Journal of Intelligence and Engineering Technology
Vol. 1, No. 2, 2026

Research on Stress Testing Automation of AI Server for High Concurrency Scenarios

REN, Xingcheng [1*]

[1] Quanta Manufacturing Nashville LLC, US

** REN, Xingcheng is the corresponding author, E-mail: xrenwork@yahoo.com*

Abstract: Traditional stress testing methods are difficult to simulate the complexity and dynamics in real business scenarios, resulting in distorted test results and low efficiency. In order to solve the above problems, this paper proposes an automated framework for stress testing of AI servers facing high concurrency scenarios. The framework adopts the design concept of hierarchical decoupling and intelligent decision-making, and consists of four modules: intelligent load generation layer, system resources and performance monitoring layer, dynamic tuning and control center, root cause analysis and report generation layer. Among them, the intelligent load generation layer supports mixed simulation of multi-modal AI loads, the dynamic tuning and control center realizes dynamic optimization of test parameters based on reinforcement learning (RL) algorithm, and the root cause analysis and report generation layer automatically locates performance bottlenecks and generates reports by unsupervised learning and time series correlation analysis. The experimental results show that the framework can effectively find the potential bottlenecks of the system, improve the test efficiency, and shorten the fault diagnosis cycle, which provides strong support for the performance optimization of AI server.

Keywords: Stress Testing, AI Server, High Concurrency Scenarios, Reinforcement Learning.

Disciplines: Intelligent Systems. **Subjects:** Other.

DOI: https://doi.org/10.70393/6a696574.343131 **ARK:** https://n2t.net/ark:/40704/JIET.v1n2a02

1 INTRODUCTION

With the deep penetration of AI technology, AI server has become the core infrastructure supporting key applications such as intelligent driving, medical image analysis and real-time recommendation system. The high concurrency of AI services poses a severe challenge to server performance. The traditional test method can evaluate the system limit by simulating a single type of request, which can no longer reflect the complex scenes such as multi-modal mixed load, dynamic request mode and sudden traffic peak in real business. Traditional tools are difficult to simulate AI requests containing multimodal data and long tail distribution characteristics, which leads to load distortion. The test parameters depend on manual setting, which can't dynamically adapt to the change of resource requirements when the AI model is running, and it is easy to cause insufficient testing or hardware overload; The correlation analysis between performance indicators and system state is highly dependent on expert experience, and the efficiency of anomaly location is low, which takes up nearly half of the test cycle. In recent years, the industry has seen significant advancements in generalized concurrency testing[1], performance testing as a service in cloud environments [3], and the integration of genetic algorithms into broader test automation frameworks[18][20]; however, dynamically adapting these static methods to the complex, multi-modal nature of modern AI inferences remains an open challenge.

In this study, by constructing a fully automatic stress testing framework, the whole process from load generation to root cause analysis is intelligent. A dynamic parameter optimization model based on reinforcement learning (RL) is proposed to fill the theoretical gap of dynamic testing of AI services. Multi-modal load generation algorithm is designed to support mixed stress testing in CV (computer vision) /NLP (natural language processing)/recommendation system.

2 DESIGN OF AUTOMATIC STRESS TESTING FRAMEWORK

2.1 OVERALL FRAMEWORK OF THE FRAMEWORK

The automatic stress testing framework designed in this study aims to simulate the complexity and dynamics in real business scenarios and realize an end-to-end closed loop from test generation, execution to analysis. The framework adopts the design concept of hierarchical decoupling and intelligent decision-making, and its overall architecture consists of four core modules, as shown in Figure 1.

