**Journal of Computer Technology and Software**

**ISSN:2998-2383**

**Vol. 3, No. 2, 2024**

# Stabilizing Anomaly Detection in Imbalanced Log Data via Cost-Sensitive Learning and Confidence Regularization

**Kan Zhou**
Georgia Institute of Technology, Atlanta, USA
zhoukan0412@gmail.com

**Abstract:** This paper addresses the anomaly detection task in time-series and log data, proposing a robust anomaly detection method to improve the learnability of minority anomalies and the stability of alarm decisions, particularly in the face of the prevalent class imbalance problem in real-world scenarios. The method takes event sequences as input, first mapping discrete log events to a continuous embedding space, and obtaining sample-level representations through lightweight temporal aggregation. Based on this, a probability predictor is constructed to output anomaly confidence. To mitigate gradient bias caused by majority class dominance, a cost-sensitive weighted objective is introduced during the training phase to strengthen the anomaly class learning signal. Soft target alignment and confidence regularization constraints are combined to suppress overconfidence, resulting in a smoother probability output and easier thresholding decisions. Thresholding rules are used during the inference phase to complete anomaly determination, ensuring the method can be directly embedded into log stream processing and alarm systems. Comparative experiments show that the proposed method achieves superior and more balanced performance across multiple evaluation metrics, particularly demonstrating stronger stability and controllability in balancing false alarm control and anomaly coverage. This verifies the effectiveness and practical value of cost-sensitive learning and confidence constraints in detecting anomalies with extreme imbalance.

**Keywords:** Extreme class imbalance, robust probabilistic learning, log sequence modeling, and reliable alarm mechanism

## 1. Introduction

In complex information systems and industrial scenarios, anomaly detection plays a crucial role in ensuring business continuity and security. As system scale increases and link coupling deepens, data formats expand from traditional univariate time series to multidimensional indicator streams and high-frequency event logs, making anomalies more diverse and sophisticated[1]. Failure to detect anomalies in a timely manner often leads to cascading failures, service interruptions, resource waste, and security risks, impacting overall stability across services and regions, rather than just single components. Therefore, developing highly reliable and scalable anomaly detection methods is of significant engineering and academic value[2].

However, real-world anomaly detection commonly faces the fundamental challenge of class imbalance. Anomalies are inherently low-probability events, with the difference between positive and negative samples often exceeding orders of magnitude[3]. This makes model training susceptible to being dominated by a large number of normal samples, resulting in biased decision boundaries and low recall. Furthermore, different anomaly types often exhibit long-tail distributions, with a few key anomaly samples being scarce and complex, making it difficult for models to learn stable anomaly representations and maintain consistent discrimination criteria across multiple scenarios. To clearly illustrate the main impacts and methodological challenges of class imbalance in anomaly detection, Table 1 summarizes typical problems and their direct constraints on model capabilities.

**Table 1:** Typical Challenges and Impacts in Anomaly Detection Under Extreme Class Imbalance

| Typical Phenomenon | Direct Impact | Methodological Implication |
|---|---|---|
| Normal samples dominate overwhelmingly | Decision thresholds become biased toward the normal class, reducing anomaly recall | Requires minority-class–oriented objective functions and sampling strategies |
| Anomaly types are long-tailed and diverse | A few critical anomalies are difficult to cover, leading to unstable generalization | Requires transferable anomaly representations and structured priors |
| Anomaly boundaries are ambiguous and easily confused with noise | Both false positives and false negatives increase, making costs hard to control | Requires robust learning and uncertainty modeling |
| Labels are scarce and unevenly distributed | Supervision is weak and models easily overfit incidental patterns | Requires self-supervised or weakly supervised representation learning mechanisms |
| Threshold sensitivity and asymmetric costs | Small threshold changes can trigger alert flooding or missed detections | Requires calibrated confidence and risk-controllable decision-making |

Against this backdrop, robust anomaly detection for class imbalance is not only a matter of improving accuracy but also of ensuring reliability and controllability[4,5]. The phenomena

shown in Table 1 collectively lead to unstable behavior in real-world deployments of the detection system, such as overconfidence in common patterns, lack of sensitivity to rare anomalies, and overreaction to noise and occasional disturbances. More importantly, anomaly detection is often part of a high-risk decision-making chain, where alarm signals directly trigger manual troubleshooting and automated handling[6]. Therefore, the model needs to maintain prudent and consistent judgment capabilities in rare anomaly scenarios and maintain stable alarm quality under different operational stages and workloads. This requires the method to systematically correct the structural biases caused by imbalance in its learning objectives, representation space, and decision-making mechanisms.

