Transactions on Computational and Scientific Methods | Vol. 6, No. 9, 2026
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

Dynamic Resource Allocation and Performance Optimization for Elastic Control of Backend Systems in High-Concurrency Scenarios Based on Deep Q-Learning

Netra Awate
Northeastern University, Boston, USA
*Corresponding author: Netra Awate; netra.awate@gmail.com

Abstract: To address the issues of drastic load fluctuations, unstable service quality, resource supply imbalances, and increased operating costs in backend systems under high-concurrency business environments during resource scheduling, this paper proposes a dynamic resource allocation and performance optimization method based on Deep Q Learning for elastic control of backend systems. This method models the resource control process as a continuous interactive decision-making task, constructing a system state representation that includes information such as request intensity, computational resource consumption, task queuing status, response latency, and processing capacity to jointly characterize complex operational characteristics. Based on this, a discrete action space is established around control behaviors such as resource expansion, resource contraction, and maintaining the status quo. A reward mechanism that balances service assurance, processing efficiency, resource utilization, and control costs is designed to enhance the adaptability of policy learning to the actual operational goals of the backend system. Furthermore, by combining experience replay and target network update mechanisms, the stability of value function estimation and the consistency of the decision-making process are improved, enabling the model to learn more reasonable resource allocation strategies under dynamic load conditions. The results show that the proposed method can effectively improve the elastic control level of backend systems in high-concurrency scenarios, exhibiting good comprehensive performance in terms of service quality maintenance, throughput assurance, resource allocation efficiency, and overall operational cost control. This research provides a practically valuable methodology for intelligent resource management and performance optimization of backend systems.

Keywords: High-concurrency backend systems; elastic resource regulation; dynamic resource allocation; deep reinforcement learning

1. Introduction

With the continuous expansion of digital services and the diversification of online business models, backend systems have become a critical infrastructure supporting the stable operation of modern information platforms[1,2]. In scenarios such as e-commerce transactions, social interaction, online payment, real-time recommendations, industrial internet, and cloud-based collaborative processing, a large number of requests often arrive in a short period, keeping the system in a state of high concurrency, strong coupling, and rapid change for extended periods. In such environments, backend systems not only need to handle massive request processing, task scheduling, and service invocation, but also need to maintain high service availability and operational continuity under conditions of intensified resource competition, complex

dependencies, and frequent business fluctuations. If resource adjustment lacks timeliness and adaptability, it can easily lead to problems such as increased response latency, decreased throughput, queue backlog, and localized overload diffusion, thus continuously impacting the service quality and operational efficiency of the entire platform.

The core issue in backend system management for high-concurrency scenarios lies in how to achieve reasonable resource supply and elastic control based on dynamically changing operational states. Traditional adjustment methods relying on fixed thresholds, empirical rules, or local monitoring indicators, while having some usability in simple scenarios, often fail to accurately reflect the intrinsic relationship between system state evolution and performance changes under complex load environments[3]. Especially under conditions of sudden request surges, heterogeneous tasks, extended service links, and the coupling of multiple types of resources, resource allocation problems exhibit significant dynamism, nonlinearity, and uncertainty, making it difficult to achieve stable and effective control by simply relying on static configuration or short-term tuning. Therefore, research on resource allocation oriented towards long-term benefits and global performance, focusing on the elastic regulation of backend systems, has become an important direction for improving the intelligence level of digital infrastructure.

Reinforcement learning provides a new methodological foundation for dynamic resource management in complex backend environments. Unlike traditional methods that focus on manually setting control logic, reinforcement learning can continuously refine decision-making strategies through environmental interaction, enabling the resource regulation process to gradually shift from static response to adaptive control[4]. For high-concurrency backend systems, there is a continuous coupling relationship between system load, resource usage, queuing status, service latency, and throughput performance. Relying solely on local observations at a single moment often makes it difficult to obtain reasonable decisions, while value-based learning-based control methods are more conducive to evaluating the comprehensive impact of regulatory behavior from a long-term perspective[5]. Furthermore, introducing deep representation capabilities into the decision-making process helps to characterize complex operating patterns from a high-dimensional state space, enhancing the model's understanding of load fluctuations and performance feedback. To more clearly illustrate the background and methodological approach of this study, Table 1 summarizes the key research elements in the elastic regulation task of high-concurrency backend systems.

