Auto-PIP: Real-time Identification of Critical Performance Inflection Points in Software Stress Testing

Shenglin Zhang*, Xiao Xiong*, Mengyao Li*, Yongqian Sun*, Yongxin Zhao*
Xia Chen†, Bowen Deng†, Dan Pei‡
* *Nankai University*, † *Huawei Cloud*, ‡ *Tsinghua University*

***Abstract*—Conducting stress testing is essential to ensure the stability and performance of software systems post-launch. Among its various aspects, the real-time identification of the "Smooth Load Area" (SLA) is crucial for establishing performance benchmarks, identifying bottlenecks, guiding optimization strategies, and strategically allocating resources. However, existing methods are incapable of real-time inflection point detection or require manual intervention. This paper proposes *Auto-PIP*, an automated identification framework for performance inflection points based on key performance indicators (KPIs) during performance stress tests. *Auto-PIP* integrates trend testing algorithms with unsupervised anomaly detection algorithms to identify optimal operating points and estimate maximum capacity. *Auto-PIP* has been deployed in *Huawei Cloud*, and the evaluation results show that *Auto-PIP* demonstrated 100% accuracy in optimal inflection point detection, a 41.7% leap over baselines, and 83.9% accuracy in maximum point identification, a 10.7% gain. Additionally, we have released the dataset to the public to promote ongoing research.**

***Index Terms*—stress testing, performance status identification, performance inflection point, SLA**

I. Introduction

Stress testing is a crucial component of an organization's process before deploying a new version of a software system, to ensure that the system can withstand the anticipated load and user demand [1], [2]. It verifies system robustness by simulating a high-load environment and identifying load-related issues that may affect performance [3], [4].

Among various stress testing analysis methodologies, the real-time identification of the "Smooth Load Area (SLA)" is particularly crucial [5], as shown in Fig. 1. This phase represents the optimal operational range where the system maintains high performance and stability as the load increases incrementally. Identifying the SLA is essential for setting performance benchmarks, uncovering potential bottlenecks, assessing system resilience, guiding optimization strategies, and planning resource allocation effectively [6]. Recognizing SLA is fundamentally about automated identification of both the optimal and maximum inflection points (see section 2 for more details). Note that, operators do not actually want to push the software system to collapse during the stress test since repairing the test environment afterward would be costly. Therefore, the real-time on-line identification of critical performance inflection points is essential for making prompt decisions.

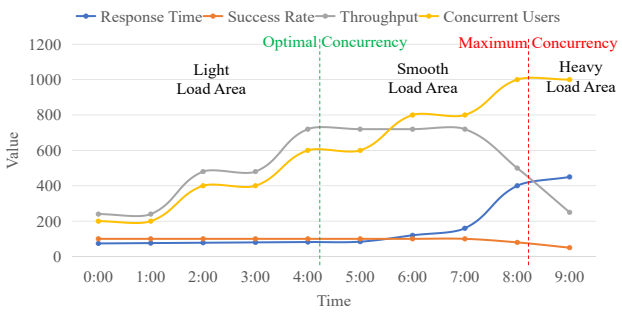

Fig. 1. The performance turning point model in a software stress testing.

Research on identifying the maximum and optimal inflection points remains relatively scarce. The existing approach [7] for identifying the maximum inflection point uses a nonlinear least squares method to fit a quadratic function representing the trend of throughput as the number of concurrent users changes. The extreme value of this quadratic function is considered the maximum inflection point. This approach assumes a direct relationship between throughput and the number of concurrent users, which can be more complex in actual applications. More importantly, it is an offline algorithm that identifies the maximum inflection point only after post-processing key performance indicators (KPIs) following stress tests. This delay can lead to system crashes during stress testing, increasing the subsequent recovery costs.