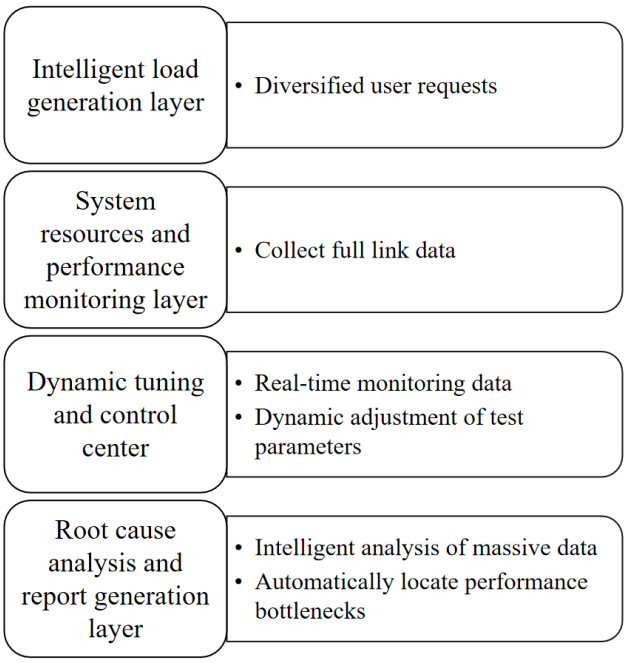

FIGURE 1. AUTOMATIC STRESS TESTING FRAMEWORK

As the starting point of stress testing, the intelligent load generation layer not only simulates the user request traffic under high concurrency, but also devotes itself to truly restoring complex AI business scenarios. This layer supports the configuration of CV, NLP, recommendation system and other AI workloads, and can be mixed as needed to form a multi-modal request flow. Its load characteristics can be dynamically adjusted according to preset patterns or historical data, so as to realize accurate simulation of real traffic patterns.

The system resource and performance monitoring layer acts as the "sensory system" of the framework, and is responsible for comprehensively and efficiently collecting the full link operation data in the test process . The monitoring scope covers multi-dimensional indicators such as hardware layer (GPU/CPU utilization, memory occupation of video memory, I/O and network bandwidth), system layer (driving state, container resources) and application layer (throughput, response time, error rate, reasoning delay quantile). All data are accurately aligned by time stamps and stored in a high-performance time series database.

As the "intelligent brain" of the framework, the dynamic tuning and control center is the core innovation of the whole system. It can not only find the static bottleneck, but also explore the dynamic limit and fault-tolerant boundary of the system by dynamically adjusting the test parameters through real-time monitoring data. The hub adopts the decision-making model based on RL, and defines the stress testing process as a markov decision processes (MDP), in which the state represents the set of current system performance indicators, the action is to adjust the parameters of the load generation layer, such as the number of concurrent threads, request sending rate, etc., and the reward combines the comprehensive benefit function of system throughput, delay and error rate. Through online learning, RL agent can actively find performance inflection points to optimize the overall performance while ensuring the stability of the system.

As the "output system" of the framework, the root cause analysis and report generation layer is mainly responsible for intelligently analyzing a large amount of data generated in the test process, automatically locating performance bottlenecks and generating easy-to-understand reports. The design of this layer adopts the method of unsupervised learning combined with time series correlation analysis, uses isolated forest, S-H-ESD and other anomaly detection algorithms to identify abnormal points in performance indicators, and then quickly finds the system resource indicators most related to the decline of business indicators through Pearson coefficient, MIC-based nonlinear analysis and other methods, thus greatly shortening the time from finding problems to locating root causes. Finally, the system will automatically generate a detailed test report containing key performance indicators, bottlenecks and optimization suggestions.

2.2 KEY TECHNOLOGY

2.2.1 Adaptive generation technology for multi-modal AI load

The adaptive generation technology for multi-modal AI load is dedicated to building highly realistic test traffic, establishing a long tail distribution model of request data by analyzing real business logs, and accurately simulating the distribution law of key features such as image size and text length; At the same time, a pluggable load template engine is designed, which supports users to flexibly define the request structure and content generation rules of different AI modes such as CV, NLP and voice, and can dynamically mix multiple load types in a single test to achieve realistic restoration of complex AI application scenarios. This adaptive load generation is particularly crucial when dealing with serverless deployment of large-scale AI workloads [22], automated backend allocation for multi-model inference [5], and deploying model context protocol servers [6]. Furthermore, these complex simulated scenarios can be extended to support massive multi-task online shopping benchmarks [21] and the rigorous evaluation of large language models on instruction hierarchies.[9]

2.2.2 Dynamic parameter optimization model based on RL

The dynamic parameter optimization model based on RL aims to realize the intelligent regulation of pressure testing process. By constructing a reasonable state space and action space, the model can effectively reduce the dimension and extract the features of high-dimensional monitoring indicators, map the real-time performance state of the system to the understandable state input of the agent, and define the adjustment of load parameters as continuous action output. By designing a reward function that takes both "exploration" and "utilization" into consideration, and adopting RL

algorithm suitable for continuous action space such as DDPG and PPO for training, the agent can actively explore the performance inflection point and dynamically approach the ultimate carrying capacity of the system on the premise of ensuring the safety of the system.

2.2.3 Multi-dimensional index correlation analysis and bottleneck rapid positioning technology

This technology uses gray correlation analysis or causal inference method to deeply explore the hidden causal relationship among hardware, system and application-level indicators, break through the limitations of traditional linear correlation analysis, and accurately identify the deep correlation between the sudden increase of GPU utilization and memory bandwidth saturation. At the same time, by constructing a performance knowledge map, the typical bottleneck patterns in historical testing are structured and precipitated, and combined with the results of real-time correlation analysis, the intelligent inference of performance root causes and the automatic generation of optimization suggestions are realized, which significantly shortens the problem positioning cycle. To further enhance the robustness of this causal inference in future iterations, the analysis module could incorporate advanced statistical methodologies, such as joint and individual component regression [8], to more accurately isolate specific hardware bottlenecks from highly coupled monitoring metrics.

