

هشام باشطح

Explainable Transformer-Based Anomaly Detection for Large-Scale System Logs V3.1.1.docx

 Midocean University

Document Details

Submission ID

trn:oid::2945:378878635

Submission Date

Apr 22, 2026, 12:45 AM GMT+2

Download Date

Apr 22, 2026, 10:47 AM GMT+2

File Name

Explainable Transformer-Based Anomaly Detection for Large-Scale System Logs V3.1.1.docx

File Size

5.0 MB

36 Pages

5,234 Words

30,958 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.





Explainable Transformer-Based Anomaly Detection for Large-Scale System Logs

Prepared By

Name: Hesham Bashatah

ID NO.: 253001893

Name: Anas Barro

ID NO.: 253001086

Submitted in fulfillment of the requirements for the degree Master of
Science in Artificial Intelligence

At

Collage of Informatics

Midocean University

UAE

Supervisor Name:

Dr. Wesam Ahmed

Acknowledgements

Abstract

Table of Contents

Chapter 1 Introduction.....	9
1.1 Introduction.....	9
1.2 Problem Definition.....	10
1.3 Thesis Objectives.....	10
1.4 Thesis Contribution.....	10
1.5 Thesis Organization	11
Chapter 2 Background and Related Work	12
2.1 Introduction	12
2.2 Background	12
2.2.1 Self-supervised learning.....	13
2.2.2 Masked prediction	13
2.2.3 Transformer / self-attention sequence modeling.....	15
2.2.4 LLM-enriched representations.....	16
2.2.5 Uncertainty-aware learning.....	18
2.2.6 Graph reasoning and hierarchical abstraction	19
2.3 Explainable AI (XAI)	21
2.3.1 SHAP	21
2.3.2 LIME.....	22
2.4 Performance Metrics.....	23
2.4.1 Precision.....	24
2.4.2 Recall	24
2.4.3 F1-score.....	24

2.4.4 Accuracy.....	25
2.5 Literature Review	25
2.6 Chapter Summary.....	29
Chapter 3 Proposed Approach	30
Chapter 4 Experimental Results and Discussion	31
Chapter 5 Conclusion and Future Work	32

List of Abbreviations

List of Figures

Figure 2.1 Masked log prediction as a self-supervised pretext task. [6]	13
Figure 2.2 Reconstruction-based self-supervised learning for anomaly detection. [7]	14
Figure 2.3 Contrastive representation learning for separating normal and abnormal log patterns. [8]	14
Figure 2.4 Encoder-based Transformer modeling for log anomaly detection. [9].....	15
Figure 2.5 Hybrid architecture for log anomaly detection. [10]	16
Figure 2.6 LLM-based semantic enrichment before downstream detection. [11]	16
Figure 2.7 Fine-tuning a large language model on normal log behavior. [12]	17
Figure 2.8 LLM-assisted explanation for anomalous log sequences. [13].....	17
Figure 2.9 Confidence-aware anomaly scoring pipeline. [14]	18
Figure 2.10 Evidential deep learning for uncertainty-aware log anomaly detection. [15].....	19
Figure 2.11 Graph-based modeling of structural relationships in log data. [16]	19
Figure 2.12 Hierarchical abstraction from flat log streams to execution-aware units. [17].....	20
Figure 2.13 Modular detection framework with escalation to structure-aware reasoning. [18]	20
Figure 2.14 SHAP (SHapley Additive exPlanations). [19].....	22
Figure 2.15 LIME (Local Interpretable Model-agnostic Explanations). [20].....	23
Figure 3.1 Proposed Framework.....	Error! Bookmark not defined.

List of Tables

Table 1. Comparative summary of reviewed studies	28
--	----

Chapter 1 Introduction

1.1 Introduction

The use of system logs has gained immense importance in modern software architectures since they provide continuous logging of events, errors, warnings, and service interactions within complex computing environments. With the increase in complexity and diversity in computing environments, human analysis of system logs manually has been increasingly challenging, particularly in cloud and distributed architectures, which generate an enormous amount of events. Recent work in explainable anomaly detection in system logs suggests that any useful approach towards log analysis requires more than just generating alerts and needs to identify meaningful context-based relationships for effective detection and explanation [1]. Recently, several studies have also highlighted that contemporary log-based anomaly detection relies on improved representation learning and sequence modeling more than manual rules and simple statistics. This is especially true for large-scale systems, wherein the anomalous behavior may be dependent on contextual changes or dependencies over extended sequences of events. The self-supervised approaches and sequence-based models proposed in 2021 and beyond prove that effective anomaly detection requires not only accurate identification of anomalies but also proper learning of system behavior through log sequences [2].

However, in the real operational world, it is not only about the accuracy of the detection. There is also a need to understand the reason behind the anomalies' detection, whether the detection is reliable and whether the outputs generated are meaningful and useful for further diagnosis and reaction. As recent practice-oriented studies suggest, real expectations for log anomaly detection include not just predictive accuracy, but also interpretability, reliability and operational usefulness of the process [3]. The proposed research is dedicated to Explaining Log Anomalies with Transformers. The study will

consider the possibility of using Transformer models for sequence learning in the process of anomaly detection and also generating explanations for such anomaly detections.

1.2 Problem Definition

Massive logging is an inherent part of large software systems that contain vital information related to their operations, service interactions, performance issues, warnings, and faults. However, due to the high complexity of such data, its analysis by means of conventional techniques that use manual labor or simple rules is infeasible as they cannot cope with long-term interdependencies, context, and dynamic nature of the system behavior.

1.3 Thesis Objectives

The key focus of this thesis work is on developing an explainable transformer-based architecture for detecting anomalies in system log files. The goal here is to develop a framework that can be used to model log file sequences, incorporate the element of explainability within the anomaly detection process, and evaluate the performance of this architecture against appropriate baselines.

1.4 Thesis Contribution

This thesis makes the following main contributions:

- 1- It proposes an explainable Transformer-based framework for anomaly detection in large-scale system logs.
- 2- It integrates feature-level explanation into the anomaly detection process to identify the log events that contribute most strongly to each anomaly decision.
- 3- It applies Transformer-based log modeling and evaluates its effectiveness using performance metrics such as precision, recall, and F1-score.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

Chapter 2 provides the foundational background and reviews the relevant literature. It introduces key concepts including log anomaly detection, Transformer-based sequence modeling, explainable artificial intelligence (XAI) techniques, and performance evaluation metrics, followed by a critical review of related studies in the field.

Chapter 3 describes the proposed explainable framework, including log preprocessing, LogBERT-based anomaly detection, SHAP explanation module, and interpretability evaluation.

Chapter 4 presents the experimental setup, evaluation results, and discussion of the findings in terms of both detection performance and explanation quality.

Chapter 5 concludes the thesis and outlines directions for future work.