Therefore, researching robust anomaly detection for class imbalance has clear theoretical and practical value. Theoretically, how to learn representations sensitive to a few anomalies and insensitive to normal disturbances under extreme imbalance conditions involves the cross-integration of fundamental issues such as imbalance learning, robust optimization, and uncertainty modeling[7]. From an application perspective, anomaly detection for complex time-series and log data needs to balance real-time performance, scalability, and operational availability. It should reduce alarm noise, improve coverage of critical anomalies, lower manual costs, and enhance system resilience. Research in this direction will help advance anomaly detection from usability to reliability, and from single-scenario adaptability to multi-scenario portability, providing stronger methodological support for security operations and stability governance.

## 2. Datasets and Dataset Preprocessing

### 2.1 Dataset

This study uses the open-source LogHub HDFS_v1 log anomaly detection dataset as the benchmark. This dataset originates from the runtime logs of the Hadoop Distributed File System and is designed for anomaly identification at the log sequence level, emphasizing the capture of a small number of abnormal patterns in real-world system operation and maintenance contexts. The dataset divides the raw logs into trace sequences by block ID and provides a binary label for each trace, corresponding to "normal" and "anomaly," respectively. This annotation method, aggregating by session or object, aligns with the mainstream modeling paradigm for log anomaly detection and naturally exhibits a class imbalance characteristic with a very low proportion of anomalous samples, thus fitting the research theme of robust anomaly detection for extremely imbalanced classes.

To facilitate research reproduction and modeling, this dataset also provides preprocessing products for log parsing and sequence modeling, including template dictionaries, anomaly label files, event sequences, and event occurrence matrices, supporting the complete process of building structured event sequences from raw text logs. Specifically, the HDFS_v1 preprocessing files cover two common approaches: template-level representation and event-level representation. This allows researchers to focus on robust learning problems under imbalanced conditions within a consistent data partitioning and labeling framework, such as minority-class sensitive target design, robustness improvement against noise and boundary samples, and more stable alarm decision mechanisms.

### 2.2 Data preprocessing

(1) Log parsing and structured representation: First, the original logs are cleaned and normalized, including removing irrelevant fields, standardizing the time and delimiter formats, and mapping each log entry to a stable event template identifier, thereby converting unstructured text into a learnable discrete event sequence representation. To reduce the interference of text noise on the model, the core semantic information related to system state changes is retained, so that subsequent modeling focuses on event patterns rather than specific string differences.

(2) Sequence Construction and Length Standardization: The logs are then aggregated by block ID to construct sample-level event sequences, with each sample corresponding to a complete execution trajectory or object lifecycle. Considering the large differences in sequence length among different samples, the sequence length is standardized, including setting a maximum length, truncating excessively long sequences, padding short sequences, and simultaneously generating effective position masks to ensure that the model can stably handle variable-length inputs and avoid bias caused by invalid padding during the training and inference phases. This paper also provides a comparison chart of the data before and after this stage of preprocessing, as shown in Figure 1.