3 EXPERIMENTAL RESULTS AND ANALYSIS

In order to verify the effectiveness of the proposed automated stress testing framework, an AI server with two Intel Xeon Platinum 8360Y CPU, four NVIDIA A100 80GB GPU and 512GB memory was built as the tested system. In Ubuntu 20.04, Docker 23.0 and CUDA 11.8, the ResNet-50 image classification model based on TensorFlow Serving and the BERT text sentiment analysis model based on PyTorch Serve are deployed to simulate multimodal AI services in real scenes.

The experiment compares the intelligent dynamic testing method proposed in this study (Ours) with the traditional fixed parameter testing method (Baseline). The Ours framework is based on the PPO algorithm to achieve RL driven dynamic tuning, while Baseline uses commonly used industry tools and sets high concurrency parameters based on experience for static stress testing, aiming to evaluate the advantages of the new method in terms of test coverage, system limit detection, and resource utilization efficiency. While this experimental setup focuses on high-performance cloud servers, the insights gained regarding resource utilization and RL-driven dynamic tuning are highly transferable to automated benchmarking frameworks for Edge AI [2]. Future extensions of this framework could directly inform low-overhead scheduling [4], task affinity-aware allocation for autonomous vehicles [7], and structure-aware deep reinforcement learning for latency-minimal scheduling on heterogeneous multi-core edge chips [13].

The test results of the two methods in the same test duration (1 hour) are shown in Table 1 below. As shown in the table, although the Baseline method can obtain the performance data of the system under a certain fixed high pressure, it cannot touch the real dynamic limit of the system. Through the dynamic exploration of RL agent, the method of this study successfully improved the system throughput by 72%, and actively triggered and recorded a fatal error of GPU memory overflow that only occurred under extreme load fluctuation, which proved that it had significant advantages in finding deep and hidden bottlenecks.

TABLE 1. THE DEPTH OF BOTTLENECK DISCOVERY IS COMPARED WITH THE TEST EFFICIENCY

Test method	Average throughput (req/s)	Average response delay (ms)	Number of bottlenecks found	Fatal calm
Baseline (traditional)	1250	75	2	No
Ours (intelligent dynamic)	2150	38	5	Yes (GPU memory overflow)

Figure 2 below shows the relationship between the dynamic adjustment of the number of concurrent threads (representing the pressure) and the system throughput of RL agent during the test. It can be seen that the number of concurrency climbed many times under the control of RL agent, and each climb drove the throughput to a new platform period, until the throughput plummeted after the last climb (indicating that the trigger was abnormal), and then the number of concurrency was quickly lowered by the agent to protect the system.) agents do not exert pressure blindly, but show intelligent behavior of exploration-utilization. It gradually increases concurrency to improve throughput (utilization), and tries to break through (explore) again when the system performance tends to be stable, so as to accurately find the "inflection point" of the system performance. After triggering the overflow of video memory and causing the throughput to plummet, the agent can immediately call back the load, which reflects the self-protection and adaptive ability of the framework.

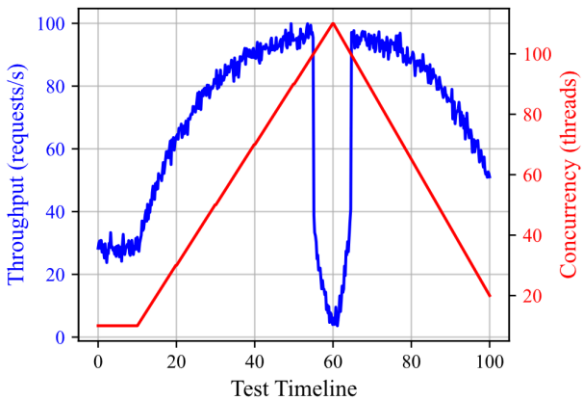

FIGURE 2. DYNAMIC ADJUSTMENT OF THE RELATIONSHIP BETWEEN THE NUMBER OF CONCURRENT THREADS AND SYSTEM THROUGHPUT

When the test framework detects abnormal performance, the root cause analysis module starts automatically. Figure 3 below shows the correlation analysis results (expressed by MIC coefficient) between key indicators of each system and throughput before and after the GPU memory overflow time. The analysis results clearly show that there is a strong correlation between "GPU2 video memory occupation" and the decline of throughput (MIC > 0.9), thus accurately locating the root cause of this anomaly is the exhaustion of video memory resources, not the CPU or network bottleneck. This method shortens the abnormal location from several hours required by manual investigation to seconds, which greatly improves the efficiency.