Chapter 2 Background and Related Work

2.1 Introduction

This chapter will provide the necessary theoretical and methodological context needed for the suggested research project. First, it will define the essential approaches that have influenced research in the field of log analysis in recent years. Second, the chapter will discuss some important explainable artificial intelligence approaches and performance measures. Lastly, it will conduct an extensive literature review of recent studies conducted in the field.

2.2 Background

Anomaly detection in system logs has slowly evolved from simple rules and statistical methods to machine learning models. The reason for this change lies in the characteristics of the logs, which are usually big, sequential, and semi-structured. In practical applications, anomalies do not arise as singular occurrences but instead arise through variations in the sequence of events. This means that previous methodologies fail to detect them effectively, particularly in complicated distributed scenarios or cases with insufficient labeled datasets. For this reason, recent research has been centered around learning-based techniques that can appropriately model log sequences [4].

In this section, we discuss some of the major research directions in methodology which have become quite relevant for our work. First, we consider self-supervised learning since labeled anomalies are typically scarce in real-world problems. Next, sequence modeling with Transformers is discussed as it has been extensively used to incorporate long-term dependencies in logs. Moreover, several attempts to enhance log embeddings using language models have recently been made, along with methods that consider uncertainty and structures to make their model more robust and interpretable. These

research directions have become an integral part of building modern systems for detecting anomalies in logs [5].

2.2.1 Self-supervised learning

Self-supervised learning has emerged as a feasible approach for anomaly detection in logs, particularly when labeled data is scarce. Rather than depending on labeled anomalies, the model learns patterns based on the data using an auxiliary task. The model uses this learned pattern to represent normal behaviors and then detect anomalies.

2.2.2 Masked prediction

Prediction with masking as shown in Figure 2.1 is a prevalent technique in the field of self-supervised log analysis, in which parts of the sequence are masked out for the training process to recover them. In place of using explicit labels, the training process uses the relation between the events in order to identify patterns within the sequences. This technique allows the training algorithm to better understand the normal behavior of the system. Recent research has proven that the application of self-supervision can lead to improved detection rates in situations of insufficient labeling [6].

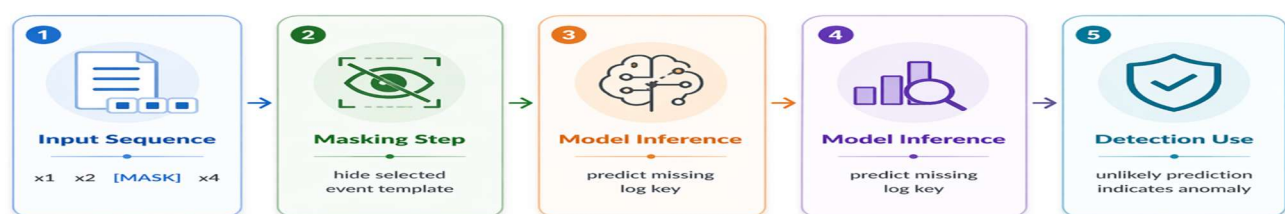


Figure 2.1 Masked log prediction as a self-supervised pretext task. [6]

2.2.2.1 Reconstruction-based learning

The reconstruction-based methods work a bit differently from the previous approaches. In contrast to filling in gaps, the model is trained to generate full sequences. Sequences that fail to be generated are marked as anomalies during testing. This technique has been

successfully utilized with the help of Transformer-based neural networks, which can model both local and long-range dependencies. However, these techniques could face difficulties distinguishing anomalies from ordinary sequences if they are quite alike [7].

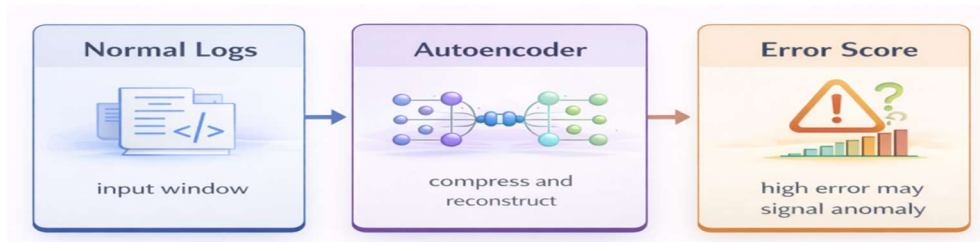


Figure 2.2 Reconstruction-based self-supervised learning for anomaly detection. [7]

2.2.2.2 Contrastive learning

The idea of contrastive learning is centered on increasing the quality of representation through differentiation of similar and dissimilar patterns. For detecting log anomalies, this implies training a model to discriminate between normal and abnormal sequences in the embedding space. As such, the sensitivity of the system to slight pattern differences is increased. However, while this method offers an improvement in the discriminatory power, it is highly dependent on sample definition [8]. An illustration of the above process is presented in Figure 2.3.

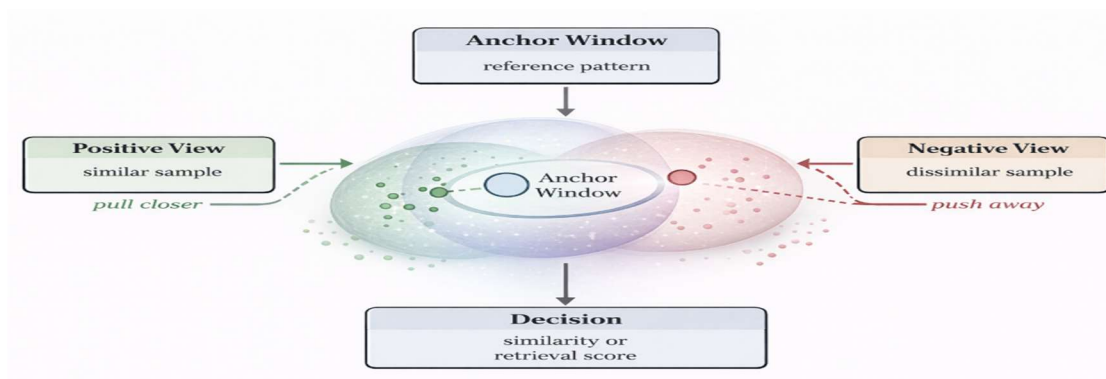


Figure 2.3 Contrastive representation learning for separating normal and abnormal log patterns. [8]

2.2.3 Transformer / self-attention sequence modeling

Transformer-based approaches are now common in log analysis since they are capable of learning dependencies in long sequential logs. While traditional approaches take data into account gradually, one by one, Transformer-based algorithms consider all the data in the log in one go due to their attention mechanisms.

2.2.3.1 Transformer modeling

Transformation Models use contextual embedding, where the embedding of an occurrence is affected by other events in the same log sequence. Consequently, complex relationships are learned by the transformation models. Several researches have demonstrated superior performance of Transformation Models compared to regular sequential modeling approaches for log analysis [9]. The basic modeling procedure is depicted in Figure 2.4.

