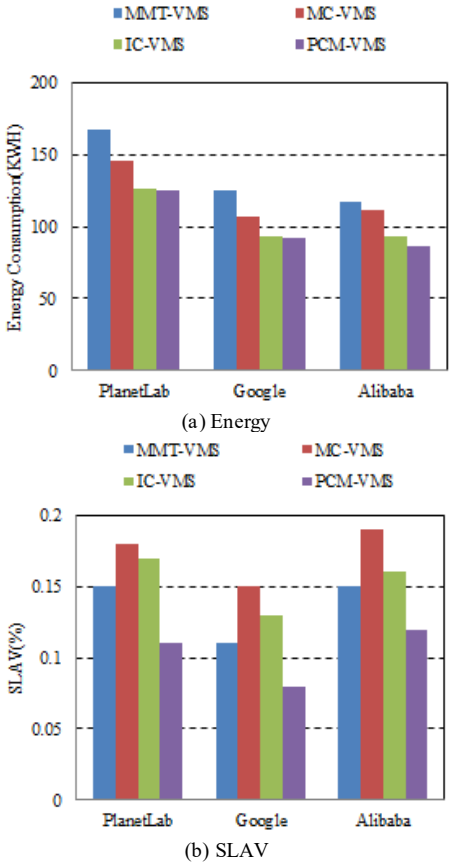


Fig. 6. Performance comparison between PCM-VMS and other VM selection algorithms.

a) Energy consumption: Compared with MMT-VMS which only considers migration time, the energy consumption of PCM-VMS is reduced by 25.0% (PlanetLab), 26.1%

(Google), and 26.4% (Alibaba) across the three datasets. Although MMT-VMS can reduce the time cost of a single migration, it fails to consider the resource correlation between VMs and hosts. This may lead to the migration of VMs that have little impact on host load, resulting in overloaded hosts not being effectively relieved of their load, and requiring multiple migrations to achieve load balancing, which instead increases overall energy consumption. In contrast, PCM-VMS balances correlation and migration cost, which enable overloaded hosts to return to a normal state through fewer migration operations, thereby reducing energy consumption.

Compared with MC-VMS which only focuses on correlation, the energy consumption of PCM-VMS is reduced by 14.1% (PlanetLab), 13.1% (Google), and 22.5% (Alibaba). Although MC-VMS can quickly alleviate host overload by migrating VMs with high correlation, it fails to consider migration time. This may lead to system performance loss due to excessive network resource occupation during migration, indirectly increasing energy consumption. In contrast, PCM-VMS penalizes the migration of VMs with large memory, reducing migration operations that incur high resource overhead and further optimizing energy consumption.

Compared with IC-VMS based on the influence coefficient, the energy consumption of PCM-VMS is reduced by 7.7% on the Alibaba dataset, and its performance is comparable to that of IC-VMS on the other datasets. This is because the influence coefficient of IC-VMS does not incorporate migration cost, whereas PCM-VMS precisely controls migration cost through a denominator term, making its decisions more aligned with the core goals of efficient load reduction and low overhead.

b) SLAV: The SLAV of PCM-VMS is the lowest across all datasets. Compared with MMT-VMS, its SLAV is reduced by 26.7% (PlanetLab), 27.3% (Google), and 20.0% (Alibaba), respectively. Since MMT-VMS ignores the correlation between VMs and hosts, it may cause host load rebound after migration, increasing SLAV; in contrast, the high-correlation VMs migrated by PCM-VMS can effectively reduce host load fluctuations and lower the risk of overload.

Compared with MC-VMS, the SLAV of PCM-VMS is reduced by 38.9% (PlanetLab), 46.7% (Google), and 36.8% (Alibaba). MC-VMS prioritizes migrating high-correlation VMs but may select VMs with large memory, leading to a decline in migration performance. In contrast, PCM-VMS restricts the migration of large-memory VMs through a denominator term, reducing performance loss during the migration process and thereby lowering the overall SLAV.

Compared with IC-VMS, the SLAV of PCM-VMS is reduced by 35.3% (PlanetLab), 38.5% (Google), and 25.0% (Alibaba). This benefit comes from PCM-VMS's refined modeling of correlation and migration cost, which ensures that migration operations can effectively alleviate host overload and control migration time. The collaborative optimization of these two aspects significantly reduces SLAV.