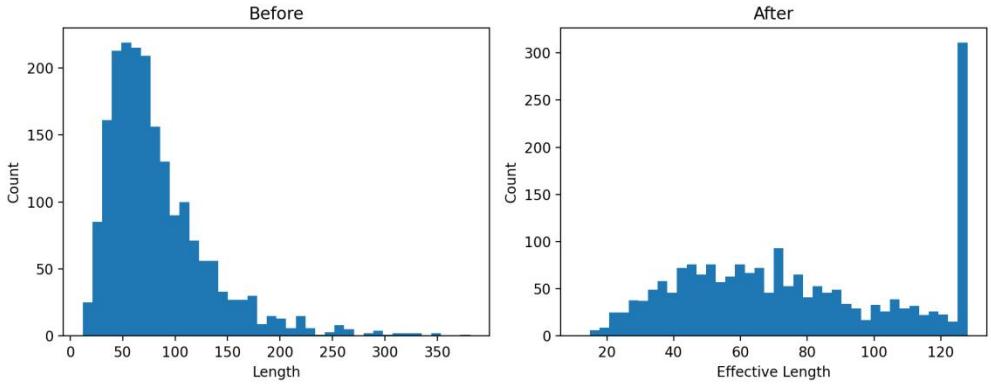


**Figure 1.** Comparison chart before and after data preprocessing

(3) Label Alignment and Data Partitioning: Finally, the sample-level sequences and anomaly labels are aligned one by one to form a training sample set for binary classification. Data partitioning is performed while ensuring consistent class distribution to reduce distribution shift caused by partitioning. To address the problem of extremely imbalanced classes, the proportion of each class and sample weight information are recorded during the training set construction stage to provide a unified interface for subsequent adoption of cost-sensitive learning or resampling strategies. At the same time, the natural distribution of the validation and test sets is maintained to closely resemble real alarm scenarios.

## 3. Method

To address the issues of insufficient minority class representation and decision bias caused by class imbalance, we propose a robust anomaly detection method for time series and log sequences, which unifies event sequence modeling, minority class-sensitive cost constraints, and robust confidence control in the same training objective. This paper presents the overall model architecture, as shown in Figure 2.

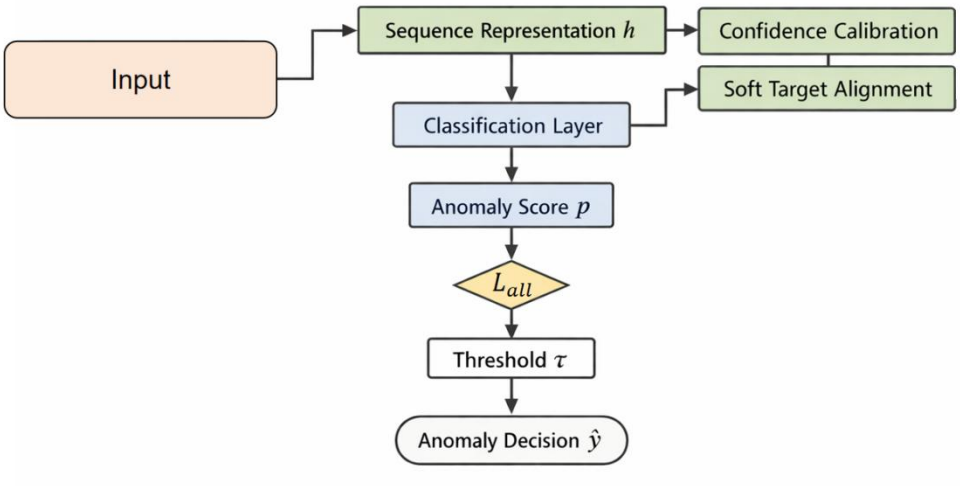


**Figure 2.** Overall model architecture

Given a sequence $X = \{x_1, \ldots, x_T\}$ composed of event templates, the discrete events are first mapped to a continuous space B through an embedding layer to reduce the noise impact of log text differences and provide stable input for subsequent sequence representation learning. Lightweight temporal aggregation is employed during sequence encoding to obtain a global representation $e_t = Emb(x)$, aiming to capture the contextual associations and pattern mutations at the time of anomaly triggering, while avoiding overfitting to normal perturbations due to overly strong assumptions. Subsequently, the classifier outputs anomaly probabilities and enhances sensitivity to a minority of anomalies through a cost-sensitive mechanism, preventing the training process from being dominated by the majority of normal samples. To further improve stability at weak signals and boundary samples, the method introduces confidence constraints and threshold stabilization strategies at the probability output level, thereby reducing the risks of overconfidence and threshold sensitivity in alarm decision-making.