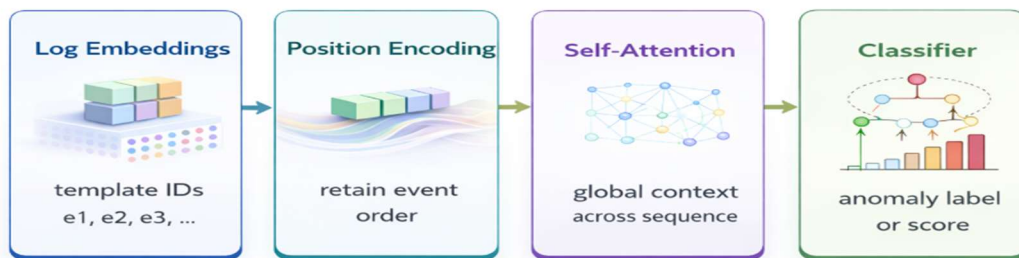


Figure 2.4 Encoder-based Transformer modeling for log anomaly detection. [9]

2.2.3.2 Hybrid Architectures

Other methods incorporate the Transformer architecture with other methodologies to increase efficiency. For instance, convolution operations can be applied to learn local dependencies, while Transformer networks learn global dependencies. Although this approach tends to offer improved performance, it adds complexity to the system

especially in the pre-processing phase and fine-tuning phase [10]. The overall structure of such models is shown in Figure 2.5.

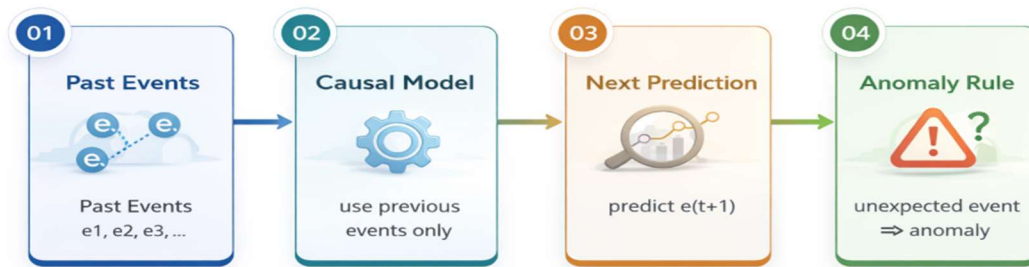


Figure 2.5 Hybrid architecture for log anomaly detection. [10]

2.2.4 LLM-enriched representations

Large language models have recently been used to enrich log events with semantic information before downstream anomaly detection. These methods are useful when raw log templates contain limited semantic meaning, and they are also relevant to explainable anomaly detection because they can generate richer contextual descriptions and operator-facing explanations for detected anomalies.

2.2.4.1 LLM semantic enrichment

In semantic enrichment, LLMs are used to transform raw log messages into more informative representations. This helps downstream models better understand the meaning of events, which improves anomaly detection performance. However, this step adds computational overhead to the system [11]. The enrichment process is illustrated in Figure 2.6.

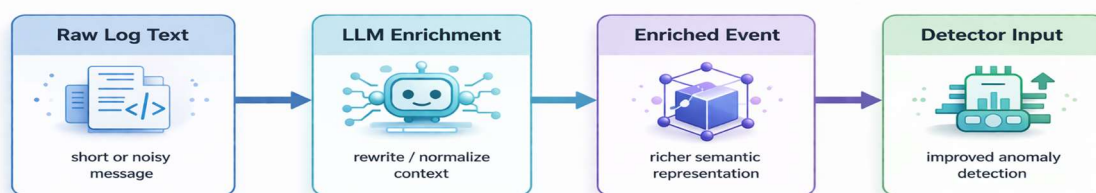


Figure 2.6 LLM-based semantic enrichment before downstream detection. [11]

2.2.4.2 LLM fine-tuning on normal logs

Another approach is to fine-tune LLMs directly on log data. This allows the model to learn patterns specific to system behavior. While this can improve performance, it requires significant computational resources and careful training [12]. The process is shown in Figure 2.7.

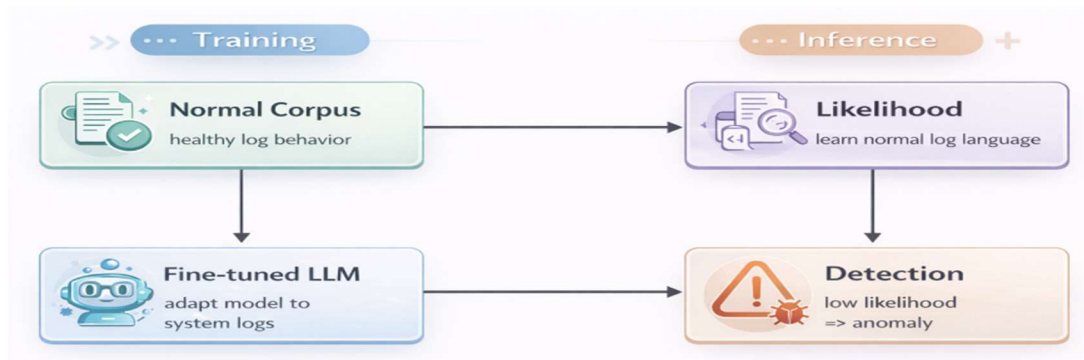


Figure 2.7 Fine-tuning a large language model on normal log behavior. [12]

2.2.4.3 LLM-assisted reasoning

Rather than using LLMs continuously, some frameworks apply them selectively for reasoning or explanation. This reduces computational cost while still benefiting from their capabilities. However, designing such systems can be complex [13]. This idea is illustrated in Figure 2.8.



Figure 2.8 LLM-assisted explanation for anomalous log sequences. [13]

2.2.5 Uncertainty-aware learning

Uncertainty-aware learning focuses on estimating not only whether an event sequence is anomalous, but also how confident the model is in that decision. This is particularly relevant in explainable anomaly detection because trustworthy outputs should communicate both the anomaly score and the model's confidence, especially in large-scale operational environments where ambiguous cases can trigger costly false alarms.

2.2.5.1 Confidence-aware anomaly scoring

These models provide both anomaly scores and confidence levels, helping distinguish between clear and uncertain cases. This is useful in real-world scenarios where incorrect decisions can be costly [14]. The process is shown in Figure 2.9.