In summary, this ablation experiment verifies the effectiveness of PCM-VMS. By integrating the Pearson correlation coefficient and migration time, a more optimized influence factor is constructed. While reducing the number of migrations and lowering energy consumption, it effectively controls SLAV, providing an efficient decision-making basis for VM consolidation.

3) Impact of VM placement on consolidation: To evaluate the performance of the VM placement algorithm based on enhanced dual Q network (EDN-VMP) proposed in this paper, we kept all other components of the proposed VM consolidation algorithm unchanged, while replacing EDN-VMP with other benchmark VM placement algorithms. In this way, we obtained performance data of VM consolidation based on different VM placement algorithms. The other benchmark VM placement algorithms used to verify the performance of EDN-VMP are as follows:

- Minimum energy increment based VM placement (MEI-VMP)
- Q-Learning-based VM placement (QL-VMP)
- Dual-Q-network-based VM placement (DN-VMP)

The comparison results of VM consolidation performance between the EDN-VMP and other VM placement algorithms are shown in Fig. 7. It can be seen that the EDN-VMP is significantly superior to other benchmark algorithms in both energy consumption and service quality, fully verifying its role in improving VM consolidation performance.

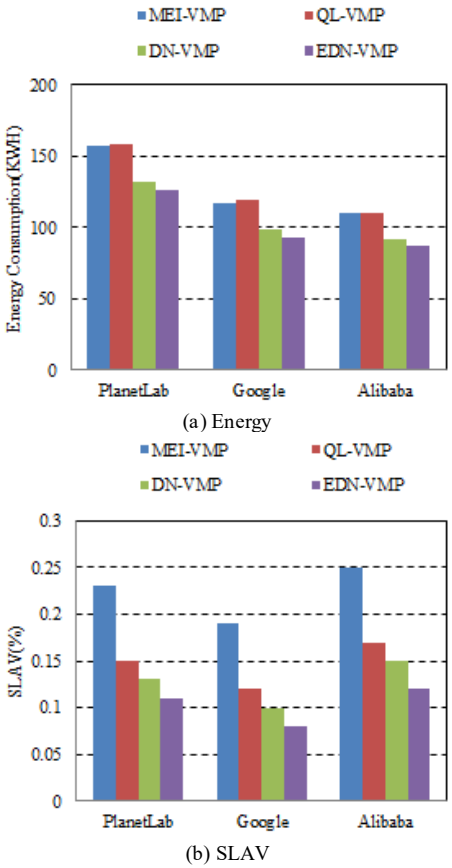


Fig. 7. Performance comparison between EDN-VMP and other VM placement algorithms.

a) Energy consumption: Compared with MEI-VMP, the energy consumption of EDN-VMP is reduced by 20.0% (PlanetLab), 21.1% (Google), and 21.1% (Alibaba) respectively. MEI-VMP adopts a greedy strategy and only takes the current energy consumption increment as the decision-making basis. It fails to consider long-term load trends and resource balance, which may lead to frequent migrations or host overload in the later stage due to short-term optimization, thereby increasing overall energy consumption. In contrast, EDN-VMP leverages the global decision-making capability of reinforcement learning and combines the UCB strategy to explore better placement schemes. While reducing current energy consumption, it avoids long-term resource imbalance, thus achieving more stable energy consumption optimization.

Compared with QL-VMP, the energy consumption of EDN-VMP is reduced by 20.6% (PlanetLab), 22.3% (Google), and 21.7% (Alibaba) respectively. QL-VMP has limited ability to model high-dimensional state spaces and struggles to accurately capture the correlation between placement decisions and long-term energy consumption, resulting in low decision-making accuracy. In contrast, EDN-VMP extracts complex state features through deep neural networks and combines a dual Q-network structure to avoid Q-value overestimation, significantly enhancing the global optimization capability of its decisions.

