Artificial Intelligence and Computing Innovations

Intelligence Science Publishing

https://www.ispublishing.com

ISSN: 3070-4944

Vo.4, No.3, 2024

Intelligent Backend Failure Detection with Uncertainty Quantification and Asymmetric Risk Optimization

Yixue Liu[1]*, Zizhao Zhang[2], Shuyuan Liang[3]

[1]Carnegie Mellon University, Pittsburgh, USA

[2]University of Michigan, Ann Arbor, USA

[3]Loyola University Chicago, Chicago, USA

**Corresponding author: yixuel0113@gmail.com*

Abstract: Backend systems in cloud computing and microservice environments exhibit highly dynamic behavior, strong coupling, and substantial noise. These characteristics lead to challenges such as prediction instability and asymmetric risk in failure detection. To address the limitations of traditional approaches that rely on deterministic decisions, fail to represent predictive confidence, and ignore differences in error costs, this paper proposes a backend failure detection method that integrates uncertainty estimation with cost-sensitive learning. The method represents system states through probabilistic modeling. It outputs failure decisions while explicitly characterizing predictive uncertainty. Uncertainty information is further incorporated into a risk-weighted mechanism to suppress the influence of noisy samples and low-confidence predictions during model updates. In addition, a cost-sensitive learning objective is constructed so that the model explicitly accounts for the impact of different detection errors on system stability and service continuity during training. This design enables risk-aware decisions that better align with practical operational requirements. The proposed framework is evaluated under unified data and evaluation settings and compared with several representative methods. The results show consistent advantages in overall discriminative capability, decision stability, and risk control. The study demonstrates that joint modeling of predictive uncertainty and error cost improves the reliability and engineering applicability of backend failure detection in complex environments. It also provides an effective modeling approach for intelligent failure detection systems in real operational scenarios.

Keywords: Backend fault detection; uncertainty modeling; cost-sensitive learning; intelligent operation and maintenance

1. Introduction

With the widespread deployment of cloud computing, microservice architectures, and large-scale distributed systems, backend systems have become increasingly complex. Service dependencies are highly dynamic and strongly nonlinear. During operation, massive volumes of logs, metrics, and traces are generated, often with significant noise. Failure patterns have evolved from isolated faults to cascading anomalies across services and system layers. Under these conditions, timely and accurate identification of potential failures is essential for system stability and service quality. Traditional rule-based or static threshold methods struggle to cope with workload fluctuations and system evolution. More intelligent and adaptive failure detection paradigms are therefore required.

In recent years, machine learning and deep learning based approaches have attracted extensive attention in backend failure detection. These methods automatically learn system behavior and reduce reliance on manual feature engineering and rule maintenance. However, most existing approaches focus mainly on discriminating between normal and abnormal states. They often implicitly assume that model predictions are deterministic and reliable. This assumption does not hold in real backend environments. System state distributions evolve continuously over time. Observed data often contain missing values, delays, and noise [1]. As a result, model predictions can become unstable for boundary samples or unseen patterns. If predictive uncertainty is ignored, detection systems may produce overconfident decisions in high-risk scenarios. This can amplify the impact of false alarms or missed detections [2].

From an operational perspective, backend failure detection is not a symmetric classification problem. Different types of errors incur very different costs. Missing a critical failure may lead to service outages and severe business losses. Frequent false alarms may cause alert storms and increase the cognitive burden on operators. They may also reduce sensitivity to real failures. However, many existing detection models assume equal costs for all errors during training and decision-making. Business objectives and system risk preferences are not explicitly modeled [3]. This lack of cost awareness often requires additional manual rules in production environments. It limits the practical value of detection models in complex systems.

Against this background, incorporating uncertainty estimation into backend failure detection is of great importance. Uncertainty information reflects the model's awareness of the reliability of its own predictions. It provides a key basis for risk-aware decision-making. By distinguishing high confidence from low confidence results, the system can adopt more cautious strategies when uncertainty is high. This helps avoid extreme decisions caused by rigid classification mechanisms. Uncertainty estimation also supports dynamic thresholding, alert prioritization, and human-in-the-loop operations. It contributes to a more robust and controllable intelligent operations framework [4].