To overcome the above limitations, we plan to automatically and in real-time determine the system's optimal and maximum inflection points based on changes in KPIs during stress testing. The key ideas of this method are: 1) Identifying the optimal inflection point when throughput stops increasing with the number of concurrent users, transforming this identification into a trend detection of throughput; 2) Identifying the maximum inflection point when response time begins to surge, transforming this identification into an anomaly detection problem for response time. However, in the planned exploration, we face the following challenges:

(1) Insufficient quality of KPIs. Inaccuracies or errors

during the collection and recording of KPIs can introduce data noise and missing values. Some methods are susceptible to the effects of outliers or missing KPI data, increasing the risk of drawing incorrect conclusions [8]. Additionally, some methods require KPIs to follow specific data distributions [9]. If the actual distribution of KPIs does not match the assumed distribution, the reliability of the test results may be compromised.

(2) Insufficient quantity of KPIs. Advanced anomaly detection methods based on KPIs mostly use deep neural networks, which require extensive training data to achieve the desired performance. However, the duration of individual software system stress tests is typically short, and the resulting KPIs may lack the periodicity and regularity required to effectively train deep learning models.

(3) Selection of KPI thresholds. Testers often need to manually set the threshold for each KPI according to traditional methods [10], which is time-consuming, labor-intensive, and prone to errors. A more automated and dynamic approach for generating KPI thresholds would be a more practical solution.

In this paper, we propose a robust performance status identification method for software systems, namely *Auto-PIP*, which can identify the optimal and maximum inflection points during stress testing in real-time, accurately, and efficiently. The main contributions of this paper are as follows:

1) To robustly handle data noise and unknown data distributions, *Auto-PIP* involves Mann-Kendall test to identify the optimal inflection points, which is insensitive to noise and suitable for various data distributions, addressing the first challenge. Additionally, *Auto-PIP* uses the statistically-based SPOT to identify the maximum inflection points, which dynamically determine KPI thresholds with only a small number of initial data points, overcoming the the second and third challenges.
2) To evaluate the performance of *Auto-PIP*, we collect data from the industrial environment of *Huawei Cloud*. Our results show that the accuracy for identifying the optimal and maximum inflection points reaches 100% and 83.9%, respectively, outperforming existing baseline methods. Additionally, to facilitate further research, we have also released our labeled dataset [1].
3) *Auto-PIP* has been deployed in the industrial environment of *Huawei Cloud* for one month, significantly improving the efficiency of test engineers in analyzing the performance of the software systems. To better understand the effectiveness of *Auto-PIP*, we analyze three real-world stress testing cases.

II. Background and related works

A. Performance Inflection Point Model

Performance Inflection Point Model (PIPM) is an essential tool for evaluating the performance of software systems [5]. As shown in Fig. 1, PIPM demonstrates how system KPIs, such as throughput, response time, and success rate, change as the number of concurrent users increases. This model is crucial for determining software systems' optimal and maximum inflection points. As the number of concurrent users rises, the service performance of the software system transitions from high to low. Within the PIPM, this performance change can be categorized into three distinct stages [5]:

- Light load area: When the system handles a small number of concurrent users, the system's processing capacity (throughput) exhibits the fastest growth trend. At this stage, the system can quickly respond to user operations and meet various user requests fully and promptly.
- Smooth load area (SLA): As the number of concurrent users reaches a moderate level, the growth of system throughput stagnates compared to the low concurrency stage. Nevertheless, the system can still process user requests at a reasonable speed, keeping response waiting times within an acceptable range for users. Recognizing SLA is undamentally identification of optimal and maximum inflection points.
- Heavy load area: As the number of concurrent users increases, system resources reach the maximum and remain unchanged, and system throughput begins to drop sharply. Concurrently, due to resource constraints, the response time for user requests significantly increases, with some requests potentially taking a long time to be processed sor failing entirely, leading to a decline in user experience.