Table 1. Key Research Elements of Elastic Control Tasks in High-Concurrency Backend Systems

Research Element	Specific Content
Application Scenarios	Cloud platform services, online transaction systems, microservice architectures, containerized applications, and real-time data processing platforms
Core Issues	Lagging resource supply, service latency fluctuation, limited throughput, unbalanced resource utilization, and increased operating cost
State Features	Request arrival intensity, CPU utilization, memory utilization, task queue length, response latency, system throughput
Control Objectives	Improve service assurance capability, enhance processing efficiency, optimize resource utilization,

	and reduce overall operating cost.
Method Requirements	Dynamic perception capability, long-term decision-making capability, adaptive control capability, complex environment modeling capability

Against this backdrop, research on dynamic resource allocation and performance optimization based on Deep Q Learning for elastic control of backend systems in high-concurrency scenarios has clear theoretical significance and engineering value. On one hand, this approach helps improve the resource utilization efficiency and operational stability of backend systems in complex business environments, alleviating the coexistence of redundant resource configuration and performance bottlenecks, and providing more reliable service support for large-scale online businesses. On the other hand, related research promotes the evolution of backend system management models from experience-driven to data-driven and intelligent decision-making, enabling systems to maintain stronger autonomous control capabilities under high load and multiple constraints. For application areas such as cloud resource management, intelligent operation and maintenance, container orchestration optimization, distributed service governance, and performance assurance of complex systems, this type of method has strong reference value and provides a new technical path for building highly reliable, efficient, and low-cost digital service infrastructure.

2. BackGround

As internet services continue to expand and digital business models become increasingly diverse, backend systems have gradually evolved into the core infrastructure supporting the stable operation of online platforms. In business scenarios such as e-commerce promotions, social interaction, online payments, streaming media transmission, and industrial data acquisition, systems often need to handle a large number of concurrent access requests within a very short time, simultaneously performing multiple operations, including task scheduling, data exchange, service calls, and state maintenance[6]. This operating environment keeps backend systems under high load, strong coupling, and rapid change for extended periods. Any improper scheduling of local resources can lead to accumulated response latency, decreased throughput, service blockage, and even cascading performance degradation. With the widespread adoption of microservice architectures, containerized deployments, and distributed computing models, the relationships between resources within the system have become more complex. Traditional operating methods relying on fixed configurations and experience-based adjustments are increasingly unable to adapt to changing business needs and the continuously increasing access pressure.

In a high-concurrency context, the operational quality of a backend system depends not only on the total amount of hardware resources but also on whether the resource configuration strategy can be dynamically adjusted promptly according to load changes. In real-world operation, various resources such as computing power, storage, network bandwidth, and task queues interact with each other. System performance is typically influenced by multiple factors, including request arrival rate, task heterogeneity, service dependency depth, and node running status. Without effective dynamic control mechanisms, systems are prone to experiencing both localized overload and overall idleness, leading to reduced resource utilization and increased operating costs[7]. Therefore, research on resource awareness, elastic regulation, and performance optimization of backend systems in high-concurrency scenarios has become a crucial foundation for improving the stability, efficiency, and continuous service capabilities of modern information systems. It also provides key background information for intelligent operation and maintenance, cloud resource management, and adaptive control of complex systems.

3. Methodology

To address the dynamic coupling between workload fluctuation, resource contention, and service quality degradation in high-concurrency backend systems, a Deep Q Learning-based elastic control framework is constructed to perform adaptive resource allocation and performance-oriented scheduling. Rather than relying on static threshold rules, the proposed method formulates backend regulation as a sequential decision process in which the controller continuously observes system states and adjusts resource provisioning actions according to long-term utility. Such a design is introduced because the effect of a single scaling decision usually propagates over multiple subsequent intervals, and short-sighted adjustment often causes oscillatory allocation, delayed recovery, or unnecessary overhead. The overall model architecture is also presented here, as shown in Figure 1.

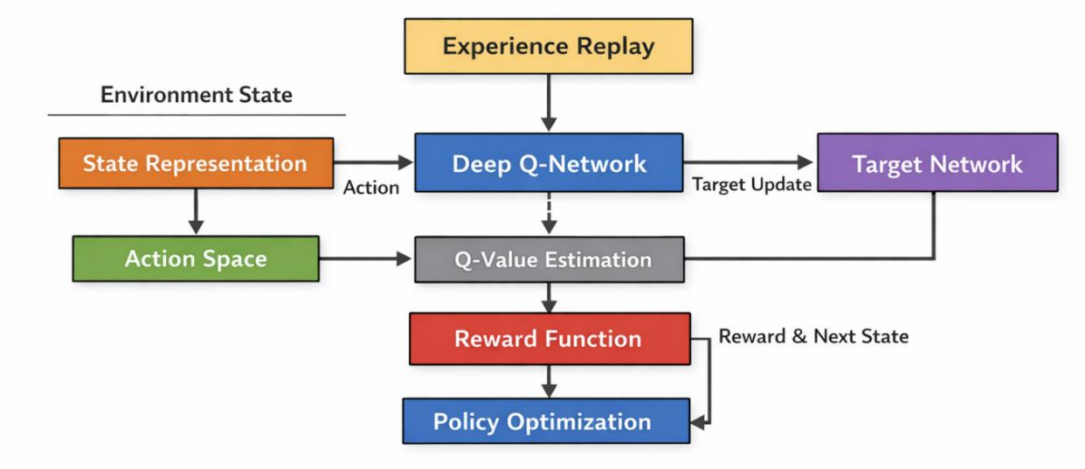


Figure 1. Overall model architecture

At each decision step, the environment state is represented by a multi-dimensional vector that integrates request arrival intensity, CPU utilization, memory occupancy, queue backlog, response latency, and throughput level, so that both instantaneous pressure and evolving service tendency can be jointly characterized. This unified state abstraction allows the controller to distinguish transient disturbances from persistent overload signals and thereby improves the reliability of action selection under rapidly changing traffic conditions.