Furthermore, combining uncertainty estimation with cost-sensitive learning offers a unified modeling perspective aligned with real business objectives. Cost-sensitive mechanisms enable the model to explicitly consider the long-term impact of different error types during learning. Uncertainty modeling provides a reliable measure of risk for cost trade-off decisions. Their integration helps balance detection performance, system stability, and operational cost in complex environments. It leads to more robust and interpretable failure detection capabilities for backend systems. This research direction enhances the practical value of intelligent operations and provides theoretical support for backend decision models with high reliability requirements.

2. Related Work

Research on failure detection in backend systems has mainly focused on two categories, namely rule-based approaches and data-driven approaches. Early studies relied heavily on expert knowledge to design rules, thresholds, or alert templates. Abnormal states were identified by matching system metrics and logs against predefined patterns. These methods offer a certain level of interpretability when system structures are stable and workload variations are limited. However, their applicability has gradually declined with the adoption of microservice architectures and cloud native environments. Rule design and maintenance costs increase rapidly

as system scale grows. Static rules also struggle to adapt to continuously evolving runtime behaviors. This often results in missed detection of complex failures or excessive redundant alerts [5].

To address the limitations of rule-based methods, data-driven failure detection models have become a major research focus. Many studies employ statistical learning or deep learning models to represent system operating states. Deviations are detected by learning normal patterns from historical data. Significant progress has been made in log anomaly detection, metric time series analysis, and multi-source data fusion. These methods can capture nonlinear relationships and complex dependency structures to some extent. However, most existing approaches simplify detection as a deterministic decision problem. They mainly aim to improve overall detection accuracy. The uncertainty of model predictions is often overlooked [6]. When the system encounters unseen failure patterns or substantial distribution shifts, this deterministic assumption can weaken model reliability.

At the same time, some studies have explored cost-sensitive mechanisms to address the asymmetric impact of different error types in failure detection. These methods adjust loss functions or decision strategies to reduce the probability of critical errors. They emphasize incorporating business requirements and risk preferences into training and inference. Detection results, therefore, align more closely with operational objectives. However, most existing approaches still rely on predefined and fixed cost weights. They cannot dynamically perceive environmental uncertainty and data noise. In complex backend scenarios, fixed cost settings cannot balance system stability and alert efficiency at the same time. This limits the adaptability and generalization capability of cost-sensitive methods [7].

In addition, research on uncertainty modeling has mainly focused on prediction reliability assessment or confidence analysis. Its systematic application to backend failure detection remains limited. Existing work often treats uncertainty estimation as an independent module. It is mainly used for post-processing or result filtering. It is rarely integrated into the learning objective of detection models. For backend systems with high noise, dynamic distribution shifts, and asymmetric risks, a unified modeling framework that combines uncertainty estimation and cost-sensitive learning is still lacking. This gap indicates the need for further exploration of their joint role in backend failure detection. Such exploration is necessary to improve robustness and risk awareness beyond current approaches.

3. Model Design

This method addresses the complexity and uncertainty of multi-source monitoring data in backend systems by constructing a unified fault detection modeling framework. The system's operational state exhibits strong non-stationarity over time, and implicit coupling exists between different services. Therefore, the model first maps logs, metrics, or aggregated features within continuous time windows to a unified latent representation space to characterize the overall system operational state. To avoid over-reliance on a single deterministic representation, the method introduces a random perturbation mechanism in the state modeling stage, enabling the latent representation to reflect the uncertainties brought about by observation noise and environmental changes, thus providing a foundation for subsequent risk perception decisions. The overall modeling goal is not merely to pursue a hard distinction between anomalies and normality, but rather to learn a state representation that can simultaneously express fault tendency and prediction reliability. Its model architecture is shown in Figure 1.