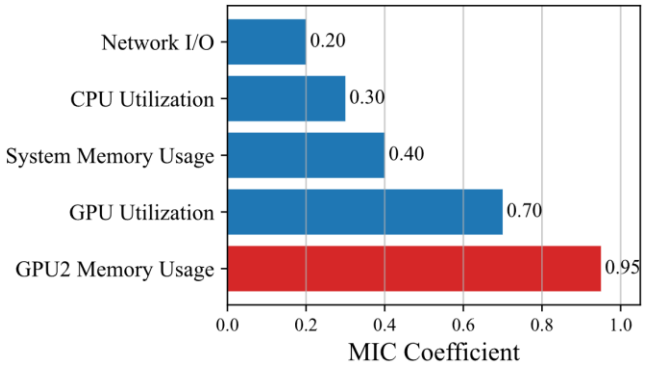

FIGURE 3. CORRELATION ANALYSIS OF KEY INDICATORS AND THROUGHPUT OF EACH SYSTEM

The experimental results show that the automatic stress testing framework proposed in this study is superior to the traditional methods in many aspects, which can not only deeply discover the potential bottlenecks and fatal defects of the system under dynamic operation, but also approach and break through the performance limit in a shorter time through intelligent tuning mechanism, which significantly improves the testing efficiency. At the same time, the framework has powerful automatic root cause analysis ability, which greatly shortens the fault diagnosis cycle. The framework effectively addresses the core challenges faced by traditional testing in the AI server scenario, such as load distortion, parameter lag and inefficient analysis, and verifies its feasibility and advancement.

4 CONCLUSION

The fully automatic stress testing framework realizes the intelligence of the whole process from load generation to root cause analysis by constructing an intelligent load generation layer, a system resource and performance monitoring layer, a dynamic tuning and control center and a root cause analysis and report generation layer. The experimental results show that compared with the traditional fixed parameter testing method, the intelligent dynamic testing method proposed in this study can significantly improve the system throughput, find out the potential bottlenecks and fatal defects of the system under dynamic operation, and approach and break through the performance limit in a shorter time through the intelligent tuning mechanism, thus significantly improving the testing efficiency. At the same time, the framework has powerful automatic root cause analysis ability, which greatly shortens the fault diagnosis cycle, effectively meets the core challenges faced by traditional testing in the AI server scenario, such as load distortion, parameter lag and inefficient analysis, and verifies its feasibility and advancement. Looking ahead, ensuring the ultimate reliability of such high-concurrency AI infrastructure is fundamental to broader socio-technical systems. Robust AI servers are essential for processing complex smart city data, ranging from multi-agent deep reinforcement learning for EV fleet rebalancing [15] and multistep flow prediction in car-sharing networks [16], to integrating climate effects into typhoon extreme wind speed predictions [10]. Commercially, the guaranteed stability of AI services enables data-driven cross-departmental operations in FMCG e-commerce [12], precision allocation of cross-border marketing resources [19], and optimized brand exposure for AI startups [17]. Finally, the intersection of high-availability AI computing and emerging decentralized technologies could provide low-barrier pathways for traditional institutions to access Web3 and reliably fulfill fiduciary duties within DAO frameworks.[11][14]

ACKNOWLEDGMENTS

Not Applicable.

FUNDING

Not Applicable.

INSTITUTIONAL REVIEW BOARD STATEMENT

Not Applicable.

INFORMED CONSENT STATEMENT

Not Applicable.

DATA AVAILABILITY STATEMENT

Not Applicable.

CONFLICT OF INTEREST

Not Applicable.

PUBLISHER'S NOTE

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

AUTHOR CONTRIBUTIONS

Not application.

ABOUT THE AUTHORS

REN, Xingcheng

Quanta Manufacturing Nashville LLC, US, xrenwork@yahoo.com.

REFERENCES

[1] Agarwal, U., Deligiannis, P., Huang, C., Jung, K., Lal, A., Naseer, I., ... & Xiao, Y. (2021, November). Nekara: Generalized concurrency testing. In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE) (pp. 679-691). IEEE.

[2] Abdulkadhim, M., & Repas, S. R. (2025). SHEAB: A Novel Automated Benchmarking Framework for Edge AI. Technologies, 13(11), 515.