The focus of this paper is to identify the optimal number of concurrent users (i.e., the optimal inflection point) that maximizes system performance, situated between the light load area and the smooth load area. Additionally, the paper aims to identify the maximum number of concurrent users (i.e., the maximum inflection point) near the fault state of the system, situated between the smooth load area and the heavy load area. Identifying the optimal inflection point helps reduce the waste of computing resources while ensuring user experience. Conversely, identifying the maximum inflection point is crucial for operators to ascertain system capacity and efficiently implement traffic limiting, service degradation and other service protection measures.

B. Related Work

KPI trend detection. Statistical methods are commonly used for KPI trend detection, often accompanied by confidence intervals or p-values to quantify their reliability [11]. The slope method [12] measures trend direction by calculating changes in the slope of KPIs over time, identifying trends when slope changes exceed a threshold. However, it assumes linearity and may not capture nonlinear trends accurately, especially in highly variable data. The Cox-Stuart test [13] evaluates trends by comparing positive and negative differences in KPI data from two distinct periods. An upward trend is indicated if later data significantly exceeds earlier data; otherwise, a downward trend is identified. Nonetheless, this test does not account for the temporal nature of the data, relying solely on sign testing, which can overlook time-related complexities.

[1]https://github.com/limengyaoi/dataset

KPI anomaly detection. In the rapidly evolving field of software development, detecting anomalies in software systems is critical. KPI anomaly detection methods are categorized into supervised and unsupervised approaches.

a) Supervised KPI anomaly detection: Supervised methods, such as Support Vector Machines (SVM) [14] and decision trees [15], learn patterns from labeled datasets to distinguish between normal and abnormal data. These methods are effective with sufficient labeled data but struggle in industrial environments where labeled anomaly data is scarce and may fail to detect unknown anomaly types.

b) Unsupervised KPI anomaly detection: Unsupervised methods are more suitable when labeled data is scarce, learning normal patterns from the data without requiring labels.

- Statistical-Based Methods: Approaches like k-sigma [16] and box plot [17] assume KPI distribution and apply statistical inference to detect anomalies. They are computationally efficient and calculate thresholds dynamically, but their outcomes depend heavily on the selection of methodological parameters.
- Deep Learning-Based Methods: Techniques such as Donut [18], Bagel [19], and Buzz [20] use Variational Autoencoders (VAE) [21] to reconstruct KPIs and identify anomalies. However, the opacity of these models and their prolonged initialization phase pose challenges in scenarios requiring high computational efficiency and data interpretability.

In summary, while supervised methods are effective with sufficient labeled data, the challenges in obtaining such data make unsupervised methods more practical in many real-world scenarios. However, both approaches have limitations, and the choice of method should consider the specific requirements and constraints of the application.

III. Approach

A. Overview

As shown in Fig. 2, the framework of *Auto-PIP* consists of two main components: optimal and maximum inflection point identification. In the optimal inflection point identification part, *Auto-PIP* detects trends in the preprocessed throughput data, determining if the optimal inflection point occurs during the stress testing process. For the maximum inflection point identification part, *Auto-PIP* preprocesses the response time and success rate data, then performs the anomaly detection in the response time and success rate data, identifying the concurrent user number corresponding to the abrupt change as the maximum inflection point.

B. Data Preprocessing

In the complex stress testing environments of large modern software systems, data noise is inevitable due to network instability and potential hardware and software failures. This noise can obscure KPI and abnormal signals. Therefore, data smoothing and differencing are applied before performing KPI trend and anomaly detection.

Smoothing. The goal of smoothing is to remove random fluctuations and reveal the trends and patterns in the data more clearly. Smoothing is achieved by calculating the mean of consecutive data points within a window, effectively reducing random fluctuations $\text{mean}(x) = \frac{1}{n}\sum_{i=1}^{n} x_i$. It helps identify and understand the underlying trends more clearly.

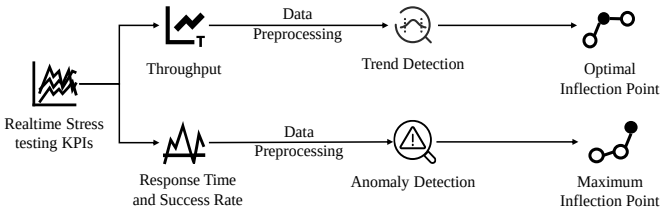

Fig. 2. The framework of *Auto-PIP*.