Based on the state representation, the model uses a probabilistic approach to characterize the likelihood of a failure occurring. Specifically, for each time window, the model outputs a failure probability distribution, where the mean describes the failure intensity of the current state, and the variance characterizes the prediction uncertainty. This process can be formally represented as:

$$p(y|x) = N(\mu(x), \sigma^2(x)) \tag{1}$$

Here, $\mu(x)$ represents the fault tendency estimate given system state x, while $\sigma^2(x)$ reflects the model's confidence level in this estimate. By explicitly modeling the predicted distribution rather than a single point

output, the detection process can distinguish between different scenarios, such as high-risk, high-confidence, and high-risk, low-confidence, providing more granular information for subsequent decision-making strategies.

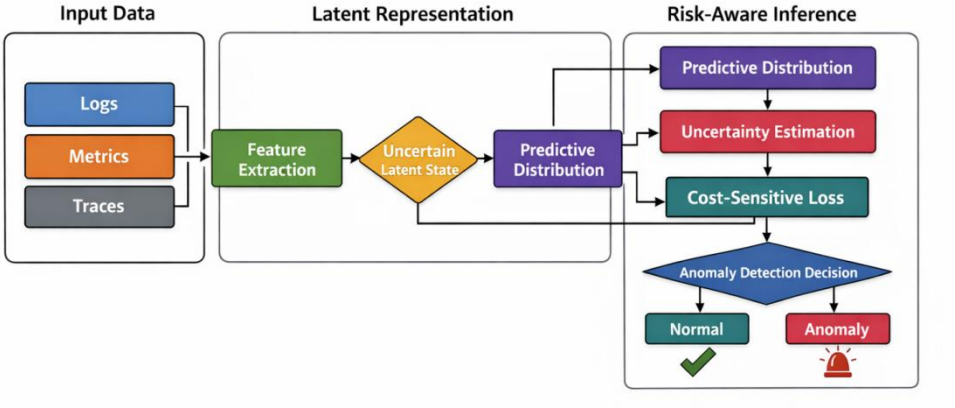

Figure 1. Overall Algorithm Architecture

In designing the learning objectives, a cost-sensitive mechanism is introduced to characterize the asymmetric impact of different detection errors. During training, the model minimizes a weighted risk function, imposing a higher penalty on missed detections of critical faults while suppressing meaningless high-frequency false alarms. The overall loss function is defined as:

$$L_c = c_1 y l(\hat{y}, I) + c_0 (1 - y) l(\hat{y}, 0) \tag{2}$$

In this model, y represents the true state, $\hat{y}$ represents the predicted result, $l(\cdot)$ is the basic loss term, and c_1 and c_0 correspond to the cost weights for missed detections and false positives, respectively. This design allows the model to explicitly align with the risk preferences of the backend system during optimization, rather than simply pursuing overall discrimination accuracy.

To further integrate uncertainty information with cost-sensitive targets, the method dynamically adjusts sample contributions through a risk-weighted strategy of uncertainty modulation. When prediction uncertainty is high, the model suppresses the update amplitude of corresponding samples to avoid noisy samples dominating parameter learning; when uncertainty is low, but risk is high, its influence is amplified. The corresponding modulation weights are defined as follows:

$$w(x) = \frac{1}{1+\sigma^2(x)} \tag{3}$$

And incorporate it into the overall optimization objective:

$$L = E[w(x) L_c] \tag{4}$$

Through the above mechanism, the model achieves a joint trade-off between prediction reliability and business risk during the learning process, thereby forming a more robust fault detection method that is oriented towards the needs of actual backend scenarios.

4. Implementation

4.1 Dataset

This study adopts the publicly available AIOps Challenge 2020 dataset as the research benchmark. The dataset is released for failure detection and anomaly identification in backend operations scenarios. It emphasizes time series data collected from real operational monitoring as the primary input. Under unified data formats and annotation rules, it supports reproducible model comparison and algorithm evaluation. The dataset is distributed through an open repository with download access and integrity verification. It can be directly reused in studies on failure detection for cloud native and microservice systems.