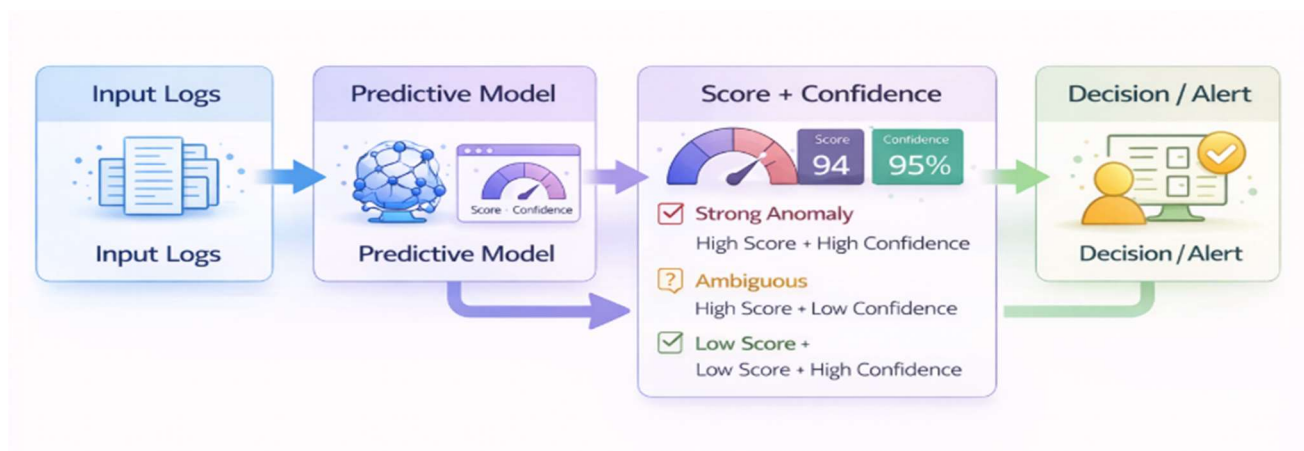


Figure 2.9 Confidence-aware anomaly scoring pipeline. [14]

2.2.5.2 Evidential deep learning

Evidential deep learning models uncertainty by estimating evidence distributions instead of single predictions. This approach improves robustness, especially in ambiguous situations, but does not directly explain which features contribute to decisions [15]. This concept is illustrated in Figure 2.10.

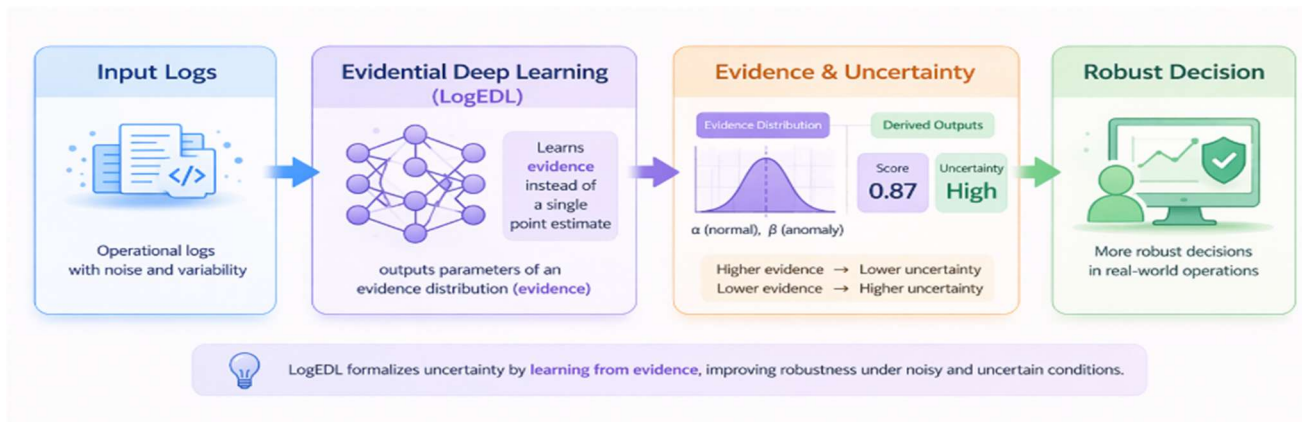


Figure 2.10 Evidential deep learning for uncertainty-aware log anomaly detection. [15]

2.2.6 Graph reasoning and hierarchical abstraction

Graph-based and hierarchical techniques allow anomaly detection on logs to move beyond simple sequences by identifying relationships between entities, services, and execution contexts. This kind of technique proves particularly effective when, besides time order, considerations related to topology and modular and nested execution become relevant, but it is also highly effective at increasing explainability by structuring the detected anomalies around execution units.

2.2.6.1 Graph-based modeling

Graph-based methods represent logs as networks of interacting components. This allows models to capture relationships between services and system components more effectively. However, building and processing such graphs can be computationally expensive [16]. The structure is shown in Figure 2.11.

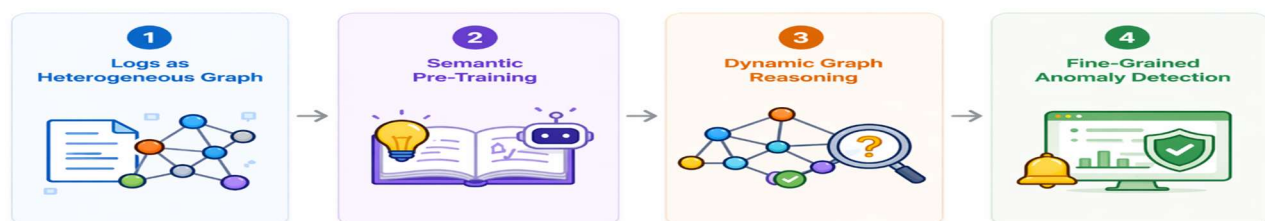


Figure 2.11 Graph-based modeling of structural relationships in log data. [16]

2.2.6.2 Hierarchical abstraction

Hierarchical models organize log events in higher structures, leading to better interpretability and thus improved detection effectiveness. However, the hierarchical model also adds more processing stages to achieve this purpose [17]. The hierarchical abstraction method is shown in Figure 2.12.