Compared with DN-VMP, the energy consumption of EDN-VMP is still reduced by 5.1% (PlanetLab), 6.1% (Google), and 5.8% (Alibaba). Although DN-VMP solves the overestimation problem, it lacks the key improvements of EDN-VMP: (1) It does not introduce the UCB strategy, resulting in insufficient exploration of high-quality placement schemes with low access frequency and a tendency to fall into local optimality; (2) It does not adopt an adaptive learning rate, leading to low efficiency in parameter updates for sparse scenarios. Through these two improvements, EDN-VMP further enhances the accuracy of resource allocation and reduces redundant energy consumption.

b) SLAV: Compared with MEI-VMP, the SLAV of EDN-VMP is reduced by 52.2% (PlanetLab), 57.9% (Google), and 52.0% (Alibaba), respectively. MEI-VMP, due to its over-pursuit of energy consumption optimization, may deploy VMs to hosts that are nearly saturated, leading to frequent host overload. Additionally, it lacks global migration planning, which may trigger unnecessary migration operations and result in decreased migration performance. In contrast, the reward function of EDN-VMP constrains both energy consumption and SLAV simultaneously. By balancing multi-objectives through reinforcement learning, it effectively reduces the risk of overload and migration losses.

Compared with QL-VMP, the SLAV of EDN-VMP is reduced by 26.7% (PlanetLab), 33.3% (Google), and 29.4% (Alibaba) respectively. QL-VMP's coarse state modeling leads to low matching accuracy between placement decisions and VM resource requirements, easily causing overload. However, EDN-VMP accurately captures state features via deep networks and explores better matching schemes by combining the UCB strategy, significantly reducing the SLA violation rate.

Compared with DN-VMP, the SLAV of EDN-VMP is reduced by 15.4% (PlanetLab), 20.0% (Google), and 20.0% (Alibaba) respectively. The performance gap arises because DN-VMP lacks an adaptive learning rate. For newly emerged state-action pairs, the fixed parameter update step size of DN-VMP makes it difficult to quickly learn optimal decisions, leading to a short-term increase in SLAV. In contrast, EDN-VMP assigns a larger update magnitude to sparse samples through its adaptive learning rate, accelerating adaptation to new scenarios and thereby reducing the violation rate.

In summary, this ablation experiment fully verifies the effectiveness of EDN-VMP. By introducing the UCB strategy to balance exploration and exploitation, adopting an adaptive learning rate to optimize parameter updates, and integrating the global decision-making capability of the dual Q-network, EDN-VMP significantly improves the stability of service quality while reducing energy consumption.

4) Impact of UCB and ALR on Dual Q-Network: To verify the impact of the UCB and ALR on the performance of the Dual Q-Network, we further conducted ablation experiments on the enhanced dual Q-Network. Based on the basic dual Q-Network, we integrated the UCB and adaptive learning rate, respectively. The several neural networks used for comparison are as follows:

- General dual Q-Network (DQN)
- Dual Q-Network with UCB (DQN+UCB)
- Dual Q-Network with ALR (DQN+ALR)
- Dual Q-Network with UCB and ALR (DQN+UCB+ ALR)

The impact of the UCB and ALR on the performance of the Dual Q-Network is shown in Table IV. The experimental results indicate that the introduction of UCB and ALR plays a significant role in improving the performance of the Dual Q-Network, and the synergistic effect of the two further optimizes the core metrics of VM consolidation.

TABLE IV. IMPACT OF UCB AND ALR ON PERFORMANCE OF DUAL Q-NETWORK

	Energy Consumption (KWH)			SLAV (%)		
	PlanetLab	Google	Alibaba	PlanetLab	Google	Alibaba
DQN	132.4	98.7	91.8	0.13	0.1	0.15
DQN+UCB	129.8	95.8	89.4	0.12	0.09	0.13
DQN+ALR	128.6	94.6	88.7	0.12	0.09	0.13
DQN+UCB+ALR	125.6	92.65	86.52	0.11	0.08	0.12

In terms of energy consumption, compared with the general dual Q-Network (DQN), the DQN+UCB achieves an energy consumption reduction of 1.96% (PlanetLab), 2.94% (Google), and 2.61% (Alibaba) across the three datasets. This benefit stems from the UCB strategy's ability to balance exploration and exploitation, which reduces resource allocation imbalance caused by local optimality and makes VM placement decisions more aligned with the goal of global energy consumption optimization. The DQN+ALR shows a more significant reduction in energy consumption. Its advantage lies in its fast learning capability for sparse state-action pairs, which accelerates the convergence of high-quality placement strategies and reduces redundant energy consumption.