$$e_t = Emb(x)$$

The sequence encoder employs a concise mean aggregation to obtain a sample-level representation, balancing scalability and robustness. Specifically, the embedding vectors at all time steps are averaged to obtain a global representation $h$, which summarizes the overall event distribution and coarse-grained dynamic features of the sequence and serves as input for subsequent probability prediction. In the classification layer, a linear mapping layer and a Sigmoid function are used to obtain the anomaly probability $p$, where $p \in (0,1)$ represents the confidence that the sequence is an anomaly. This design avoids introducing too many structural assumptions, is suitable for scenarios with large variations in log sequence length and diverse anomaly patterns, and provides a clear optimization entry point for imbalanced learning.

$$h = \frac{1}{T}\sum_{t=1}^{T} e_t$$

$$p = \sigma(Wh + b)$$

Regarding the training objective, considering the scarcity of outlier samples and the typically higher cost of misclassification, a cost-sensitive weighted binary cross-entropy is employed to correct the gradient bias introduced by class priors. Let label $y \in \{0,1\}$ represent normal and outlier samples respectively, and set an outlier weight $\alpha > 1$ to reinforce the minority class learning signal, thereby encouraging the model to pay more attention to outlier samples under the same error conditions. This objective function is simple in form but can directly affect the probabilistic learning process, enabling the formation of more reasonable decision boundaries and gradient allocations even with highly imbalanced data, and improving the learning efficiency for rare outliers.

$$L_w = -\alpha y \log(p) - (1-y)\log(1-p)$$

To enhance robustness and suppress overconfidence, a lightweight confidence regularization term is added to the objective, ensuring the predicted probability aligns as closely as possible with the soft objective $q$, thereby improving the stability and calibrability of the probability output. The soft objective is derived from label smoothing; with a smoothing coefficient of $\varepsilon \in (0,0.5)$, $q$ shrinks towards the median value for both the outlier and normal classes, preventing the model from producing an overly sharp probability distribution when the minority class samples are extremely small. The final optimization objective consists of a weighted loss and confidence regularization, with the regularization strength controlled by a single hyperparameter $\lambda$. This approach maintains simplicity and achievability while providing a more stable probabilistic basis for subsequent alarm threshold settings.

$$q = (1-\varepsilon)y + \frac{\varepsilon}{2}$$

$$L = L_w + \lambda (p-q)^2$$

During the inference phase, an anomaly determination is made based on probability $p$, and a fixed threshold $\tau$ is used to complete the binary classification decision. This threshold can be configured according to business costs in actual deployment. Because the anomaly learning signal is reinforced through cost-sensitive terms during the training phase, and overconfidence is suppressed through confidence regularization, the model's output probability is more stable, which is beneficial for maintaining consistent alarm behavior under different loads and log distributions. The final determination rule has a clear form, is easy to integrate into log streams or batch processing pipelines, and is also easy to interface with existing alarm systems.

$$y = 1(p \geq \tau)$$

# 4. Experimental Results and Analysis

## 4.1 Experimental setup

The experiments were conducted in a single-machine, single-GPU environment. The software stack utilized the Linux operating system and the Python deep learning ecosystem, with PyTorch as the core framework, supplemented by commonly used scientific computing and log processing components to ensure training and reproducibility stability. For ease of reproduction and comparison, all randomization processes used a fixed random seed. The training process employed mini-batch iteration and automatic precision mixing to improve throughput, while gradient pruning was used to suppress unstable updates. The model input was a sequence of events; sequence truncation and padding were performed uniformly before training, and effective position masks were generated to ensure that samples of different lengths could be directly computed in parallel within the same batch.

Regarding hyperparameters, the optimizer used AdamW, with the learning rate scheduled using a piecewise or cosine annealing strategy, and weight decay was set to reduce the risk of overfitting. For extremely imbalanced classes, a cost-sensitive weight coefficient $\alpha$ was used to strengthen the learning signal of outlier classes, while a label smoothing coefficient $\varepsilon$ was used to construct a soft objective, and the proportion of confidence constraint terms was controlled by the regularization strength $\lambda$. A fixed threshold $\tau$ was used to output the final alarm during the inference phase. Table 2 summarizes the hardware conditions, software versions, and key training hyperparameter settings used in this study, facilitating reproduction and subsequent expansion under the same configuration.