Differencing. Differencing aims to remove the trend and seasonal components in the response time data. It involves calculating the difference between consecutive observations: $\text{diff}(x_i) = x_i - x_{i-1}$, where x_i is the observation at the ith time point in the time series, x_{i-1} is the observation at the previous time point, and $\text{diff}(x_i)$ is the difference between these two consecutive observations.

By implementing these preprocessing steps, *Auto-PIP* effectively mitigates the impact of noise, enhancing the detection of meaningful performance trends and mutations during stress testing.

C. Optimal Inflection Point Identification

Mann-Kendall test. The Mann-Kendall test [22] is a non-parametric test that does not require the sample to follow a specific distribution and is robust against outliers. The null hypothesis of the Mann-Kendall test states that the data shows no trend, while the alternative hypothesis posits that the data shows an upward or downward trend.

In time series analysis, the Mann-Kendall test identifies monotonic trends (increasing, decreasing, or stable) by comparing the value of each data point with previous points. Each data point (except the first) is compared with previous points, and differences are recorded as 1 (positive), -1 (negative), or 0 (equal). The sum of these differences, S, is calculated as follows:

$$S = \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} sgn(x_j - x_k) \tag{1}$$

$$sgn(x) = \begin{cases} 1 & if\ x > 0 \\ 0 & if\ x = 0 \\ -1 & if\ x < 0 \end{cases} \tag{2}$$

A positive S indicates an increasing trend, while a negative S indicates a decreasing trend. For $n \geq 8$, S approximately follows a normal distribution with mean $E(S) = 0$ and variance $Var(s) = \frac{n(n-1)(2n+5)}{18}$. The standardized test statistic Z is calculated as follows:

$$Z = \begin{cases} \frac{S-1}{\sqrt{Var(S)}} & S > 0 \\ 0 & S = 0 \\ \frac{S+1}{\sqrt{Var(S)}} & S < 0 \end{cases} \tag{3}$$

The p-value is then calculated from the standard normal distribution. If the p-value is less than the chosen significance level, the null hypothesis is rejected, indicating a trend. Otherwise, the null hypothesis is accepted, suggesting the data is stable.

The optimal inflection point identification process. As shown in Fig. 3, the optimal inflection point identification process involves acquiring throughput from the most recent specified stress testing step and performing the Mann-Kendall test after smoothing the data. The optimal inflection point is identified if the test result indicates a stable trend. This method ensures robust trend detection by mitigating the influence of outliers and providing a reliable assessment of the throughput data during stress testing.

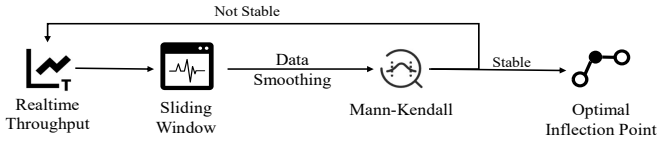

Fig. 3. The process of the optimal number of concurrent users identification.

D. Maximum Inflection Point Identification

SPOT [23]. SPOT is an algorithm based on extreme value theory, dynamically sets anomaly thresholds for KPIs. As shown in Fig. 4, SPOT consists of two primary components: calibration and detection. During the calibration phase, the anomaly threshold z_q and the peak threshold t are calculated based on the first n values of the entire sequence. During this phase, the first n data points and a predefined risk probability q are input into the POT model [24] to determine z_q and t. In the subsequent detection phase, any X_i exceeding z_q is considered an anomaly. Data points falling between z_q and t are regarded as peaks and are used to dynamically update the Generalized Pareto Distribution (GPD) model and the threshold z_q in a streaming manner. Data points below t are considered normal.