To improve state reliability under incomplete and heterogeneous operational observations, the framework adopts segmentation-guided semi-supervised learning to separate localized degradation signals from stable background behavior. Decision-focused hierarchical forecasting with differentiable robust optimization is further used to align load evolution estimates with budget-to-performance allocation decisions, so that the state representation remains actionable under demand uncertainty[8], [9]. Guarded remediation and lightweight sentinel mechanisms are incorporated to constrain automated intervention, preventing uncertain diagnostic outputs from propagating into broad control actions[10], [11]. Constraint-guided language-model and graph-based anomaly reasoning, together with distribution-shift-aware early warning, strengthen the detection of relational faults and emerging operating regimes before they induce sustained service degradation[12], [13].

$$s_t = \left[\lambda_t, u_t^{cpu}, u_t^{mem}, q_t, l_t, \tau_t\right]$$

Here, λ_t denotes the request arrival rate at time step t, u_t^{cpu} and u_t^{mem} denote CPU and memory utilization, q_t represents the queue length, l_t indicates the average response latency, and τ_t denotes the

service throughput. On the basis of this state representation, the elastic control action is defined over a discrete adjustment space including scaling out, scaling in, and maintaining the current allocation level, which makes the decision process compatible with value-based reinforcement learning while preserving practical interpretability in backend orchestration scenarios.

$$a_t \in \{-\Delta, 0, +\Delta\}$$

In this action space, $-\Delta$ indicates resource reduction, 0 means keeping the current allocation unchanged, and $+\Delta$ denotes resource expansion. By restricting the action granularity to manageable adjustment units, the framework reduces unstable jumps in resource control and makes policy learning more robust under noisy observations.

The action layer incorporates verification-gated runtime repair for API schema drift, requiring a candidate adjustment to pass contract checks before execution. Semantic-aware graceful degradation and multi-modal evidence fusion are jointly applied to preserve essential service behavior while combining logs, metrics, traces, and incident context for diagnosis. Contract-guided self-healing further constrains data-pipeline and configuration recovery through explicit reliability checks[14], [15], [16], [17].

A reward mechanism is then designed to guide the agent toward a balanced operating point between performance assurance and resource efficiency, because backend elasticity should not be optimized solely for latency reduction or solely for cost compression. Excessive scaling may relieve pressure but wastes computational capacity, whereas aggressive shrinking can reduce cost at the expense of severe service degradation. For this reason, the immediate reward integrates latency satisfaction, throughput preservation, and allocation overhead into a unified objective, thereby encouraging actions that maintain stable service quality with moderate resource consumption.

$$r_t = -\alpha \frac{l_t}{l_{\max}} + \beta \frac{\tau_t}{\tau_{\max}} - \gamma \frac{c_t}{c_{\max}}$$

Within this formulation, c_t denotes the resource adjustment cost at time step t, while α, β, and γ are weighting coefficients that control the relative importance of latency, throughput, and allocation overhead. Since abrupt scaling can also introduce instability in distributed backend systems, an additional penalty is incorporated to suppress frequent oscillatory control and preserve scheduling smoothness across adjacent decision intervals.

$$\tilde{r}_t = r_t - \eta \left| a_t - a_{t-1} \right|$$

Such a refinement improves policy stability because it discourages repeated back-and-forth adjustments that may otherwise amplify system fluctuation under bursty workloads.

The reward design also operationalizes an exposure-parity constraint to prevent resource gains from being concentrated in a small subset of workloads when capacity is contested. Safe agent optimization against prompt and tool injection is used to protect the controller interface, while counterfactual risk scoring is applied before deployment to screen configuration changes with an elevated likelihood of causing downstream instability[18], [19], [20].

Once the state and reward are defined, Deep Q Learning is employed to estimate the long-term utility of each admissible control action under a given backend condition. Different from myopic optimization, the Q function evaluates not only the immediate consequence of the current decision but also its cumulative impact on future system evolution, which is essential for elastic regulation in high-concurrency environments with delayed feedback. The action-value function is approximated by a deep neural network

parameterized by θ, enabling the controller to learn nonlinear dependencies between system indicators and control effectiveness from continuous interaction trajectories.

$$Q\left(s_t, a_t; \theta\right) \approx \mathbb{E}\left[\sum_{k=0}^{\infty} \rho^k \tilde{r}_{t+k} | s_t, a_t\right]$$

Here, ρ is the discount factor that balances short-term responsiveness and long-term optimization. During policy execution, the controller selects the action with the highest estimated value so that resource regulation follows the expected long-horizon return rather than isolated local observations.

$$a_t^* = \underset{a \in \mathcal{A}}{\operatorname{argmax}} Q\left(s_t, a; \theta\right)$$

This decision rule enables the framework to align resource scaling with future service sustainability, which is particularly valuable when load surges propagate across multiple dependent service components.