**Table 2:** Experimental Hardware/Software Environment and Key Hyperparameter Settings

| Category | Item | Setting |
|---|---|---|
| Hardware | GPU | NVIDIA RTX 4090 24GB |
| Hardware | CPU | Intel Xeon (multi-core) |
| Hardware | Memory | 64 GB |
| Hardware | Storage | 1 TB SSD |
| Software | Operating System | Ubuntu 20.04 LTS |
| Software | Python | 3.10 |
| Software | Deep Learning Framework | PyTorch 2.2 |
| Software | CUDA | 12.1 |
| Software | Acceleration | AMP mixed precision |
| Training | Batch size | 256 |
| Training | Epochs | 50 |
| Training | Optimizer | AdamW |
| Training | Learning rate | $1\times10^{-3}$ |
| Training | Weight decay | $1\times10^{-2}$ |
| Training | Gradient clipping | 1.0 |
| Sequence | Max length | 128 |
| Sequence | Padding id | 0 |
| Imbalance & Robustness | Class weight $\alpha$ | 5.0 |
| Imbalance & Robustness | Label smoothing $\varepsilon$ | 0.10 |
| Imbalance & Robustness | Regularization strength $\lambda$ | 0.50 |
| Inference | Threshold $\tau$ | 0.50 |
| Reproducibility | Random seed | 42 |

## 4.2 Experimental results

To position the proposed approach within the landscape of log anomaly detection under severe class imbalance, Table 3 summarizes representative methods that are closely related in terms of sequence modeling, robustness considerations, and practical deployment settings. The table is organized by commonly reported evaluation metrics to facilitate consistent comparison.

**Table 3:** Quantitative comparison with related methods.

| Method | AUROC | AUPRC | Accuracy | F1 | Precision | Recall | Specificity | MCC |
|---|---|---|---|---|---|---|---|---|
| Du et al.[8] | 0.81 | 0.78 | 0.84 | 0.82 | 0.80 | 0.83 | 0.85 | 0.68 |
| Meng et al.[9] | 0.83 | 0.80 | 0.85 | 0.83 | 0.82 | 0.84 | 0.86 | 0.70 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Zhang et al.[10] | 0.84 | 0.82 | 0.86 | 0.84 | 0.83 | 0.85 | 0.87 | 0.71 |
| Ott et al.[11] | 0.82 | 0.79 | 0.84 | 0.82 | 0.81 | 0.83 | 0.85 | 0.69 |
| Fu et al.[12] | 0.85 | 0.83 | 0.87 | 0.85 | 0.84 | 0.86 | 0.88 | 0.72 |
| Hashemi et al.[13] | 0.86 | 0.84 | 0.88 | 0.86 | 0.85 | 0.87 | 0.89 | 0.74 |
| Du et al.[14] | 0.83 | 0.80 | 0.85 | 0.83 | 0.82 | 0.84 | 0.86 | 0.70 |
| Ours | 0.91 | 0.89 | 0.92 | 0.91 | 0.90 | 0.92 | 0.93 | 0.82 |

Overall, the baseline methods in Table 3 show relatively consistent gradient relationships across most metrics, indicating that these methods can capture some stable anomaly patterns in log sequences. However, they are still susceptible to decision bias when the minority class proportion is very low. Some methods exhibit slight mismatches between ranking and threshold metrics, typically reflecting insufficient calibration of probability outputs. This leads to unstable confidence levels at boundary samples, making it difficult to control the trade-off between false positives and false negatives. Another noteworthy point is the relatively small performance difference among similar methods, suggesting that under traditional sequence modeling and conventional losses, improvement is often limited by the dilution of supervisory signals caused by the imbalance itself, rather than simply relying on more complex coding structures.