The DQN+UCB+ALR performs the best, with energy consumption reduced by 5.14% (PlanetLab), 6.13% (Google), and 5.75% compared with DQN, reflecting the synergistic effect of the two mechanisms. UCB ensures sufficient exploration of the decision space, while ALR improves learning efficiency, enabling the algorithm to quickly find the balance between energy consumption and service quality in dynamic cloud environments.

In terms of SLAV, the SLAV of DQN+UCB is reduced by 7.69% (PlanetLab), 10.00% (Google), and 13.33% (Alibaba) compared with DQN. This indicates that the UCB strategy reduces the risk of host overload caused by decision limitations by exploring more potential high-quality placement schemes. The SLAV performance of DQN+ALR is comparable to that of DQN+UCB. This verifies the ALR's ability to quickly adapt to new scenarios, which can effectively reduce short-term service interruptions.

The SLAV advantage of DQN+UCB+ALR is further expanded. Compared with DQN, its SLAV is reduced by 15.38% (PlanetLab), 20.00% (Google), and 20.00% (Alibaba). The reason lies in that the UCB strategy reduces the proportion of suboptimal decisions caused by insufficient exploration, while the ALR accelerates the learning of high-risk state-action pairs. The combined effect of the two enables the algorithm to significantly reduce the SLAV rate.

In summary, the introduction of the UCB and ALR optimizes the dual Q-Network from two dimensions: the breadth of decision exploration and the depth of learning efficiency. Their combination achieves a "1+1>2" optimization effect through a synergistic effect, providing more accurate and efficient intelligent support for VM placement decisions. Ultimately, it significantly improves the stability of service quality while reducing energy consumption.

VIII. Conclusion and Future Work

To address the issues of energy optimization and service quality management in VM consolidation for cloud data centers, this paper proposes an adaptive consolidation strategy based on Autoformer and enhanced dual Q-Network. Through collaborative innovations in three stages: host state detection, VM selection, and VM placement, an effective balance between energy consumption and service quality in dynamic cloud environments is achieved.

In the host state detection stage, a prediction method based on Autoformer is adopted. This method performs multi-scale decomposition of load time series through an autocorrelation mechanism, explicitly separating trend components from periodic components. This significantly improves both load prediction accuracy and the accuracy of host state classification. In the VM selection stage, a VM migration impact factor that integrates the Pearson correlation coefficient and migration time is proposed. This factor balances resource correlation and migration overhead, effectively avoiding performance degradation caused by excessively high migration costs or insufficient load mitigation. In the VM placement stage, an enhanced dual Q-Network algorithm is designed. It introduces the UCB strategy to optimize the exploration-exploitation trade-off and adopts an ALR mechanism to accelerate model convergence, thereby achieving intelligent decision-making under multi-objective trade-offs.

Experimental results show that the strategy proposed in this paper significantly outperforms the comparative benchmark methods on three real load datasets, and can effectively reduce cloud data center energy consumption and SLAV rates. Ablation experiments further verify the effectiveness of each innovative module and their contributions to the overall performance.

Despite the good performance of the AEDQN-VMC strategy, there are still some limitations that can be further explored in future research:

- Adaptation to multi-resource dimensions and heterogeneous workloads: The current study mainly focuses on CPU and memory resources. In the future, it can be extended to multi-dimensional resource constraints such as network I/O and disk bandwidth, and the adaptability to heterogeneous workloads can be enhanced.
- Model lightweighting and online learning: The Autoformer and Dual Q-Network models have a large number of parameters, resulting in high training and inference overhead. Future research can explore model compression, knowledge distillation, and online incremental learning mechanisms to improve the practicality and real-time performance of the algorithm in ultra-large-scale data center environments.
- Cross-layer collaboration and cooling energy consumption optimization: The current energy consumption model only considers server energy consumption. In the future, a cooling system energy consumption model can be introduced, and cross-layer collaborative optimization between computing resource scheduling and cooling system control can be explored to further improve overall energy efficiency.
- Extension to multi-cloud and edge environments: This study focuses on a single data center scenario. In the future, the framework can be extended to multi-cloud collaboration or edge-cloud collaboration environments, and cross-domain resource scheduling and fault-tolerance mechanisms can be studied to enhance the generalization ability and application scope of the strategy.