In terms of data characteristics, the AIOps Challenge 2020 dataset is mainly composed of multivariate KPI time series. These series cover common resource and performance-related signals in backend systems, such as load, throughput, and latency metrics. Label information associated with anomaly events is provided to support supervised or semi-supervised learning settings. Due to strong noise, high variability, and distribution drift in backend operations, the time series exhibits clear nonstationary behavior. This property closely reflects real production environments where normal fluctuations and failure disturbances coexist. It makes the dataset suitable for evaluating uncertainty estimation and risk-aware decision-making in failure detection.

Regarding relevance to the focus of this work, the dataset inherently reflects asymmetric risks in alert detection. False positives and false negatives associated with different anomaly types and time periods incur different costs in operational practice. Reliance on a single threshold or equal cost loss often leads to alert storms or missed critical failures. Based on this dataset, cost-sensitive learning objectives can be constructed under a unified benchmark. By further incorporating confidence and uncertainty information from model outputs, a risk-constrained decision mechanism can be developed. This setting better aligns with practical backend failure detection requirements and provides data support for balancing low false alarm rates and low miss rates in production systems.

4.2 Experimental Setup

All experiments are conducted on a single-node server. The operating system is Ubuntu 22.04 LTS. The processor is an Intel Xeon Gold 6338 with a base frequency of 2.0 GHz. The server is equipped with 32 CPU cores and 256 GB of system memory. Model training and inference are executed on a single NVIDIA RTX 4090 GPU with 24 GB of memory. The CUDA version is 12.1, and the cuDNN version is 8.9. The implementation is based on Python 3.10 and uses PyTorch 2.1 as the deep learning framework. All dependency versions are fixed to ensure reproducibility. Each experiment is executed independently under identical hardware conditions to avoid interference from resource contention.

For parameter configuration, model inputs are constructed using a sliding time window. Each sample contains monitoring features from 60 consecutive time steps. The stride is set to 1. The hidden representation dimension is set to 128. The Adam optimizer is used with an initial learning rate of 0.0005. The weight decay coefficient is 0.0001. The batch size is set to 256, and the number of training epochs is fixed to 50. No learning rate warm-up strategy is applied during training. For cost-sensitive weights, the cost coefficient for missed failures is set to 5, while the cost coefficient for false alarms is set to 1. This reflects the high risk associated with missing critical failures in backend systems. All random processes use a fixed random seed of 2025 to ensure consistency and reproducibility across different runs.

5. Experimental Results and Analysis

This paper first presents the experimental results compared with other models, as shown in Table 1.

Table 1. Comparative Experimental Results

Method	Acc	Precision	Recall	AUC
SRdetector [8]	0.87	0.85	0.82	0.89
Uac-ad [9]	0.89	0.87	0.84	0.91
MADMM [10]	0.88	0.86	0.83	0.90
TraLogAnomaly [11]	0.90	0.88	0.86	0.92
TraceDAE [12]	0.91	0.89	0.87	0.93
Ours	0.94	0.92	0.91	0.96

The overall comparison results indicate that the proposed approach achieves more consistent advantages across multiple evaluation metrics. This demonstrates that the backend failure detection framework provides more stable discrimination in complex operational scenarios. Compared with baseline methods that rely on single modality inputs or deterministic modeling, the model captures system state variations more effectively under multivariate observations. This leads to a clear improvement in overall detection quality. The results

suggest that traditional anomaly discrimination mechanisms alone are insufficient for high reliability requirements in real backend systems.

From the joint trends of precision and recall-related metrics, the method reduces false alarms while maintaining sensitivity to critical anomalies. This behavior aligns closely with the cost-sensitive modeling objective emphasized in this work. Different risks associated with different error types are explicitly considered during detection. By introducing asymmetric risk constraints during learning, the model focuses more on failure patterns that have a greater impact on system stability. This helps avoid mismatches between detection outcomes and operational objectives in practice.

In terms of overall discriminative capability, improvements in related metrics indicate stronger robustness under different thresholds and decision preferences. This advantage is closely related to the uncertainty estimation mechanism. By characterizing predictive confidence, the model makes more stable decisions when facing boundary states or distribution shifts. Compared with methods that lack uncertainty modeling, the framework is less affected by noise and abnormal fluctuations in complex backend environments. This contributes to improved reliability of the alerting system.