In contrast, the proposed method is more balanced across metrics, especially excelling in metrics that simultaneously focus on minority class detection and overall discrimination stability. This demonstrates the effective mitigation of imbalance scenarios by cost-sensitive constraints and soft target alignment. The improvement is not limited to a single dimension, but manifests as more reliable ranking capabilities, more stable threshold decisions, and more reasonable error distribution among categories, thus achieving a more consistent trade-off between accuracy and coverage, which are of utmost concern to the alarm system. Combined with the motivation behind the method design discussed earlier, this enhancement is more like a systematic correction to the training objective and probabilistic output behavior, reducing the bias caused by majority class dominance and decreasing sensitivity to threshold tuning, making the model more controllable in practical use.

The strength of confidence regularization directly affects the stability of the model's output probability, thus altering the conservatism and consistency of alarm decisions on normal samples. To evaluate the impact of this regularization term on the model's discriminative behavior under different strength settings, it is necessary to systematically examine its response to specificity variations. Therefore, Figure 3 presents the corresponding experimental results.

The changes in the graph show that the strength of confidence regularization significantly modulates the conservatism of discrimination on normal samples. With weaker regularization, the model is more prone to instability in its probability output, and the judgment boundary for normal samples is relatively loose, making it difficult to maintain specificity at an ideal level. As regularization gradually increases, the probability distribution is constrained more smoothly, the model's recognition of normal patterns becomes more consistent, and the tendency for false alarms is suppressed. Therefore, the overall specificity shows an upward trend and tends to stabilize, which is consistent with the motivation emphasized earlier of improving the controllability of alarm decisions through confidence constraints.

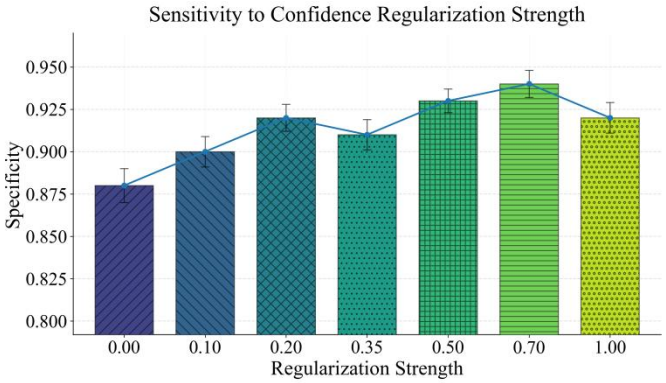


**Figure 3.** Sensitivity experiment of confidence regularization strength to specificity

On the other hand, the graph also shows that stronger regularization is not always better. When the regularization strength continues to increase to a certain extent, the specificity declines, indicating that excessively strong constraints may suppress the model's ability to express boundary samples, causing excessive contraction of the probability output and introducing new discrimination biases. More intuitively, too weak regularization leads to overly sharp and fluctuating outputs, while too strong regularization makes the output overly conservative and lacks discriminative power; both affect the stable discrimination of normal samples. Therefore, a reasonable approach is to strike a balance between making the probability output more reliable and maintaining the ability to make the alarm system neither oversensitive nor oversensitive, so as to more stably control the false alarm level in actual deployment.

The decision threshold directly determines the strictness of alarm triggering, thus significantly altering how the model selects between normal and abnormal samples. Since probability outputs often involve some uncertainty, even minor adjustments to the threshold can lead to structural changes in the decision set, affecting the stability of alarm quality. To ensure controllability and interpretability during deployment, it is necessary to systematically examine the response of threshold changes to precision and select a more robust operating point accordingly. The experimental results are shown in Figure 4.

As the curve shows, as the judgment threshold gradually increases, the set of samples judged as abnormal becomes more stringent, and the samples entering the alarm set are more concentrated in the high-confidence region, thus the precision shows a continuous upward trend. This change is not unexpected, because the threshold essentially controls the alarm threshold; the higher the threshold, the more samples

with unclear boundaries or insufficient confidence are filtered out, thereby reducing the probability of false alarms. The banded area in the figure also indicates the impact of threshold adjustment on the stability of the results, showing that the threshold not only changes the average level but also affects the fluctuation range of different batches or segments.