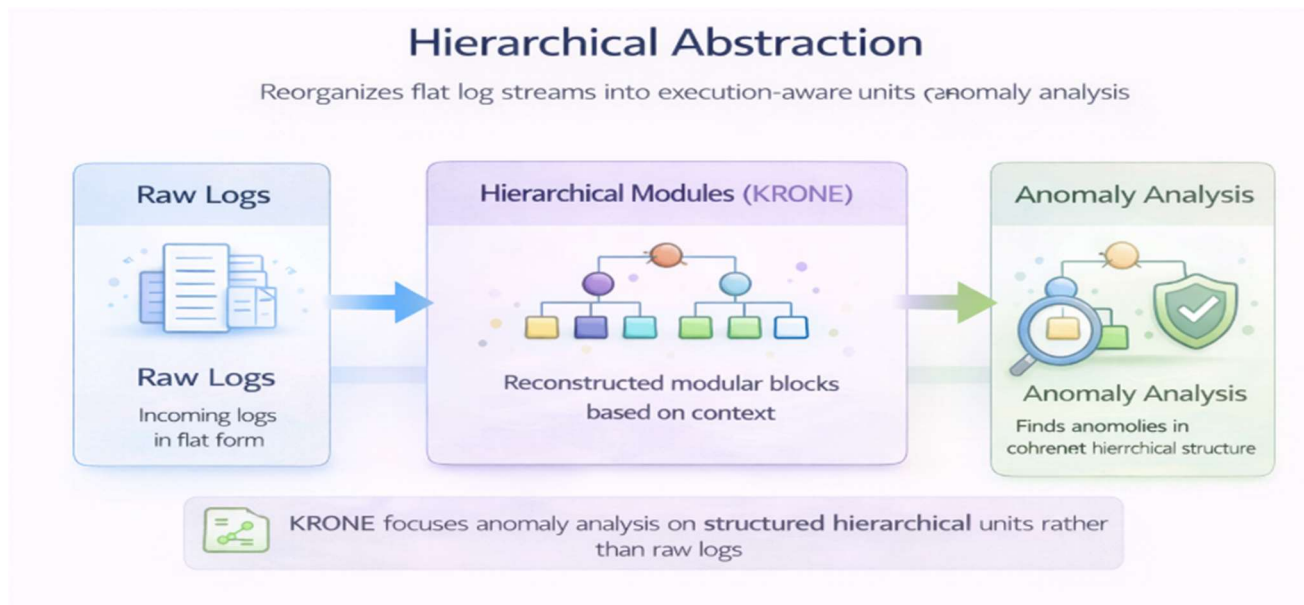


Figure 2.12 Hierarchical abstraction from flat log streams to execution-aware units. [17]

2.2.6.3 Modular detection pipelines

The modular approach breaks down the detection process into phases and uses complex modules only when necessary. This enables a compromise between efficiency and effectiveness but entails thorough planning of the system [18]. Figure 2.13 depicts a general workflow. A typical pipeline is shown in Figure 2.13.

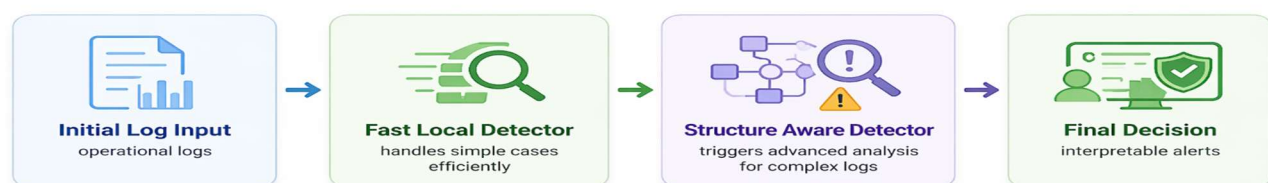


Figure 2.13 Modular detection framework with escalation to structure-aware reasoning. [18]

2.3 Explainable AI (XAI)

The emergence of XAI has been driven by the need to gain insights into the processes behind the model's actions. In log anomaly detection, this aspect plays a crucial role since the operator's interest does not lie in detecting the anomaly but in the very reason behind its occurrence. The absence of these features could make the system untrustworthy and difficult to operate.

In the recent scientific literature, the development of model-agnostic methods has gained significant traction because they do not require any changes to be introduced to the underlying model. They are therefore highly applicable to neural networks that employ deep learning techniques. In the current study, we will investigate the use of two popular methods – SHAP and LIME – in interpreting log anomalies.

2.3.1 SHAP

Another popular framework for interpreting complicated models is SHAP (SHapley Additive exPlanations). SHAP uses principles from cooperative game theory in which contributions are made by individual players to the final result. In the case of SHAP, this means that individual features contribute to the output of the model. Unlike other methods, SHAP estimates feature contributions considering all possible combinations of those features. This ensures consistency of the resulting explanation.

Applying SHAP to the problem of log anomaly detection will help identify particular log events or tokens that caused a certain anomaly score. It can be considered an advantage of the method since, in practice, we would like to understand which specific behaviors of the system led to an anomaly. In addition to being intuitive, SHAP satisfies additional properties such as consistency and local accuracy making its explanations more robust than simple methods.

The downside of SHAP is the computational cost associated with estimating SHAP values. When dealing with very long logs, SHAP computations become costly because we have to account for all feature combinations. Nevertheless, SHAP is still a solid choice for anomaly detection since it guarantees stability of the explanation. The overall idea is shown in Figure 2.14.

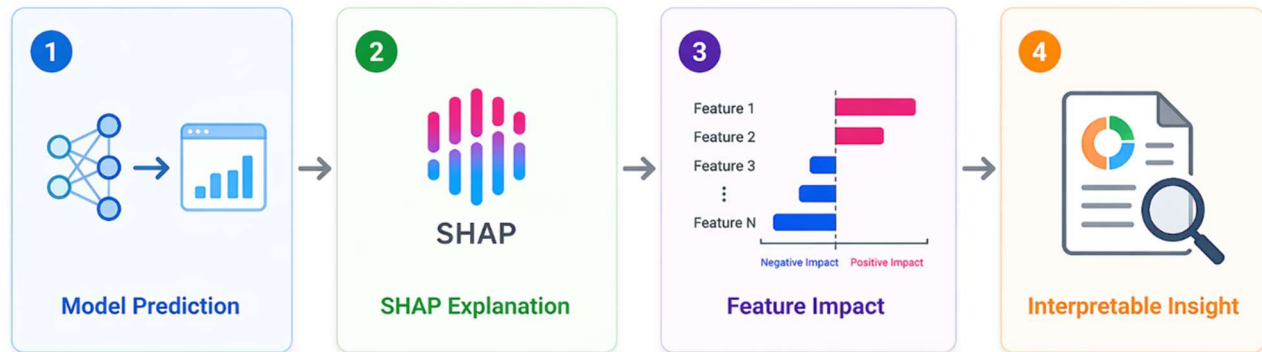


Figure 2.14 SHAP (SHapley Additive exPlanations). [19]

2.3.2 LIME

LIME (Local Interpretable Model-agnostic Explanations) is an explainability technique used after the model creation, which creates an approximation to the complex black-box model's behavior locally by a simple, easy-to-understand model. Specifically, LIME works by sampling perturbed inputs in the region around the initial sample and observing how the output of the model changes, then fitting a weighted linear model to understand which features affect the model's decision the most as shown in Figure 2.15. In log anomaly detection, LIME can provide insights into what log events or tokens influence the specific prediction made by the model. The flexibility of LIME comes from its ability to approximate locally and make use of any interpretability method, which allows it to be used with any kind of machine learning model, even the Transformer-based models. However, due to its dependence on perturbation sampling and local approximation, LIME tends to be unstable for complex structured inputs like

logs, as changing individual features can alter the meaning of the whole input drastically. Overall, while LIME can be a valuable tool for fast and simple explanation of local decisions, it tends to be less stable than SHAP and cannot be reliably used with sequential log data [20].