ACKNOWLEDGMENT

This work was supported in part by Heilongjiang Provincial Natural Science Foundation Funded Project under Grant PL2024F003, and in part by Heilongjiang Provincial General Project of College Students' Innovation and Entrepreneurship Training Program under Grant S202510240008.

REFERENCES

[1] S. S. Gill, R. Buyya, "A taxonomy and future directions for sustainable cloud computing: 360 degree view." ACM Computing Surveys, vol. 51, no. 5, pp. 1-33, 2019.

[2] A. H. T. Dias, L. H. A. Correia, N. Malheiros, "A systematic literature review on virtual machine consolidation." ACM Computing Surveys, vol. 54, no. 8, pp. 1-38, 2021.

[3] J. Singh, N. K. Walia, "A comprehensive review of cloud computing virtual machine consolidation." IEEE Access, vol. 11, pp. 106190-106209, 2023.

[4] S. Goyal, L. K. Awasthi, "Adaptive multi-objective virtual machine consolidation for energy-efficient cloud data centers." Journal of Grid Computing, vol. 23, no. 2, article no. 21, 2025.

[5] J. Zeng, D. Ding, K. Kang, H. Xie, Q. Yin, "Adaptive drl-based virtual machine consolidation in energy-efficient cloud data center." IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 11, pp. 2991-3002, 2022.

[6] B. Feng, Z. Ding, "Application-oriented cloud workload prediction: A survey and new perspectives." Tsinghua Science and Technology, vol. 30, no. 1, pp. 34-54, 2025.

[7] F. Zhao, W. Lin, S. Lin, H. Zhong, K. Li, "Tfegru: Time-frequency enhanced gated recurrent unit with attention for cloud workload prediction." IEEE Transactions on Services Computing, vol. 18, no. 1, pp. 467-478, 2025.

[8] M. Zakarya, L. Gillam, M. R. C. Qazani, A. A. Khan, K. Salah, et al., "Backfillme: An energy and performance efficient virtual machine scheduler for iaas datacenters." IEEE Transactions on Services Computing, vol. 18, no. 2, pp. 660-672, 2025.

[9] N. Ma, A. Tang, Z. Xiong, F. Jiang, "A deep multi-agent reinforcement learning approach for the micro-service migration problem with affinity in the cloud." Expert Systems with Applications, vol. 273, article no. 126856, 2025.

[10] D. Zhao, J. t. Zhou, K. Li, "Cfws: Drl-based framework for energy cost and carbon footprint optimization in cloud data centers." IEEE Transactions on Sustainable Computing, vol. 10, no. 1, pp. 95-107, 2025.

[11] G. Long, S. Wang, C. Lv, "Qos-aware resource management in cloud computing based on fuzzy meta-heuristic method." Cluster Computing, vol. 28, no. 4, article no. 276, 2025.

[12] J. Li, Y. Deng, R. Wang, Y. Zhou, H. Feng, et al., "Btvmp: A burst-aware and thermal-efficient virtual machine placement approach for cloud data centers." IEEE Transactions on Services Computing, vol. 17, no. 5, pp. 2080-2094, 2024.

[13] X. Zhang, T. Wu, M. Chen, T. Wei, J. Zhou, et al., "Energy-aware virtual machine allocation for cloud with resource reservation." Journal of Systems and Software, vol. 147, pp. 147-161, 2019.

[14] J. Li, Y. Deng, Z. Zhong, Z. Wu, S. Pang, et al., "An energy-aware virtual machine scheduling approach for cloud data centers." IEEE Transactions on Sustainable Computing, vol. 10, no. 5, pp. 891-907, 2025.

[15] Y. Chen, Z. Zhang, Y. Deng, G. Min, L. Cui, "A combined trend virtual machine consolidation strategy for cloud data centers." IEEE Transactions on Computers, vol. 73, no. 9, pp. 2150-2164, 2024.