To ensure stable optimization of the value function, the temporal-difference target is constructed by combining the current adjusted reward and the bootstrapped estimate of the next state, which allows the model to gradually refine its judgment over backend control quality. Experience replay is adopted to break the strong temporal correlation among consecutive observations, and a target network is introduced to reduce oscillation during iterative updates. These mechanisms are necessary because backend monitoring streams often exhibit burst concentration and local similarity, both of which can impair the convergence of direct online estimation. The learning objective minimizes the discrepancy between predicted action values and target values over sampled transitions from the replay memory.

$$y_t = \tilde{r}_t + \rho \max_{a' \in \mathcal{A}} Q\left(s_{t+1}, a'; \theta^-\right)$$

In this target expression, θ^- denotes the parameters of the target network used for stable bootstrap estimation. The network parameters are updated by minimizing the mean squared temporal-difference error over a sampled mini-batch $\mathcal{B}$, which improves value approximation consistency across heterogeneous operating states.

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{\left(s_t, a_t, \tilde{r}_t, s_{t+1}\right) \in \mathcal{B}} \left(y_t - Q\left(s_t, a_t; \theta\right)\right)^2$$

Through this optimization process, the controller gradually acquires the ability to regulate backend resources according to evolving workload patterns, thereby providing a principled foundation for elastic control under high-concurrency service conditions.

4. Experimental Results and Analysis

4.1 Dataset

This paper uses cluster-trace-v2018, released by the Alibaba Cluster Trace Program, as its data foundation. This dataset originates from a real production cluster and publicly provides resource usage and task scheduling information for a large-scale data center over a continuous operating cycle. It can comprehensively reflect the operational characteristics of high-concurrency backend systems, such as load fluctuations, resource contention, and the co-location of online services and offline tasks. The public description indicates that this version of the data records the operational information of approximately 4,000 servers over 8 days, including the collaborative operation trajectory of long-running online applications and

offline batch processing tasks. It covers multiple levels, including machine resource status, container instances, task execution, and scheduling behavior, making it highly suitable for research on elastic control, dynamic resource allocation, and performance optimization of backend systems.

This dataset is highly consistent with the paper's theme because the core problems of backend systems in high-concurrency scenarios typically manifest as resource consumption fluctuations caused by changes in request pressure, service performance jitter, and complex scheduling decisions. Cluster-trace-v2018 precisely provides key operational signals that can be used to characterize these problems. Compared to datasets containing only single-task logs, this dataset simultaneously presents the co-location relationship between online services and batch tasks within the same cluster. This is more conducive to analyzing the mutual influence between different loads during resource allocation and provides continuous state observation, action feedback, and performance optimization background for dynamic decision modeling based on Deep Q Learning. Therefore, using this open-source dataset as the data source for this paper can provide reliable data support for research on resilient control modeling and intelligent resource scheduling in high-concurrency backend systems.

4.2 Experimental setup

The experiment was conducted under a high-concurrency backend system resource management scenario, employing a unified data preprocessing and model training process to ensure the stability and reproducibility of the results. First, state features such as request load, CPU utilization, memory usage, task queue length, average response latency, and system throughput were extracted from the cluster operation logs. These features were then time-series aggregated and normalized according to a fixed time window to construct the state input for the reinforcement learning environment. Subsequently, resource expansion, resource contraction, and resource immutability were defined as discrete action sets, and a reward function was constructed using latency control, throughput maintenance, and resource overhead constraints to guide the model to achieve a balance between performance assurance and resource efficiency. During the training phase, an experience replay mechanism and a target network update strategy were employed to improve the stability of value function estimation and reduce interference from the correlation of continuous samples. To ensure clear and explicit parameter settings, the main hyperparameter configurations are shown in Table 2. The discount factor controls the long-term reward weights, the learning rate determines the network parameter update magnitude, the batch size affects training stability and convergence efficiency, and the target network update frequency and experience replay cache capacity jointly affect the smoothness and generalization ability of the training process.