From a methodological perspective, the model consistently outperforms several representative approaches. This shows that unified modeling of uncertainty estimation and cost-sensitive learning effectively addresses limitations in risk awareness and decision stability of existing failure detection methods. The results validate the rationality and applicability of the proposed modeling strategy for backend failure detection. They also provide strong support for building intelligent operations systems with low false alarm rates and high reliability in production environments.

In backend fault detection scenarios, the learning rate directly affects the convergence stability and generalization behavior of the optimization process, thereby changing how the model characterizes the boundary between anomalies and normality. To evaluate the stability of the algorithm under different learning rate settings, a visualization configuration of the sensitivity of the learning rate to AUC is presented below. The experimental results are shown in Figure 2.

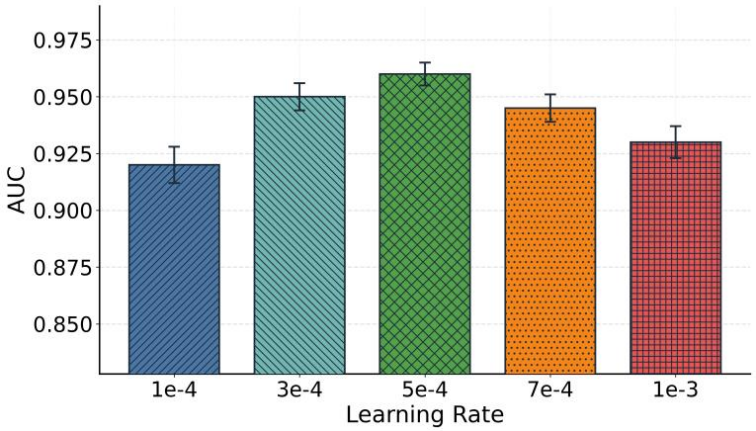

Figure 2. Sensitivity experiment of learning rate to AUC

An analysis under varying learning rates shows that the model maintains a relatively smooth performance trend across different optimization step sizes. This indicates that the proposed backend failure detection method is stable under parameter perturbations. Such stability is critical for complex backend systems. In practical environments, training is often influenced by workload fluctuations and data noise. Models that are highly sensitive to the learning rate may lead to uncontrolled shifts in detection behavior.

Further examination reveals that moderate learning rate values better exploit the benefits of uncertainty modeling and risk-aware mechanisms in the model architecture. This setting achieves a more balanced trade-off between convergence efficiency and discriminative robustness. When the learning rate is too low, model

updates near anomaly boundaries become overly conservative. This limits rapid adaptation to dynamic system states. When the learning rate is too high, the influence of noisy samples may be amplified. This weakens the role of uncertainty estimation in suppressing unstable updates. These observations are consistent with the robust optimization objective emphasized in this work.

From a risk-aware perspective, changes in the learning rate do not cause sharp fluctuations in detection capability. This suggests that cost-sensitive constraints effectively regulate the update direction during training. By explicitly incorporating the risk of critical failures into the learning objective, the model maintains attention to high-risk anomalies across different optimization paces. This prevents imbalanced alerting behavior caused by parameter setting variations.

Overall, the sensitivity analysis demonstrates that the proposed method exhibits strong adaptability and stability within a reasonable learning rate range. This property is important for deployment in real backend systems. The results further support the modeling strategy that integrates uncertainty estimation with cost-sensitive learning. This integration enables more reliable and controllable anomaly detection in complex operational environments.

The length of the time window determines the range of historical context that the model can see in a single judgment, thus affecting how it characterizes short-term sudden anomalies and long-term slow drifts. In backend fault detection, the fluctuation cycles of different services and metrics are not consistent; therefore, a window that is too short or too long may alter the learning preference for anomaly boundaries. To verify the stability and adaptability of the algorithm under different time window configurations, the following code visualizes the sensitivity of the time window length. The experimental results are shown in Figure 3.