[16] W. Yao, Z. Wang, Y. Hou, X. Zhu, X. Li, et al., "An energy-efficient load balance strategy based on virtual machine consolidation in cloud environment." Future Generation Computer Systems, vol. 146, pp. 222-233, 2023.

[17] R. Shaw, E. Howley, E. Barrett, "Applying reinforcement learning towards automating energy efficient virtual machine consolidation in cloud data centers." Information Systems, vol. 107, article no. 101722, 2022.

[18] P. Qin, C. Guo, "Construction of virtualization adaptive algorithm for virtual machine environment," in Proc. of 2025 3rd International Conference on Integrated Circuits and Communication Systems (ICICACS), 2025, pp. 1-6.

[19] S. Vila, F. Guirado, J. L. Lérida, "Cloud computing virtual machine consolidation based on stock trading forecast techniques." Future Generation Computer Systems, vol. 145, pp. 321-336, 2023.

[20] C. Guo, L. Li, M. Zukerman, "Q-learning-based workload consolidation for data centers with composable architecture." IEEE Transactions on Industrial Informatics, vol. 21, no. 3, pp. 2324-2333, 2025.

[21] N. Moocheet, B. Jaumard, P. Thibault, L. Eleftheriadis, "Minimum-energy virtual machine placement using embedded sensors and machine learning." Future Generation Computer Systems, vol. 161, pp. 85-94, 2024.

[22] S. Rahmani, V. Khajehvand, M. Torabian, "Burstiness-aware virtual machine placement in cloud computing systems." The Journal of Supercomputing, vol. 76, no. 1, pp. 362-387, 2020.

[23] Z. Tong, J. Wang, Y. Wang, B. Liu, Q. Li, "Energy and performance-efficient dynamic consolidate vms using deep-q neural network." IEEE Transactions on Industrial Informatics, vol. 19, no. 11, pp. 11030-11040, 2023.

[24] N. K. Sharma, S. Bojjagani, Y. C. A. P. Reddy, M. Vivekanandan, J. Srinivasan, et al., "A novel energy efficient multi-dimensional virtual machines allocation and migration at the cloud data center." IEEE Access, vol. 11, pp. 107480-107495, 2023.

[25] J.-Y. Luo, L. Chen, W.-K. Chen, J.-H. Yuan, Y.-H. Dai, "A cut-and-solve algorithm for virtual machine consolidation problem." Future Generation Computer Systems, vol. 154, pp. 359-372, 2024.

[26] A. Ignatov, I. Maslova, M. Posypkin, W. Yang, J. Wu, "A two-phase heuristic algorithm for power-aware offline scheduling in iaas clouds." Journal of Parallel and Distributed Computing, vol. 178, pp. 1-10, 2023.

[27] A. R. Hummaida, N. W. Paton, R. Sakellariou, "Scalable virtual machine migration using reinforcement learning." Journal of Grid Computing, vol. 20, pp. 1-26, 2022.

[28] K. Haghshenas, A. Pahlevan, M. Zapater, S. Mohammadi, D. Atienza, "Magnetic: Multi-agent machine learning-based approach for energy efficient dynamic consolidation in data centers." IEEE Transactions on Services Computing, vol. 15, no. 1, pp. 30-44, 2022.

[29] H. Wu, J. Xu, J. Wang, M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting." Advances in neural information processing systems, vol. 34, pp. 22419-22430, 2021.

[30] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software: Practice and Experience, vol. 41, no. 1, pp. 23-50, 2011.

[31] A. Beloglazov, "Planetlab workload traces," GitHub, 2011. https://github.com/beloglazov/planetlab-workload-traces.

[32] Google, "Google cluster workload traces (version 2011)," GitHub, 2011. https://github.com/google/cluster-data/blob/master/ClusterData2011_2.md.

[33] Alibaba, "Alibaba cluster trace (version 2018)," GitHub, 2018. https://github.com/alibaba/clusterdata/tree/master/cluster-trace-v2018.

[34] K. D. Lange, "Identifying shades of green: The specpower benchmarks." Computer, vol. 42, no. 3, pp. 95-97, 2009.

[35] A. Beloglazov, R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers." Concurrency and Computation: Practice & Experience, vol. 24, no. 13, pp. 1397–1420, 2012.