Maximum inflection point identification process. The process of identifying the maximum inflection point involves analyzing both the success rate and response time data. The process begins with smoothing the incoming data window, followed by these analyses:

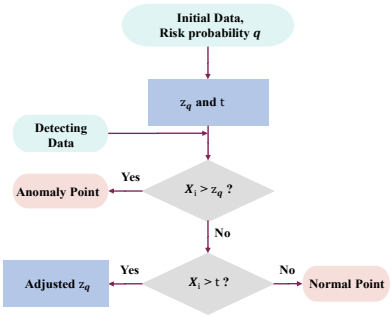

Fig. 4. The running process of SPOT.

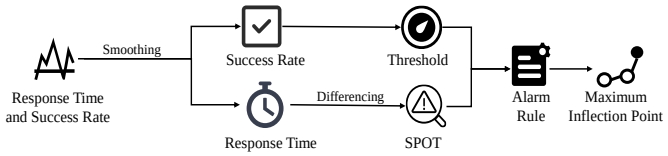

Fig. 5. The process of the maximum inflection point identification.

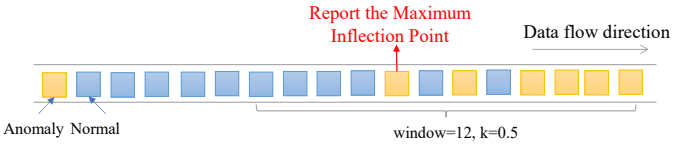

Fig. 6. The process of the maximum inflection point identification.

1) Success rate analysis: When the success rate falls below a certain threshold, the concurrent user number at that point is identified as a potential maximum inflection point.

2) Response time analysis: Simultaneously, SPOT is employed to detect any sudden shifts in the response time, which could indicate a potential maximum inflection point.

A customized alarm rule is used to mitigate interference caused by system jitter. It involves maintaining two queues to track recently detected suspicious maximum inflection points, as shown in Fig. 6. The maximum inflection point is confirmed when the proportion of suspicious points in the window reaches the threshold k within a specified period. At this point, a signal is sent to stop the stress testing. This structured approach ensures the accurate identification of significant inflection points by leveraging both statistical anomaly detection and customized alarm rules to handle potential data noise and system instability.

IV. Evaluation

In this section, we evaluate the performance of *Auto-PIP* using a dataset collected from the industrial environment of *Huawei Cloud*. We aim to answer the following research questions (RQs):

RQ1: How does the performance (accuracy and efficiency) of *Auto-PIP* compare to the baseline methods?

RQ2: Does each component of *Auto-PIP* significantly contribute to its performance?

A. Experiment Setup

Datasets. Based on past experience, identifying the optimal and maximum inflection points requires manual labeling, which is time-consuming and laborious. In order to evaluate the accuracy of *Auto-PIP* identifying the optimal and maximum inflection points, we invite experienced test experts to manually label the user intervals of the optimal and maximum inflection points. A total of 128 cases constitute the test data set, denoted as D.

Baseline methods and evaluation metrics. Due to the scarcity of methods directly related to inflection point identification, we primarily selected baselines from the transformed problems. In the optimal inflection point identification process,

we compared *Auto-PIP* with the slope method [12] and the Cox-Stuart test [13], both of which are KPI trend detection algorithms. For identifying the maximum inflection point, we compared *Auto-PIP* with the k-sigma method [16], the box plot [17], and Bagel [19], which are all KPI anomaly detection algorithms. Accuracy and efficiency metrics were employed to evaluate *Auto-PIP*:

$$Accuracy = \frac{1}{|A|}\sum_{a \epsilon A} II(f(a) \epsilon Y_a) \tag{4}$$

A denotes the test case set. Y_a denotes the interval range of the number of concurrent users corresponding to the inflection point of the test case, annotated by test experts with five years of experience. $f(a)$ denotes the optimal or maximum inflection point predicted by *Auto-PIP*. Efficiency refers to the algorithm's runtime during a 10-minute stress test data session.