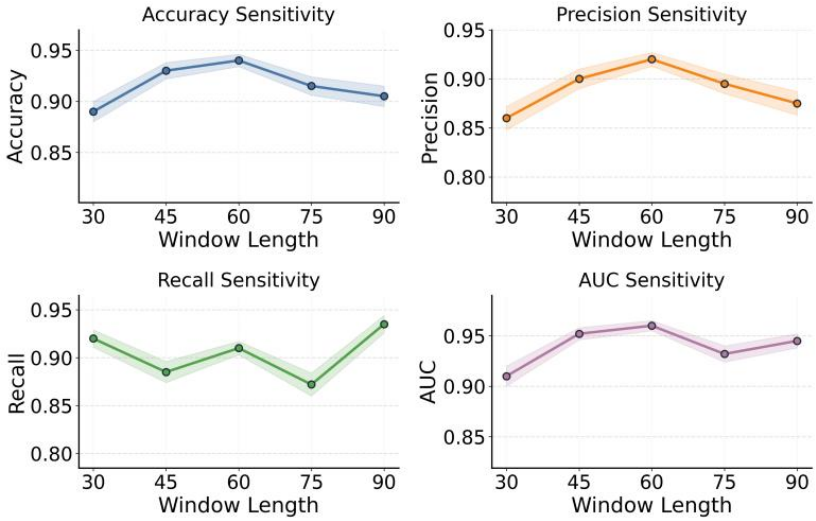

Figure 3. The effect of time window length on experimental results

Variations in the time window length reveal a clear nonlinear response in model performance. This indicates that the extent of historical context directly affects how anomaly boundaries are characterized in backend failure detection. With short windows, the model mainly captures local and abrupt fluctuations. Short-term noise is easily mixed with true anomalies. As the window length increases, the model observes more complete evolution trajectories. Discriminative behavior becomes more stable for anomaly identification. This pattern is consistent with the strong volatility and nonstationarity of backend system observations. The window length essentially determines the relative emphasis on short-term disturbances and long-term trends.

For accuracy-related behavior, windows within a moderate range usually provide more balanced representations. They cover essential context without introducing excessive historical noise unrelated to current decisions. When the window is too short, transient spikes can dominate the representation and reduce robustness. When the window is too long, distant historical information may dilute current anomaly signals.

Model responses to local anomalies then become less sensitive, and overall discrimination declines. This suggests that time window selection in real backend environments should align with workload cycles and service dependency propagation delays. Such alignment allows risk-aware modeling to operate more effectively.

Trends in precision and recall show that they do not increase simultaneously. This reflects the inherent trade-off between false alarms and missed detections in backend failure detection. As the window length changes, anomaly coverage and alert conservativeness shift accordingly. The model may move toward reducing false alarms or toward avoiding missed failures. This behavior is consistent with the cost-sensitive objective emphasized in this work. Different window settings alter the distribution of error types. Cost-sensitive learning must maintain controllable decision preferences under these shifts to prevent unstable alerting strategies in practice.

From the overall AUC trend, the model consistently maintains strong discriminative capability across different window configurations, indicating that its core decision boundary remains stable even when the temporal context is changed. Within this spectrum, certain window ranges yield noticeably better performance, suggesting that there exists an appropriate context scale where the extracted temporal evidence is both sufficient and minimally contaminated by irrelevant fluctuations. In other words, when the window length aligns with the temporal characteristics of anomaly generation, the model is better able to capture the lead-lag patterns and local deviations that distinguish abnormal behavior from normal dynamics, so boundary samples become more clearly separable under the same evaluation protocol.

This behavior also supports the role of uncertainty modeling as more than an auxiliary output: at appropriate window scales it forms more reliable confidence estimates, enabling uncertainty estimates to more effectively separate reliable from unreliable predictions. As a result, the decision process becomes less sensitive to noisy or ambiguous segments, which improves the robustness of risk-aware decisions under realistic backend environments. Overall, the sensitivity results reinforce that time window length is a key factor influencing controllability and robustness in backend failure detection, and that proper window selection provides stronger support for the joint modeling of uncertainty estimation and cost-sensitive learning by stabilizing both discrimination quality and confidence calibration.