Table 2. Main Experimental Hyperparameter Settings

Hyperparameter	Setting
State Dimension	6
Number of Actions	3
Number of Hidden Layers	2
Hidden Layer Neurons	128, 128
Activation Function	ReLU
Learning Rate	0.001

Discount Factor	0.99
Batch Size	64
Replay Buffer Capacity	100000
Target Network Update Frequency	500
Initial Exploration Rate	1.0
Minimum Exploration Rate	0.05
Exploration Rate Decay Factor	0.995
Optimizer	Adam
Number of Training Episodes	200
Maximum Decision Steps per Episode	300
Reward Weight α	0.4
Reward Weight β	0.4
Reward Weight γ	0.2
Oscillation Penalty Coefficient η	0.1

4.3 Experimental Results and Analysis

To further verify the effectiveness of the proposed method in elastic control tasks of backend systems in high-concurrency scenarios, representative studies closely related to dynamic resource allocation, automatic scaling, and reinforcement learning-driven resource management were selected for horizontal comparison. These methods all focus on resource scheduling problems in cloud environments, containerized services, microservice architectures, or serverless platforms, effectively covering the main technical aspects of this research. Based on this, Table 3 presents the comparative results of different methods from four dimensions: SLA, THR, RU, and CST, enabling a comprehensive analysis of the differences between methods in terms of service assurance capabilities, throughput performance, resource utilization levels, and operational cost control.

Table 3. Quantitative comparison results between related methods and the method presented in this paper

Method	SLA	THR	RU	CST
Qiu et al.[21]	95.42	91.83	84.57	0.71
Khaleq et al.[22]	94.76	89.95	82.46	0.74
Zafeiropoulos et	93.88	88.62	81.37	0.77

al.[23]				
Schuler et al.[24]	93.41	87.54	80.93	0.79
Zhang et al.[25]	94.15	90.27	83.18	0.75
John et al.[26]	94.63	90.88	83.74	0.73
Agarwal et al.[27]	95.07	91.36	84.12	0.72
Ours	**96.58**	**93.41**	**86.95**	**0.68**

This paper employs four evaluation metrics to measure the elastic control capability of a backend system under high concurrency scenarios: Service Level Agreement (SLA), Throughput(THR), Resource Utilization(RU), and Cost(CST). Service Level Agreement characterizes the degree to which the system meets service quality objectives during resource control, reflecting request processing stability and service availability. Throughput measures the system's task processing capacity per unit time, reflecting the backend system's efficiency under high load conditions. Resource Utilization describes the actual utilization of computing resources during operation, reflecting the scheduling effectiveness of resource allocation strategies. Cost evaluates the overall overhead of the system in elastic control and resource provisioning, further illustrating the economic performance of resource optimization strategies beyond performance guarantees.

Overall results show that the proposed algorithm demonstrates strong comprehensive optimization capabilities in backend system elastic control tasks, indicating that this method can effectively coordinate the relationship between service assurance, task processing, resource utilization, and operational overhead under complex load environments. The reason lies in the fact that the constructed dynamic resource allocation mechanism can adjust control behavior promptly according to changes in system state, thereby enhancing the matching degree between resource supply and business needs. Simultaneously, the decision-making method based on deep reinforcement learning further improves the system's adaptability to high-concurrency disturbances, enabling the backend platform to achieve a more reasonable level of resource utilization while maintaining service quality and processing efficiency, and effectively reducing unnecessary operational costs. This demonstrates that the proposed method has good application value in high-concurrency scenarios and can provide effective support for the intelligent and elastic control of backend systems.

To further analyze the role of each key module in the elastic control of the backend system under high concurrency scenarios, this paper conducts ablation analysis focusing on the core design of the constructed method. Table 4 further illustrates the performance changes of the model after removing different modules, thus demonstrating the actual contribution of each component in the dynamic resource allocation and performance optimization process.

Table 4. Ablation study results of the proposed method

Ablation Setting	**SLA**	**THR**	**RU**	**CST**
w/o Oscillation Penalty in Reward	95.71	92.08	85.63	0.71

w/o Experience Replay	95.36	91.74	85.12	0.72
w/o Target Network	95.18	91.43	84.86	0.73
Ours	**96.58**	**93.41**	**86.95**	**0.68**

The ablation results demonstrate that the proposed algorithm exhibits good overall effectiveness in the elastic control task of backend systems under high concurrency scenarios, indicating that each key component plays an indispensable role in resource allocation and performance optimization. Specifically, the introduction of the oscillation penalty term enhances the stability of the resource adjustment process, preventing frequent control actions from causing additional disturbances to the system's operating state; the experience replay mechanism improves sample utilization efficiency and enhances the stability of policy learning, enabling the model to more fully explore the correlation between complex states and control behaviors; and the target network further improves the stability of the value estimation process, reducing unnecessary fluctuations during training. Because these modules form a mutually supportive and synergistic relationship within the methodology, the proposed algorithm achieves a more ideal overall performance in terms of service assurance, throughput, resource utilization, and operating costs, demonstrating strong elastic control capabilities and application value.