B. Auto-PIP vs. Baseline Methods (RQ1)

Table I demonstrates that the time consumption of the three methods in the optimal inflection point identification is similar. However, considering the accuracy, *Auto-PIP* outperforms the other two methods. During the maximum inflection point identification phase, although *Auto-PIP* consumes more time than the baseline methods, it achieves higher accuracy within an acceptable time.

Given the critical importance of identifying the maximum inflection point for determining system capacity, we also evaluate the performance of Bagel, a representative deep learning-based KPI anomaly detection method. Experimental results show that Bagel's accuracy is lower than that of *Auto-PIP*, and its execution time is considerably longer. This discrepancy may be due to deep learning models' challenges in learning effective knowledge from short-term stress testing data. Moreover, the method's inefficiency could hinder the timely identification in practical scenarios, increasing the risk of system crashes.

TABLE I
OVERALL PERFORMANCE OF *Auto-PIP* AND BASELINE METHODS

Detection Type	Method	Accuracy	Efficiency
Optimal	Slope method [12]	53.3%	0.231s
	Cox-Stuart test [13]	58.3%	**0.220s**
	Auto-PIP	**100%**	0.228s
Maximum	K-sigma [16]	73.2%	2.085s
	Box plot [17]	21.4%	**1.398s**
	Bagel [19]	66.1%	5.830s
	Auto-PIP	**83.9%**	2.229s

C. Contribution of Key Components (RQ2)

To identify the maximum inflection point, *Auto-PIP* considers sudden drops in success rates and uses SPOT to detect abrupt increases in response time data. Table II shows that *Auto-PIP* without SPOT achieves higher execution efficiency but lacks accuracy. Additionally, *Auto-PIP* without a success rate threshold results in lower accuracy for maximum inflection point identification and increased reporting time due to the relaxed condition restrictions, leading to higher time consumption. Therefore, monitoring the decline in success rates while using SPOT to identify anomalies in response time data is necessary.

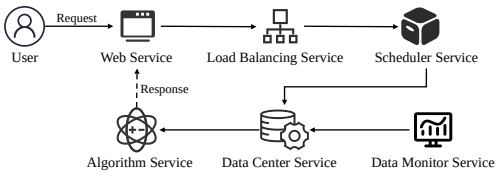

Fig. 7. The workflow of *Auto-PIP* in *Huawei Cloud*' industrial environment.

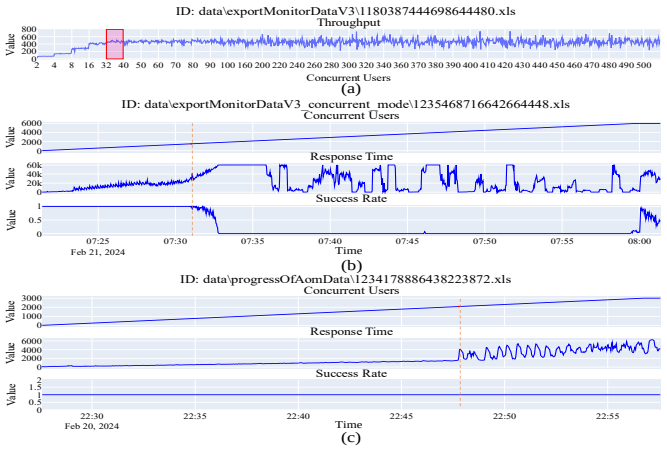

Fig. 8. The Cases of optimal in and maximum inflection points: the red rectangle represents the identified optimal inflection and the red point represents the identified maximum inflection points.