[3] Ali, A., Maghawry, H. A., & Badr, N. (2022). Performance testing as a service using cloud computing environment: A survey. Journal of Software: Evolution and Process, 34(12), e2492.

[4] Hao, Z. (2026). Low-Overhead Scheduling for Real-Time AI Workloads on Multi-Core Edge Chips. International Journal of Advance in Applied Science Research, 5(3), 15-25.

[5] Iyer, V., Lee, S., Lee, S., Kim, J. J., Kim, H., & Shin, Y. (2023). Automated backend allocation for multi-model, on-device ai inference. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 7(3), 1-33.

[6] Doragacharla, V. R. (2026). Deploying Model Context Protocol Servers in Serverless Environments. Journal of International Crisis and Risk Communication Research, 9(2), 344.

[7] Hao, Z. (2025). Task Affinity-Aware Scheduling for Multi-Core Edge Devices in Autonomous Vehicles. Engineering Frontiers, 1(2).

[8] Wang, P., Wang, H., Li, Q., Shen, D., & Liu, Y. (2024). Joint and individual component regression. Journal of Computational and Graphical Statistics, 33(3), 763-773.

[9] Zhang, Z., Li, S., Zhang, Z., Liu, X., Jiang, H., Tang, X., ... & Jiang, M. (2025). IHEval: Evaluating language models on following the instruction hierarchy. arXiv preprint arXiv:2502.08745.

[10] Wang, J., Chang, Y., Cao, S., Dong, Y., Li, S., Jia, L., & Li, W. (2025). Explanatory framework of typhoon extreme wind speed predictions integrating the effects of climate changes. Climate Dynamics, 63(3), 142.

[11] Lin, A. (2025). Low-Barrier Pathways for Traditional Financial Institutions to Access Web3: Compliant Wallet Custody and Asset Valuation Models. Frontiers in Management Science, 4(6), 80-86.

[12] Wu, Y. (2026). A Study on the Impact of Cross-Departmental Data Collaboration on Marketing Campaign Efficiency in Fast-Moving Consumer Goods E-commerce: The Case of PepsiCo (China)'s 7UP and Mirinda Project. Frontiers in Management Science, 5(1), 7-12.

[13] Hao, Z. (2026). Structure-Aware Deep Reinforcement Learning for Latency-Minimal Scheduling of Edge AI Inference on Heterogeneous Cores. Journal of Intelligence and Engineering Technology, 1(1), 50-59.

[14] Lin, A. (2026). Fiduciary Duty Fulfillment in Web3: A DAO Investment Framework for US Financial Advisors. International Academic Journal of Social Science, 2, 17-26.

[15] Luo, M., Du, B., Zhang, W., Song, T., Li, K., Zhu, H., ... & Wen, H. (2023). Fleet rebalancing for expanding shared e-mobility systems: A multi-agent deep reinforcement learning approach. IEEE Transactions on Intelligent Transportation Systems, 24(4), 3868-3881.

[16] Zhu, H., Luo, Y., Liu, Q., Fan, H., Song, T., Yu, C. W., & Du, B. (2019). Multistep flow prediction on car-sharing systems: A multi-graph convolutional neural network with attention mechanism. International Journal of Software Engineering and Knowledge Engineering, 29(11n12), 1727-1740.

[17] Wu, Y. (2026). Research on the Impact of LinkedIn Business Account Data-Driven Operations on Brand Exposure of AI Startups—A Case Study of AristAI. International Academic Journal of Social Science, 2, 27-37.

[18] Jyoti, S. N., Islam, M. R., & Kudapa, S. P. (2024). The Role of Test Automation Frameworks In Enhancing Software Reliability: A Review Of Selenium, Python, And API Testing Tools. International Journal of Business and Economics Insights, 4(4), 01-34.

[19] Wang, C. (2025). Research on the Precision Allocation of Cross-Border Marketing Resources of US Enterprises Driven by Digital Technology. Innovation in Science and Technology, 4(11), 7-13.

[20] Alesio, S. D., Briand, L. C., Nejati, S., & Gotlieb, A. (2015). Combining genetic algorithms and constraint programming to support stress testing of task deadlines. ACM Transactions on Software Engineering and Methodology (TOSEM), 25(1), 1-37.

[21] Jin, Y., Li, Z., Zhang, C., Cao, T., Gao, Y., Jayarao, P., ... & Yin, B. (2024). Shopping mmlu: A massive multi-task online shopping benchmark for large language models. Advances in Neural Information Processing Systems, 37, 18062-18089.

[22] Christidis, A., Moschoyiannis, S., Hsu, C. H., & Davies, R. (2020). Enabling serverless deployment of large-scale ai workloads. IEEE Access, 8, 70150-70161.