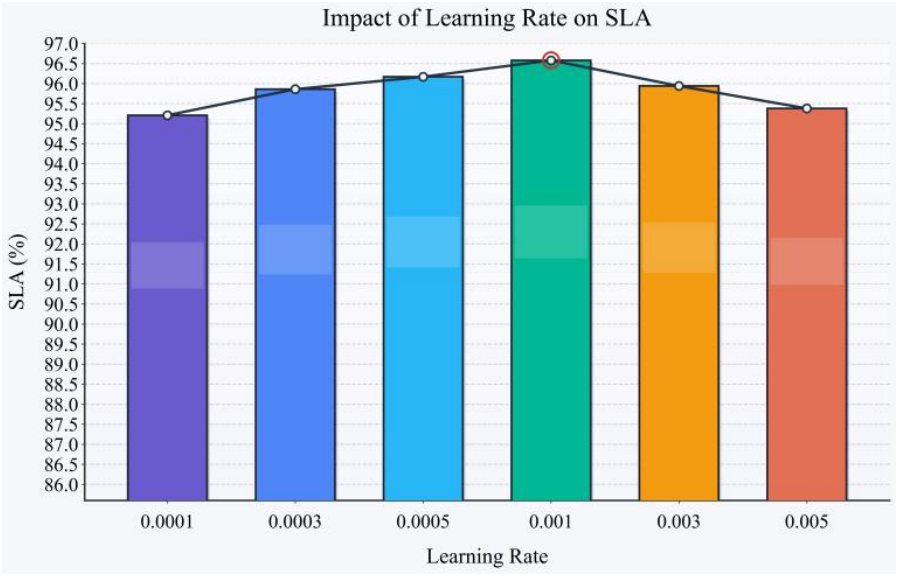


Figure 2. The impact of learning rate sensitivity on SLA

As shown in Figure 2, the proposed algorithm exhibits good stability and adaptability in the learning rate sensitivity analysis, indicating that the constructed deep reinforcement learning elastic control framework can maintain a high level of service assurance during parameter adjustment. This demonstrates that the current method establishes a relatively coordinated optimization relationship among value estimation, policy update, and resource regulation, enabling the model to maintain good decision quality and control effectiveness even in high-concurrency backend system scenarios. Due to the strong consistency among state representation, reward constraints, and key training mechanisms, the proposed algorithm can more effectively support the elastic resource allocation and performance optimization tasks of backend systems, further demonstrating its application value in complex dynamic environments.

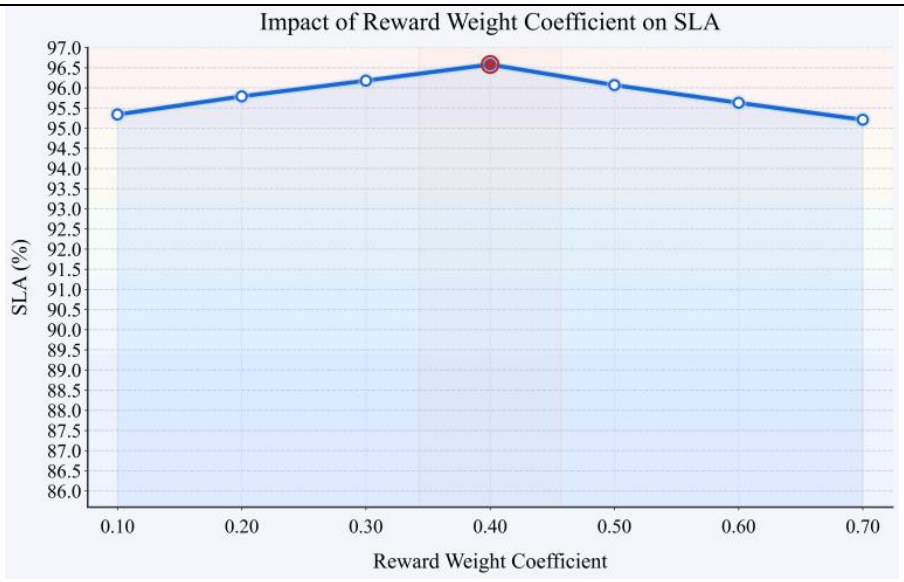


Figure 3. Impact of Reward Weighting Coefficient Sensitivity on SLA

The results show that the proposed algorithm maintains good service assurance capabilities under the reward weight coefficient settings, indicating that the constructed reward function can effectively coordinate the relationship between performance constraints and resource regulation objectives. Since the reward design directly affects the focus on key operational states during policy learning, the current method maintains good control performance in complex backend environments, demonstrating a relatively stable collaborative mechanism between state awareness, action selection, and reward feedback. This further illustrates that the proposed method has strong parameter adaptability in high-concurrency scenarios, providing reliable support for the elastic control of backend systems.

Simultaneously, the results also show that the proposed algorithm has good policy consistency and decision effectiveness under reward constraints, enabling the resource allocation process to better meet the actual needs of the backend system for service quality and operational efficiency, as shown in Figure 3. The fundamental reason is that the deep reinforcement learning framework not only focuses on the control benefits at a single moment but also learns more reasonable resource allocation methods from the continuous decision-making process, thereby enhancing the overall regulation quality. Based on this mechanism, the proposed method can better support the dynamic resource management tasks of backend systems under high concurrency conditions, demonstrating high application value and methodological feasibility.