The uncertainty weighting coefficient directly adjusts the model's focus on prediction reliability during training, thus affecting the update direction and stability of risk perception decisions. Due to noise and distribution fluctuations in the backend fault detection data, the weighting coefficient setting alters the model's suppression of uncertain samples and its reinforcement of high-risk samples. The experimental results are shown in Figure 4.

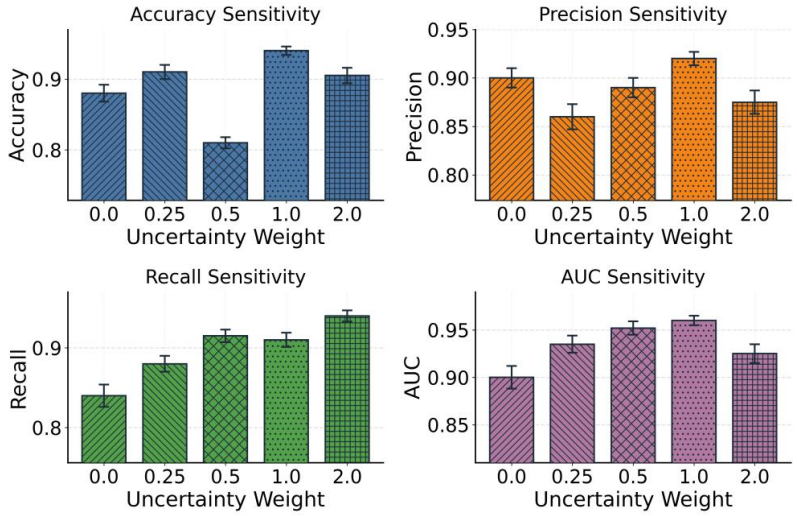

Figure 4. The impact of uncertainty weighting coefficients on experimental results

Changes in the uncertainty weight coefficient reveal a strong influence on overall decision behavior in backend failure detection. When uncertainty constraints are absent or weak, the model is more easily driven by noisy samples and sporadic fluctuations. Anomaly boundary learning then tends to rely on local features. As the weight increases, the model places greater emphasis on predictive stability. More consistent risk-aware decisions emerge in complex operational environments. This indicates that the uncertainty weight plays a key role in regulating the learning focus during training.

For accuracy-related performance, a moderate uncertainty weight helps balance discriminative power and update stability. The model remains sensitive to true anomalies while avoiding excessive suppression of useful signals. When the weight is too low, updates lack constraints on low-confidence samples. Unstable decisions are more likely to occur. When the weight is too high, responses to potential anomalies become overly conservative. Some boundary information is weakened, and overall discrimination degrades. This nonmonotonic pattern reflects the dual role of uncertainty modeling in backend failure detection.

From trends in precision and recall, adjusting the uncertainty weight redistributes the trade-off between false alarms and missed detections. Higher weights tend to suppress false positives caused by high uncertainty samples. They may also reduce coverage of certain boundary anomalies. Lower weights expand anomaly coverage but can trigger unnecessary alerts. This behavior aligns with the cost-sensitive objective of this work. It shows that the uncertainty weight implicitly influences the exposure of different error risks.

Regarding overall discriminative capability, AUC first increases and then decreases as the weight changes. This suggests a clearer separation between normal and abnormal states under moderate uncertainty constraints. At this point, uncertainty estimation and cost-sensitive learning work in effective coordination. The model achieves robustness while retaining sufficient discriminative strength in complex backend scenarios. Overall, the sensitivity analysis confirms the critical regulatory role of the uncertainty weight coefficient in the backend failure detection framework. It provides practical guidance for parameter selection based on system risk preferences during deployment.