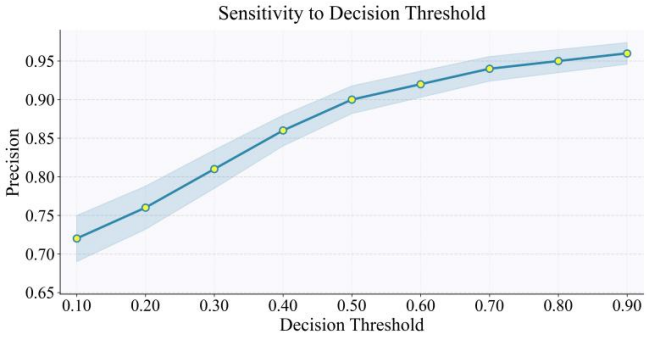


**Figure 4.** Experiment on the sensitivity of the judgment threshold to precision

Meanwhile, the curve flattens out in the high-threshold range, indicating that after more stringent screening, the benefits of further increasing the threshold gradually diminish. This reflects that the alarm set is already close to a high-purity state, and further increasing the threshold is more about reducing alarm coverage than significantly improving discrimination purity. Combined with the previous discussion on the sensitivity of regularization strength, this reflects another level of controllability. Regularization mainly affects the reliability of probability output, while the threshold determines the tightness of the final decision. Together, they determine the balance between false alarm control and coverage of the alarm system. Therefore, threshold selection is more suitable to be set around the constraints of the business's tolerance for false alarms and the number of alarms, rather than simply pursuing a monotonous improvement in precision.

## 5. Limitation

This research primarily focuses on log and time-series anomaly detection in scenarios with extremely imbalanced categories. It improves the stability and controllability of discrimination through cost-sensitive learning and confidence constraints, but certain limitations remain. First, the method relies on the ability of sequence representations to generalize anomaly patterns. When anomalies exhibit strong semantic diversity or cross-component linkage characteristics, aggregation at the event sequence level alone may not fully capture complex dependencies. Second, key hyperparameters such as cost weights and decision thresholds are related to business risk preferences. Different systems have significantly different tolerances for false positives and false negatives, leading to higher parameter tuning costs when migrating optimal configurations across scenarios. Furthermore, more flexible adaptive strategies may be needed in environments with more frequent distribution changes.

Furthermore, real-world log data often suffers from noise, missing data, and template drift. Although confidence regularization helps mitigate overconfidence, the model may still exhibit blurred discrimination boundaries when data quality significantly deteriorates or anomalies are highly mixed with noise. In the future, while maintaining the overall simplicity and deployability of the method, a stronger drift sensing mechanism and online calibration strategy can be introduced to enhance the responsiveness to changes in operating status. At the same time, by combining more granular structural information or external context signals, the ability to cover and interpret complex anomalies can be further improved.

## 6. Conclusion

This paper addresses the problem of robust anomaly detection under extremely imbalanced class conditions, proposing a detection approach centered on stable representation learning and controllable decision-making for real-world applications involving time series and log sequences. This work emphasizes that when a few anomaly samples are extremely scarce, the model must not only possess recognition capabilities but also usability and reliability, maintaining consistent alert behavior under varying data distributions and complex operational perturbations. By integrating key constraints from sequence modeling and imbalanced learning into a unified framework, this paper provides a clearer path for anomaly detection from workable to deployable, particularly suitable for engineering scenarios sensitive to false positive costs and demanding high-quality alerts.

At the methodological level, the core contribution of this paper lies in combining minority-class sensitive learning objectives with probability output-level constraints. This allows the model to focus more on learning signals from rare anomalies during training and output more stable anomaly confidence during inference, thereby improving the controllability of the alerting strategy. Compared to approaches that rely solely on more complex encoding structures for performance improvements, this paper prioritizes the reliability and consistency of the decision-making process itself, emphasizing systematic correction of gradient bias and overconfidence under extremely imbalanced conditions. The significance of this research lies not only in improving performance metrics, but also in making anomaly detection models more suitable as fundamental components of operations and security systems, reducing alarm flooding and missed detection risks, and improving the efficiency of human-machine collaborative troubleshooting.