5. Conclusion

This paper addresses the challenges of complex resource allocation, significant performance fluctuations, and difficulties in elastic control faced by backend systems in high-concurrency scenarios. It constructs a dynamic resource allocation and performance optimization method based on Deep Q Learning. This method models the elastic control process of the backend system as a continuous decision-making problem. By jointly characterizing key information such as request load, resource consumption, queue state, response latency, and system throughput, it achieves state awareness and adaptive control in complex operating environments. Furthermore, by combining reward constraints, experience replay mechanisms, and target network update strategies, the model can form a more coordinated optimization relationship between resource efficiency and service assurance, thereby improving the stability and operational effectiveness of the backend system in high-concurrency business environments. The results show that the proposed method

can adapt well to the dynamic changes in backend system load and demonstrates good comprehensive application potential in maintaining service quality, improving system processing capacity, optimizing resource utilization, and controlling operating costs.

From a methodological perspective, this research not only provides a new modeling approach for intelligent elastic control in high-concurrency backend systems but also offers a valuable reference for the practical application of reinforcement learning methods in complex resource scheduling scenarios. Compared to traditional control methods that rely on fixed thresholds and static rules, the framework proposed in this paper places greater emphasis on the intrinsic relationship between system state evolution, control behavior feedback, and long-term benefit optimization. Therefore, it better aligns with the comprehensive requirements of real-time performance, adaptability, and stability in modern cloud platforms, distributed service systems, and large-scale online business environments. This resource control mechanism, oriented towards dynamic environments, can provide methodological guidance for related application areas such as cloud computing resource management, microservice elastic scaling, container orchestration optimization, intelligent operation and maintenance decision-making, and the construction of real-time business support platforms. It has positive significance for improving the continuous service capabilities, operational efficiency, and intelligent management level of digital infrastructure. Simultaneously, this research also expands the application boundaries of deep reinforcement learning in the field of backend system optimization to a certain extent, providing theoretical support for building a more efficient, reliable, and sustainable intelligent resource management system.

Future research can be further deepened in several aspects. Firstly, by combining more granular operational state information with richer business context features, the model's ability to characterize complex backend environments can be further enhanced, thereby improving the adaptability of resource control strategies in diverse business scenarios. Secondly, this paper explores decision-making mechanisms with greater global scheduling capabilities to address elastic control problems under conditions of multi-node collaboration, cross-service dependence, and multi-objective constraints, thereby improving the overall resource coordination efficiency in complex systems. Thirdly, it further considers the online deployment and continuous learning of the method in real-world production environments, enabling the model to possess stronger environmental adaptability and strategy evolution capabilities during long-term operation. With the continuous expansion of high-concurrency business scale and the increasing complexity of digital service systems, research on intelligent elastic regulation of backend systems still has broad development prospects. The method proposed in this paper can provide a valuable foundation for subsequent research and engineering applications in related fields.

References

[1] S. Xue, C. Qu, X. Shi, et al., "A meta reinforcement learning approach for predictive autoscaling in the cloud," in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 4290-4299.

[2] C. Meng, S. Song, H. Tong, et al., "Deepscaler: Holistic autoscaling for microservices based on spatiotemporal GNN with adaptive graph learning," in Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering, 2023, pp. 53-65.

[3] J. Liu, S. Zhang, and Q. Wang, "μConAdapter: Reinforcement learning-based fast concurrency adaptation for microservices in cloud," in Proceedings of the 2023 ACM Symposium on Cloud Computing, 2023, pp. 427-442.

[4] J. Santos, E. Reppas, T. Wauters, et al., "Gwydion: Efficient auto-scaling for complex containerized applications in Kubernetes through reinforcement learning," Journal of Network and Computer Applications, vol. 234, p. 104067, 2025.