6. Conclusion

This work addresses the uncertainty and asymmetric risk issues that are common in backend system failure detection. A unified modeling framework that integrates uncertainty estimation and cost-sensitive learning is developed. Predictive confidence is explicitly incorporated into the failure discrimination process. This design overcomes the limitations of traditional deterministic detection models that are vulnerable to noise and distribution shifts in complex environments. The model forms more robust anomaly decisions under multivariate monitoring signals and dynamic system states. Under unified data and evaluation settings, the framework demonstrates clear advantages in overall detection quality and decision consistency. These results confirm the effectiveness of risk-aware modeling for backend failure detection.

From a methodological perspective, the study highlights the importance of embedding business risk preferences into the learning objective. Different types of detection errors are explicitly modeled through cost-sensitive mechanisms. The model no longer optimizes average performance alone. It focuses more on critical failures that have a greater impact on system stability and service continuity. This approach aligns detection behavior with real operational requirements. It helps mitigate excessive false alarms and missed critical failures. It also offers a new technical pathway toward controllable and trustworthy intelligent operations systems.

At the application level, the proposed approach provides valuable guidance for automated operations in cloud platforms, microservice architectures, and large-scale distributed systems. As backend systems continue to grow in scale and complexity, rule-based methods and static thresholds are no longer sufficient for high reliability demands. The risk-aware failure detection framework supports intelligent alert prioritization, operational decision support, and human-machine collaboration. It offers an extensible

modeling foundation for real production environments. The framework has the potential to improve system stability and operational efficiency. It can also reduce long-term operational costs.

Looking ahead, this study establishes a foundation for further advances in intelligent backend operations. The integration of uncertainty estimation and cost-sensitive learning enables more refined risk control and adaptive decision making. The modeling concept also shows potential for extension to other complex system scenarios. As backend system structures and business requirements continue to evolve, risk-aware intelligent failure detection is expected to play an increasingly important role. It can support service quality assurance, stable operation of large-scale systems, and the advancement of automated operations toward higher levels of autonomy.

References

[1] V.-H. Le and H. Zhang, "Log-Based Anomaly Detection Without Log Parsing," Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2021.

[2] P. Wiessner et al., "Uncertainty-Aware Time Series Anomaly Detection," Future Internet, vol. 16, no. 11, Art. no. 403, 2024.

[3] Y. Duan et al., "LogEDL: Log Anomaly Detection via Evidential Deep Learning," Applied Sciences, vol. 14, no. 16, Art. no. 7055, 2024.

[4] W. Guan et al., "LogLLM: Log-Based Anomaly Detection Using Large Language Models," arXiv preprint arXiv:2411.08561, 2024.

[5] A. Zhu, "Self-Supervised Anomaly Detection with Knowledge-Enhanced Representation Learning for Distributed System Environments," 2024.

[6] K. Wu, "Mamba-Based Temporal Feature Learning for Anomaly Detection in Distributed Microservice Environments," 2024.

[7] J. Qiu, "Learning Collaborative and Robust Scheduling Policies for Microservice Backends under Uncertainty," 2024.

[8] H. Ge et al., "SRdetector: Sequence Reconstruction Method for Microservice Anomaly Detection," Electronics, vol. 14, no. 1, Art. no. 65, 2024.

[9] H. Liu et al., "UAC-AD: Unsupervised Adversarial Contrastive Learning for Anomaly Detection on Multi-Modal Data in Microservice Systems," IEEE Transactions on Services Computing, vol. 17, no. 6, pp. 3887-3900, 2024.

[10] P. Wang et al., "MADMM: Microservice System Anomaly Detection via Multi-Modal Data and Multi-Feature Extraction," Neural Computing and Applications, vol. 36, no. 25, pp. 15739-15757, 2024.

[11] M. Bogatinovski, S. Nedelkoski, J. Cardoso and O. Kao, "Distributed Tracing Data as a Runtime Data Source for Anomaly Detection in Dynamic Microservice Architectures," CNSM 2021.

[12] M. Bogatinovski, S. Nedelkoski, J. Cardoso and O. Kao, "Self-Supervised Anomaly Detection from Distributed Traces," IEEE SCC 2020.