TABLE II
CONTRIBUTION OF KEY COMPONENTS

Model	Accuracy	Efficiency
Auto-PIP	**83.9%**	**2.229s**
Auto-PIP w/o SPOT	60.7%	0.359s
Auto-PIP w/o success rate	21.4%	2.790s

V. DEPLOYMENT

A. Workflow of Auto-PIP's Deployment

As shown in Fig. 7, the workflow of *Auto-PIP* in the industrial environment of *Huawei Cloud* includes the following steps. First, the test user initiates a stress test request via the front-end web service. This request is routed to the scheduler service by the load balancer HAProxy [25]. The scheduler service then allocates the necessary computational resources for the stress test task and instructs the data center service to push KPI data to the algorithm service (i.e., *Auto-PIP*). Finally, the algorithm service analyzes the real-time KPI data to identify performance inflection points and returns the results to the front-end web service.

B. Case Study

To better understand the workflow and practical effectiveness of *Auto-PIP*, we present three cases from the dataset D, as shown in Fig. 8.

(1) Optimal inflection point. As shown in Fig. 8(a), when the system transitions from a light load to a smooth load, the

throughput ceases to increase as the number of concurrent users rises. *Auto-PIP* successfully identifies this transition point, determining that 32 concurrent users represent the optimal inflection point.

(2) Maximum inflection point due to success rate drop. As shown in Fig. 8(b), when the number of concurrent users linearly increases to 1512, a significant number of user requests fail. Due to the sharp drop in the success rate, *Auto-PIP* promptly reports the occurrence of the maximum inflection point.

(3) Maximum inflection point due to response time surge. As shown in Fig. 8(c), in the light and smooth load areas, the response time gradually increases with the growth in concurrent users. However, when the number of concurrent users reaches 2082, the response time suddenly surges from 1400 ms to 4000 ms and subsequently fluctuates significantly. *Auto-PIP* accurately identifies this phenomenon and reports the maximum inflection point.

Based on the above analysis, it is evident that *Auto-PIP* can accurately identify critical performance inflection points, offering valuable insights into system capacity and performance optimization.

VI. Conclusion

Conducting stress testing is crucial to ensuring the reliability of software systems. In this paper, we propose *Auto-PIP*, a novel approach that combines trend testing and unsupervised anomaly detection to identify optimal and maximum inflection points during stress testing. Experiments conducted on a real-world dataset of *Huawei Cloud* demonstrate the effectiveness and efficiency of *Auto-PIP*. Moreover, *Auto-PIP* has been successfully deployed and verified within *Huawei Cloud*, demonstrating its practicability. In the future, we intend to integrate *Auto-PIP* into our product service CodeArts PerfTest [2], aiming to deliver an intelligent full-link stress testing experience to a broader user base.

References

[1] Dongliang Nan, Jinlong Tan, Lu Zhang, Jaydar Jingus, Chang Wang, and Weilin Liu. Research on the remote automatic test technology of the full link of the substation relay protection fault information system. *Energy Reports*, 8:1370–1380, 2022.

[2] Kun Qiu, Zheng Zheng, Kishor S Trivedi, and Beibei Yin. Stress testing with influencing factors to accelerate data race software failures. *IEEE Transactions on Reliability*, 69(1):3–21, 2019.

[3] Zhen Ming Jiang and Ahmed E Hassan. A survey on load testing of large-scale software systems. *IEEE Transactions on Software Engineering*, 41(11):1091–1118, 2015.

[4] Yan Gong and Lin Huang. A reliable and efficient stress generation and control system. *International Journal of Future Computer and Communication*, 8(2), 2019.

[5] HongHui Li, XiuRu Li, Huan Wang, Jie Zhang, and ZhouXian Jiang. Research on cloud performance testing model. In *2019 IEEE 19th International Symposium on High Assurance Systems Engineering (HASE)*, pages 179–183, 2019.

[6] Alex S Santos, Andre K Horota, Zhizhen Zhong, Juliana De Santi, Gustavo B Figueiredo, Massimo Tornatore, and Biswanath Mukherjee. An online strategy for service degradation with proportional qos in elastic optical networks. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2018.

[2]https://www.huaweicloud.com/intl/zh-cn/product/cpts.html

[7] Hua Guo, Chen-Jiang Guo, Yan Qu, and Jun Ding. Pattern synthesis of concentric circular antenna array by nonlinear least-square method. *Progress In Electromagnetics Research B*, 50:331–346, 2013.