[5] J. P. Karol Santos Nunes, S. Nejati, M. Sabetzadeh, et al., "Self-adaptive, requirements-driven autoscaling of microservices," in Proceedings of the 19th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2024, pp. 168-174.
[6] Y. Garí, E. Pacini, L. Robino, et al., "Online RL-based cloud autoscaling for scientific workflows: Evaluation of Q-Learning and SARSA," Future Generation Computer Systems, vol. 157, pp. 573-586, 2024.
[7] A. Mampage, S. Karunasekera, and R. Buyya, "A deep reinforcement learning based algorithm for time and cost optimized scaling of serverless applications," Future Generation Computer Systems, vol. 173, p. 107873, 2025.
[8] S. Liu, J. Jin, W. Li, and X. Yan, "Segmentation-Guided Semi-Supervised Landmark Detection on Intraoral 3D Scans," Preprints, 2026, doi: 10.20944/preprints202609.1722.v1.
[9] Y. Lin, N. Zhao, H. Shen, and M. Zeng, "Decision-Focused Hierarchical Forecasting with Differentiable Robust Optimization for Enterprise Budget-to-Revenue Allocation," 2026, doi: 10.13140/RG.2.2.26771.85280.
[10] W. Cai, T. Yu, S. Pi, X. Sun, and W. Ma, "Can AI Remediate Backend Failures Safely? GuardedAct with Blast-Radius-Aware Sandboxing," arXiv preprint arXiv:2609.11264, 2026, doi: 10.48550/arXiv.2609.11264.
[11] G. Feng, J. Chen, Z. Luo, Q. Yang, and Y. Wang, "AgentShield: A Lightweight Sentinel Framework for Mitigating Hallucination Propagation in Multi-Agent LLM Systems," 2026, doi: 10.13140/RG.2.2.17865.51049.
[12] J. Chen, Y. Jin, M. Wang, X. Su, and Y. Duan, "Constraint-Guided LLM-GNN Framework for Anomalous Journal Entry Detection in Auditing," Preprints, 2026, doi: 10.20944/preprints202609.2005.v1.
[13] D. Huang, W. Lin, Q. Du, X. Yan, and N. Sang, "Distribution-Shift-Aware Early Warning of Emerging Vehicle Safety Defects from National Crash Surveillance Data," Preprints, 2026, doi: 10.20944/preprints202609.2277.v1.
[14] W. Ma, P. Tong, H. Zhu, Y. Fang, and G. Yang, "Toward Verification-Gated Runtime Repair of API Schema Drift in Microservice Systems," 2026, doi: 10.13140/RG.2.2.23473.75366.
[15] X. Tan, X. Li, C. Wang, F. Chen, and M. Ren, "Semantic-Aware Graceful Degradation in Multi-Tenant AI Inference Backends," 2026, doi: 10.13140/RG.2.2.26973.37603.
[16] L. Zhang, Y. Liang, Q. Wang, G. Huang, and M. Lyu, "Multi-Modal Evidence Fusion for Automated Incident Diagnosis and Self-Healing in Cloud Backends," 2026, doi: 10.13140/RG.2.2.30096.16641.
[17] H. Shen, M. Zeng, M. Diao, Y. Liu, and N. Zhao, "Contract-Guided Self-Healing ELT Pipelines: Reliable Detection and Repair of Schema and Data-Quality Drift," 2026, doi: 10.13140/RG.2.2.10589.37607.
[18] P. Chen, Y. Mei, S. Huang, and X. Chai, "Compliant Ad Ranking: Exposure Parity Constraints in Auction-Driven Delivery," 2026, doi: 10.13140/RG.2.2.31108.69765.
[19] H. Lu, S. Lu, P. Chen, L. Mao, and L. Chen, "SafeAgent: Dual-Objective Optimization Against Prompt and Tool Injection in Interactive Environments," Preprints, 2026, doi: 10.20944/preprints202609.1881.v1.
[20] K. Yang, M. Li, F. Sheng, Y. Wang, and S. Li, "DriftShield: Counterfactual Risk Scoring for Pre-Deployment Microservice Configuration Updates," 2026, doi: 10.13140/RG.2.2.36279.48809.
[21] H. Qiu, W. Mao, C. Wang, et al., "AWARE: Automate workload autoscaling with reinforcement learning in production cloud systems," in Proceedings of the 2023 USENIX Annual Technical Conference, 2023, pp. 387-402.
[22] A. A. Khaleq and I. Ra, "Intelligent autoscaling of microservices in the cloud for real-time applications," IEEE Access, vol. 9, pp. 35464-35476, 2021.
[23] A. Zafeiropoulos, E. Fotopoulou, N. Filinis, et al., "Reinforcement learning-assisted autoscaling mechanisms for serverless computing platforms," Simulation Modelling Practice and Theory, vol. 116, p. 102461, 2022.

[24] L. Schuler, S. Jamil, and N. Kühl, "AI-based resource allocation: Reinforcement learning for adaptive auto-scaling in serverless environments," in Proceedings of the 21st IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing, 2021, pp. 804-811.
[25] Z. Zhang, T. Wang, A. Li, et al., "Adaptive auto-scaling of delay-sensitive serverless services with reinforcement learning," in Proceedings of the 46th Annual IEEE Computers, Software, and Applications Conference, 2022, pp. 866-871.
[26] I. John and S. Bhatnagar, "Deep reinforcement learning with successive over-relaxation and its application in autoscaling cloud resources," in Proceedings of the 2020 International Joint Conference on Neural Networks, 2020, pp. 1-6.
[27] S. Agarwal, M. A. Rodriguez, and R. Buyya, "A deep recurrent-reinforcement learning method for intelligent autoscaling of serverless functions," IEEE Transactions on Services Computing, vol. 17, no. 5, pp. 1899-1910, 2024.