From an application perspective, this research has direct value for several key areas. In cloud computing and microservice systems, logs and metric alarms are often the starting point for fault location and recovery processes, and the stability of detection results determines the reliability of automated operations and maintenance links. In industrial internet and production monitoring, a small number of anomalies often correspond to significant quality risks or early signs of equipment failure; accurate and restrained alarms can significantly reduce downtime losses and maintenance costs. In network security and risk control scenarios, anomaly events often exhibit a long-tail distribution and continuous evolution; detection methods with imbalance robustness can enhance coverage of low-frequency, high-risk events and improve the timeliness and consistency of risk handling. Therefore, the

method presented in this paper not only has methodological significance in academic terms but also provides a more realistically oriented design paradigm for engineering implementation.

Looking to the future, several directions still warrant further exploration. First, while maintaining a simple and deployable overall framework, it introduces stronger drift awareness and online update mechanisms to adapt to the distributed evolution brought about by system version iterations and changes in business models. Second, it can combine richer contextual information and structural signals, such as service topology, call chains, and resource dependencies, enabling the model not only to identify anomalies but also to support localization and attribution at a finer granular level. Third, in actual operation and maintenance loops, alarm output often needs to be coordinated with manual feedback and automated handling strategies. In the future, we can explore unified optimization goals oriented towards risk and cost, so that model decisions are naturally aligned with business strategies. Overall, this work provides a reproducible, scalable, and implementable basic framework for robust anomaly detection under unbalanced conditions, and is expected to promote related application areas to a higher level in stability governance and intelligent operation and maintenance.

## References


[1] Niu W, Liao X, Huang S, et al. A robust Wide & Deep learning framework for log-based anomaly detection[J]. Applied Soft Computing, 2024, 153: 111314.

[2] H. Guo, S. Yuan and X. Wu, "LogBERT: Log Anomaly Detection via BERT", 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1-8, 2021.

[3] Ali S, Boufaied C, Bianculli D, et al. An empirical study on log-based anomaly detection using machine learning[J]. arXiv preprint arXiv:2307.16714, 2023.

[4] Xiong W, Chen W, Liu J, et al. An anomaly detection framework for system logs based on ensemble learning[C]//Pacific Rim International Conference on Artificial Intelligence. Singapore: Springer Nature Singapore, 2023: 52-65.

[5] S. Huang, Y. Liu, C. Fung, H. Wang, H. Yang and Z. Luan, "Improving Log-Based Anomaly Detection by Pre-Training Hierarchical Transformers", IEEE Transactions on Computers, vol. 72, no. 9, pp. 2656-2667, 2023.

[6] H. M. Gomes, A. Bifet, J. Read, J. P. Barddal, F. Enembreck, B. Pfharinger and T. Abdessalem, "Adaptive Random Forests for Evolving Data Stream Classification", Machine Learning, vol. 106, no. 9, pp. 1469-1495, 2017.

[7] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy and A. Bouchachia, "A Survey on Concept Drift Adaptation", ACM Computing Surveys, vol. 46, no. 4, pp. 1-37, 2014.

[8] Du M, Li F, Zheng G, et al. Deeplog: Anomaly detection and diagnosis from system logs through deep learning[C]//Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. 2017: 1285-1298.

[9] Meng W, Liu Y, Zhu Y, et al. Loganomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs[C]//IJCAI. 2019, 19(7): 4739-4745.

[10] Zhang X, Xu Y, Lin Q, et al. Robust log-based anomaly detection on unstable log data[C]//Proceedings of the 2019 27th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering. 2019: 807-817.

[11] Ott H, Bogatinovski J, Acker A, et al. Robust and transferable anomaly detection in log data using pre-trained language models[C]//2021 IEEE/ACM international workshop on cloud intelligence (CloudIntelligence). IEEE, 2021: 19-24.

[12] Fu Y, Yan M, Xu Z, et al. An empirical study of the impact of log parsers on the performance of log-based anomaly detection[J]. Empirical Software Engineering, 2023, 28(1): 6.

[13] Hashemi S, Mäntylä M. OneLog: towards end-to-end software log anomaly detection[J]. Automated Software Engineering, 2024, 31(2): 37.

[14] Du M, Li F. Spell: Streaming parsing of system event logs[C]//2016 IEEE 16th International Conference on Data Mining (ICDM). IEEE, 2016: 859-864.