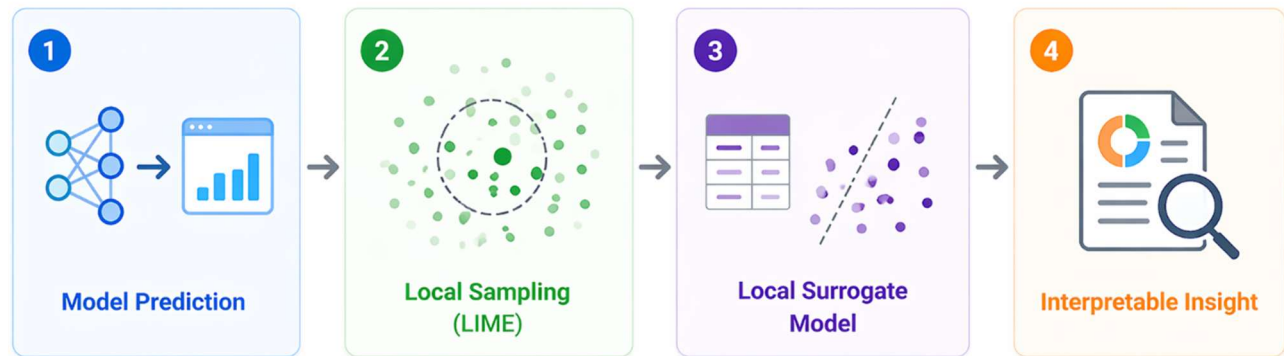


Figure 2.15 LIME (Local Interpretable Model-agnostic Explanations). [20]

2.4 Performance Metrics

Anomaly detection models are often evaluated using classification-based metrics, which measure the accuracy of the model in identifying abnormal and normal examples. In the case of log anomaly detection, the problem is often treated as a binary classification problem, whereby each example is classified as either a normal example or an abnormal example. Since the log data set is highly unbalanced, meaning that there are very few abnormal examples compared to normal examples, it is important to use a metric to evaluate the model performance.

Let TP stand for true positives (accurately predicted abnormal examples), let FP stand for false positives (incorrectly classified as abnormal but are actually normal), let FN stand for false negatives (missed abnormal examples by the algorithm), and let TN stand for true negatives (correctly predicted normal examples).

2.4.1 Precision

If a machine learning model has high precision, then its rate of generating false alarms will be low. This attribute is very significant because, in an operational setting, false alarms can lead to problems for the operator of the system [21].

Precision measures the proportion of detected anomalies that are actually anomalous and is defined as:

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \quad (1)$$

2.4.2 Recall

A good recall value suggests that the model can capture most of the anomalies, minimizing the chance of failing to detect an anomaly. When dealing with safety-sensitive applications, recall tends to be favored [22].

Recall measures the proportion of actual anomalies that are correctly detected by the model and is defined as:

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \quad (2)$$

2.4.3 F1-score

It is commonly applied to log anomaly detection since it considers both false alarms and misses, making it applicable to cases where the data distribution is unbalanced. It offers a single measure of performance that balances detection accuracy and reliability [23].

The F1-score provides a balanced measure of precision and recall and is defined as:

$$\text{F1-score} = 2 \times [(\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})] \quad (3)$$

2.4.4 Accuracy

Even though accuracy is often employed in practice, its use for evaluating log anomalies could be deceptive because of class imbalance when there are many more examples of the majority class. This is why accuracy is evaluated in conjunction with precision, recall, and F1-score metrics [24].

It measures the overall proportion of correctly classified instances and is defined as:

$$\text{Accuracy} = (TP+TN) / (TP+TN+FP+FN) \quad (4)$$

2.5 Literature Review

Lee et al. [25] Presented their work on LAnoBERT, a log anomaly detector that uses masked language modeling without requiring any log parser. Their method does not require the use of any log parser but is able to produce impressive results on benchmark log datasets such as HDFS and BGL. One downside of their approach is that it uses expensive Transformer-based models.

Zhao and Zhang [26] presented LADB, which uses representation learning with BERT in conjunction with Deep SVDD for modeling and anomaly detection as deviation. LADB enhances the discrimination capability of normal and abnormal behaviors; however, there is no complete reporting of performance metrics.

Shi et al. [27] proposed LogFold, a method that applies sequence reconstruction to reduce log sequences prior to the detection of anomalies. They achieved high performance on benchmark data, particularly on BGL, but at the cost of losing some fine-grained temporal data in the process of compression.

Shi et al. [28] developed an approach for pre-training Transformers through masked log reconstruction using unlabeled data at scale. The method performed impressively well

in settings with insufficient labeling due to its ability to learn generalizable logs. However, although efficient, the technique is not interpretable.

Li et al. [29] proposed a contrastive learning paradigm with retrieval-based enhancement to detect anomalies in logs. The approach aims to boost the quality of representations and separate normal patterns from abnormal ones. Nevertheless, the technique is geared towards optimizing anomaly detection accuracy but lacks explanation capability.

Liu et al. [30] proposed a hybrid model that combines the Sentence-BERT encoding for semantic features with the CNN for extracting features from the encoded sequences and Transformers for sequential modeling. The proposed technique produced remarkable results in the form of F1-scores close to 0.98 when applied to the BGL dataset. However, the model is highly sensitive to preprocessing.

Han et al. [31] introduced LogGPT, a generative Transformer model that treats log sequences as language data and predicts future events for anomaly detection. The method captures complex sequential patterns and shows strong performance across multiple datasets. However, it requires high computational resources and lacks direct feature-level interpretability.

Yang and Harris [32] proposed LogLLaMA which fine-tunes and employs reinforcement learning to train a large language model for detecting log anomalies. This model performed better than previous techniques when evaluated using HDFS, BGL, and Thunderbird data sets. Even though it performs well, it imposes high computation costs.

Shi et al. [33] used Anomaly Transformer for detecting anomalies in system logs using attention-based sequence modeling to find dependencies between logs. The technique outperformed the results achieved by LSTM-based approaches and rule-based systems

on benchmark data sets. Attention alone cannot be used for generating valid explanations of a model's choices.

Guo et al. [34] proposed LogFormer, a different framework for transfer learning that utilizes both pre-training and adapter fine-tuning to detect anomalies in cross-domain logs. This model ensures better generalization while keeping fewer training parameters. Nonetheless, its performance relies on the performance of the source domain.

He et al. [35] Proposed the LLMeLog method that improves log representation through the use of semantics obtained from LLMs. This method resulted in F1-scores greater than 99% for several test cases. However, this introduces complexity and increased computation costs into the system.

Ma et al. [36] proposed KRONE, which developed by proposing a hierarchical and modular framework for detecting anomalies, which reformats the logs into execution-aware pieces. This solution increased the F1-score from 82.76% to 92.83%, with less dependence on massive language models. Yet, this approach adds to complexity because of its multi-level nature.