[8] Heru Nugroho, Nugraha Priya Utama, and Kridanto Surendro. Normalization and outlier removal in class center-based firefly algorithm for missing value imputation. *Journal of Big Data*, 8:1–18, 2021.

[9] Jiahao Bu, Ying Liu, Shenglin Zhang, Weibin Meng, Qitong Liu, Xiaotian Zhu, and Dan Pei. Rapid deployment of anomaly detection models for large number of emerging kpi streams. In *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, pages 1–8. IEEE, 2018.

[10] Nengwen Zhao, Jing Zhu, Yao Wang, Minghua Ma, Wenchi Zhang, Dapeng Liu, Ming Zhang, and Dan Pei. Automatic and generic periodicity adaptation for kpi anomaly detection. *IEEE Transactions on Network and Service Management*, 16(3):1170–1183, 2019.

[11] Ali Bou Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Mohamad Dakalbab. Machine learning for anomaly detection: A systematic review. *Ieee Access*, 9:78658–78700, 2021.

[12] Samuel Hawley, M Sanni Ali, Klara Berencsi, Andrew Judge, and Daniel Prieto-Alhambra. Sample size and power considerations for ordinary least squares interrupted time series analysis: a simulation study. *Clinical epidemiology*, pages 197–205, 2019.

[13] Ana F Militino, Mehdi Moradi, and M Dolores Ugarte. On the performances of trend and change-point detection methods for remote sensing data. *Remote Sensing*, 12(6):1008, 2020.

[14] Derek A Pisner and David M Schnyer. Support vector machine. In *Machine learning*, pages 101–121. Elsevier, 2020.

[15] Bahzad Charbuty and Adnan Abdulazeez. Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, 2(01):20–28, 2021.

[16] Uzay Çetin and Mursel Tasgin. Anomaly detection with multivariate k-sigma score using monte carlo. In *2020 5th International Conference on Computer Science and Engineering (UBMK)*, pages 94–98. IEEE, 2020.

[17] JA Ndako, JA Olisa, IC Ifeanyichukwu, SKS Ojo, and CE Okolie. Evaluation of diagnostic assay of patients with enteric fever by the boxplot distribution method. *New Microbes and New Infections*, 38:100795, 2020.

[18] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pages 187–196, 2018.

[19] Zeyan Li, Wenxiao Chen, and Dan Pei. Robust and unsupervised kpi anomaly detection based on conditional variational autoencoder. In *2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, pages 1–9. IEEE, 2018.

[20] Wenxiao Chen, Haowen Xu, Zeyan Li, Dan Pei, Jie Chen, Honglin Qiao, Yang Feng, and Zhaogang Wang. Unsupervised anomaly detection for intricate kpis via adversarial training of vae. In *IEEE INFOCOM 2019-IEEE conference on computer communications*, pages 1891–1899. IEEE, 2019.

[21] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[22] Martine Collaud Coen, Elisabeth Andrews, Alessandro Bigi, Giovanni Martucci, Gonzague Romanens, Frédéric PA Vogt, and Laurent Vuilleumier. Effects of the prewhitening method, the time granularity, and the time segmentation on the mann–kendall trend detection and the associated sen's slope. *Atmospheric measurement techniques*, 13(12):6945–6964, 2020.

[23] Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. Anomaly detection in streams with extreme value theory. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1067–1075, 2017.

[24] Paul Embrechts, Claudia Klüppelberg, and Thomas Mikosch. *Modelling extremal events: for insurance and finance*, volume 33. Springer Science & Business Media, 2013.

[25] Luthfan Hadi Pramono, Robby Cokro Buwono, and Yanuar Galih Waskito. Round-robin algorithm in haproxy and nginx load balancing performance evaluation: a review. In *2018 international seminar on research of information technology and intelligent systems (ISRITI)*, pages 367–372. IEEE, 2018.