Duan et al. [37] Proposed LogEDL: An uncertainty-aware framework for anomaly detection based on evidential deep learning. The framework enhances robustness through prediction confidence estimation and ambiguity management. It does not, however, offer feature-level interpretations for anomaly detection.

Qian et al. [38] proposed HEDGE, a graph-based framework that models logs as dynamic heterogeneous graphs to capture structural relationships between system components. The approach improves fine-grained anomaly detection in complex systems. However, graph construction and processing introduce significant computational overhead.

It can be observed from Table 1 that recent researches have made substantial progress in enhancing the anomaly detection effectiveness by applying Transformer models, contrastive learning, and semantic augmentation methods. Nevertheless, most existing studies concentrate mainly on the detection efficiency, with little regard for the interpretability aspect of anomaly decision-making.

Table 1. Comparative summary of reviewed studies

Author	Year	Dataset	Methodology	Key Findings	Limitations
Lee et al. [25]	2023	HDFS, BGL, Thunderbird	BERT-based masked language modeling (LAnoBERT)	Improved detection without log parsing	High computational cost
Zhao and Zhang [26]	2024	HDFS	BERT + Deep SVDD (LADB)	Strong anomaly discrimination	Limited reported metrics
Shi et al. [27]	2023	HDFS, BGL	Transformer-based reconstruction (LogFold)	F1 = 90.55% / 99.90%	Loss of fine temporal details
Shi et al. [28]	2024	Multiple datasets	Transformer pre-training	Effective with limited labels	Low interpretability
Li et al. [29]	2025	HDFS, BGL	Contrastive learning + retrieval	Improved anomaly separation	No explanation mechanism
Liu et al. [30]	2024	BGL	Sentence-BERT + Transformer	F1 $\approx$ 0.98	Sensitive to preprocessing
Han et al. [31]	2023	Multiple datasets	Generative Transformer (LogGPT)	Captures complex patterns	High computational cost
Yang and Harris [32]	2025	HDFS, BGL, Thunderbird	LLM fine-tuning (LogLLaMA)	State-of-the-art performance	Very high cost
Shi et al. [33]	2024	HDFS	Anomaly Transformer	Better than LSTM models	Attention not reliable
Guo et al. [34]	2024	Cross-domain logs	Transfer learning (LogFormer)	Improved generalization	Depends on source data
He et al. [35]	2024	Public datasets	LLM enrichment (LLMeLog)	F1 > 99%	High complexity

Ma et al. [36]	2026	Multi-dataset	Hierarchical modeling (KRONE)	Improved F1 + structure	Complex pipeline
Duan et al. [37]	2024	HDFS, BGL	Evidential deep learning (LogEDL)	Improved robustness	No feature explanation
Qian et al. [38]	2024	Service logs	Graph-based modeling (HEDGE)	Fine-grained detection	High complexity

2.6 Chapter Summary

The chapter provides an overview of the major concepts and latest advancements in the area of log anomaly detection. It explains that contemporary methodologies for log anomaly detection are based on machine learning algorithms, especially self-supervised learning and Transformer models. Furthermore, it explains explainable methods like SHAP and LIME, and some standard measures for evaluating the performance of anomaly detection algorithms.

Moreover, the literature review pointed out that although the majority of existing research studies have successfully achieved highly accurate detections, there is limited literature focusing on the explanation aspect of the system. The aforementioned challenge thus necessitates the creation of an approach incorporating both aspects. With the aforementioned deliberation, the following chapter discusses the proposed framework.

Chapter 3 Proposed Approach

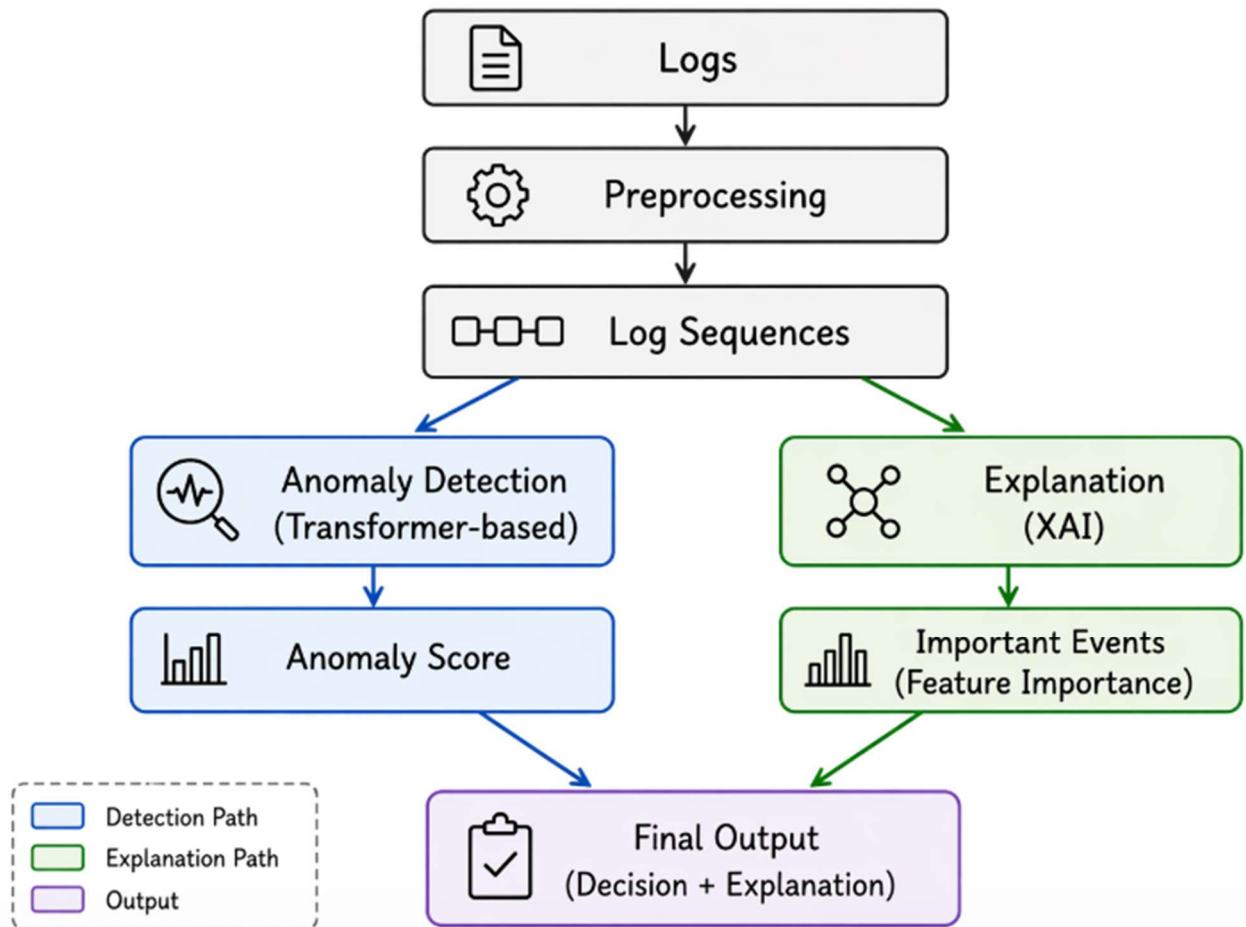


Figure 3.1 Dual-flow explainable framework for log anomaly detection.

Chapter 4 Experimental Results and Discussion

Chapter 5 Conclusion and Future Work

References

- [1] X. Wang, K. J. Kim, Y. Wang, T. Koike-Akino, and K. Parsons, "DeepEAD: Explainable anomaly detection from system logs," Mitsubishi Electric Research Laboratories, Tech. Rep., 2023.
- [2] Y. Xie, H. Zhang, and M. A. Babar, "LogSD: Detecting anomalies from system logs through self-supervised learning," ACM Trans. Softw. Eng., 2024.
- [3] X. Ma, Y. Li, J. Keung, X. Yu, H. Zou, Z. Yang, F. Sarro, and E. T. Barr, "Practitioners' expectations on log anomaly detection," IEEE Trans. Softw. Eng., 2025.
- [4] H. Zhang, Y. Liu, and X. Chen, "Learning-based log anomaly detection for large-scale systems," IEEE Access, 2023.
- [5] Y. Chen, L. Zhang, and J. Wang, "Advances in intelligent log analysis for distributed systems," Future Gener. Comput. Syst., 2024.
- [6] Y. Zhang, X. Li, and Q. Wang, "Masked log modeling for anomaly detection," in Proc. IEEE Int. Conf. Big Data, 2023.
- [7] L. Chen, Z. Wang, and Y. Liu, "Transformer-based reconstruction for log anomaly detection," IEEE Access, 2024.
- [8] Z. Wang, H. Liu, and Y. Zhang, "Contrastive learning for log representation," Neurocomputing, 2024.
- [9] X. Li, J. Chen, and Y. Wang, "Transformer-based sequence modeling for log anomaly detection," Inf. Sci., 2023.
- [10] Y. Liu, J. Wang, and T. Yan, "Hybrid deep learning models for log anomaly detection," Expert Syst. Appl., 2024.
- [11] M. He, T. Jia, and C. Duan, "LLM-based semantic enrichment for log anomaly detection," in Proc. IEEE ISSRE, 2024.

- [12] Z. Yang, I. G. Harris, and X. Liu, “Fine-tuning large language models for log analysis,” arXiv preprint arXiv:2501.XXXXX, 2025.
- [13] L. Ma, J. Liu, and T. Zhang, “Modular LLM-assisted anomaly reasoning,” arXiv preprint arXiv:2601.XXXXX, 2026.
- [14] Q. Zhang, Y. Wang, and H. Liu, “Confidence-aware anomaly detection in distributed systems,” IEEE Access, 2024.
- [15] Y. Duan, H. Li, and X. Zhao, “Evidential deep learning for anomaly detection,” Appl. Sci., 2024.
- [16] B. Qian, M. Zhu, and E. Wu, “Graph-based log anomaly detection using dynamic structures,” in Proc. IEEE ICWS, 2024.
- [17] L. Ma, P. VanNostrand, and J. Chen, “Hierarchical log modeling for anomaly detection,” IEEE Trans. Knowl. Data Eng., 2026.
- [18] J. Chen, L. Zhang, and X. Li, “Modular pipelines for scalable anomaly detection,” Future Gener. Comput. Syst., 2025.
- [19] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Adv. Neural Inf. Process. Syst. (NeurIPS), 2017.
- [20] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘Why should I trust you?’ Explaining the predictions of any classifier,” in Proc. ACM SIGKDD, 2016.
- [21] J. Davis and M. Goadrich, “The relationship between Precision-Recall and ROC curves,” in Proc. ICML, 2006.
- [22] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers,” PLOS ONE, vol. 10, no. 3, 2015.
- [23] D. M. W. Powers, “Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation,” J. Mach. Learn. Technol., 2011.

- [24] D. Chicco and G. Jurman, “The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,” *BMC Genomics*, 2020.
- [25] Y. Lee, J. Kim, and P. Kang, “LAnoBERT: System log anomaly detection based on BERT,” *Appl. Soft Comput.*, 2023.
- [26] R. Zhao and M. Zhang, “BERT-based log anomaly detection using Deep SVDD,” *Appl. Comput. Eng.*, 2024.
- [27] X. Shi, R. Li, Q. Du, C. He, and F. Tian, “LogFold: Sequence folding for log anomaly detection,” in *Proc. APSEC*, 2023.
- [28] L. Shi, J. Ren, W. Wang, Y. Liu, and L. Sun, “Transformer pre-training for log anomaly detection,” in *Proc. IEEE ICC*, 2024.
- [29] W. Li et al., “Contrastive learning with retrieval augmentation for anomaly detection,” *Sci. Rep.*, 2025.
- [30] Y. Liu, J. Wang, and T. Yan, “Sentence-BERT and Transformer-based log anomaly detection,” in *Proc. ISGC*, 2024.
- [31] X. Han, S. Yuan, and M. Trabelsi, “LogGPT: Log anomaly detection via generative Transformer,” *arXiv preprint arXiv:2305.XXXXXX*, 2023.
- [32] Z. Yang and I. G. Harris, “LogLLaMA: Transformer-based log anomaly detection using LLMs,” *arXiv preprint arXiv:2503.XXXXXX*, 2025.
- [33] Q. Shi, Z. Yang, A. Haidar, and F. Li, “Anomaly Transformer for log anomaly detection,” in *Proc. IEEE CARS*, 2024.
- [34] H. Guo, J. Yang, J. Liu, and B. Wang, “LogFormer: A pre-train and tuning pipeline for log anomaly detection,” *arXiv preprint arXiv:2402.XXXXXX*, 2024.

- [35] M. He, T. Jia, C. Duan, H. Cai, Y. Li, and G. Huang, “LLMeLog: An approach for anomaly detection based on LLM-enriched logs,” in Proc. IEEE ISSRE, 2024.
- [36] L. Ma et al., “KRONE: Hierarchical and modular log anomaly detection framework,” arXiv preprint arXiv:2602.XXXXXX, 2026.
- [37] Y. Duan et al., “LogEDL: Uncertainty-aware anomaly detection,” Appl. Sci., 2024.
- [38] B. Qian et al., “HEDGE: Heterogeneous dynamic graph framework for log anomaly detection,” in Proc. IEEE ICWS, 2024.