2025 32nd Asia-Pacific Software Engineering Conference (APSEC) | 979-8-3315-6653-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/APSEC66846.2025.00054

Understanding Industrial Log Analysis: A Multi-Dataset Evaluation of Parsing and Anomaly Detection

Yicheng Sun, Jacky Keung, Yihan Liao, and Hi Kuen Yu*
Department of Computer Science, City University of Hong Kong, Hong Kong, China
yicsun2-c@my.cityu.edu.hk, jacky.keung@cityu.edu.hk, yihanliao4-c@my.cityu.edu.hk, hikuenyu2-c@my.cityu.edu.hk
*Corresponding author

***Abstract*—Log analysis plays a critical role in monitoring and maintaining the safety of industrial software systems. However, most existing research relies heavily on benchmark datasets derived from legacy or open-source systems, which fail to capture the structural diversity and operational complexity of real-world industrial logs. In this study, we present a comprehensive empirical evaluation of log parsing and anomaly detection models across four diverse datasets, including three collected from large-scale industrial software deployed in manufacturing, process control, and energy monitoring environments. Our analysis reveals that state-of-the-art models—particularly rule-based parsers and supervised detectors—experience substantial performance degradation when applied to industrial settings. To address this gap, we introduce a unified evaluation framework using representative training subsets, and we highlight the effectiveness of semi-supervised and LLM-based approaches in handling heterogeneous, low-resource log environments. The findings offer practical insights into the limitations of current log analysis techniques and suggest design principles for building more robust, domain-adaptive solutions for industrial software risk mitigation.**

***Index Terms*—Industrial Software, Log Parsing, Anomaly Detection, Semi-Supervised Learning, Large Language Models**

I. Introduction

Log data has become an essential asset in maintaining the safety, stability, and reliability of industrial software systems [1]–[3]. Generated automatically during system execution, logs provide valuable records of internal operations, offering insights into system behaviors, failures, and potential risks [4]–[6]. As software continues to increase in complexity and scale, the role of automated log analysis—especially log anomaly detection—has become increasingly critical for timely risk mitigation and operational resilience [7]–[9].

A significant body of research on log anomaly detection has relied on publicly available datasets, most notably those hosted on LogHub [10]. While these datasets have served as important benchmarks, they exhibit several limitations [11]. One significant limitation of LogHub is that it consists primarily of logs collected from systems over a decade old, many of which follow relatively consistent and uniform logging formats [12]–[14]. As a result, models that perform well on these datasets may not generalize to contemporary industrial software systems, which are increasingly complex, modularized, and deployed in heterogeneous environments.

To better understand the characteristics and challenges of modern industrial logs, we conducted in-depth, one-on-one interviews with seven software maintenance engineers from large-scale industrial organizations across sectors such as energy, automation, and manufacturing. The engineers reported that log data in modern industrial software systems tends to be significantly more diverse and complex than that of traditional open-source systems. Logs are often verbose, contain proprietary terminologies, and are deeply tied to both hardware and software subsystems. Moreover, logging practices vary widely across components, introducing inconsistencies in format, granularity, and semantic structure.

To quantify these observations, we compared logs from two relatively complex LogHub datasets—BGL and Thunderbird—with five anonymized industrial systems provided by our collaborators. As shown in Table I, BGL and Thunderbird have average log lengths of 197 and 205 tokens, respectively, and 1,531 and 2,732 unique log templates. In contrast, logs from the industrial systems exhibit substantially greater complexity, with average lengths ranging from 342 to 428 tokens and template counts exceeding 4,000 in all cases. These differences highlight the unique structural and semantic challenges inherent in processing industrial logs.

TABLE I: Comparison of log statistics between LogHub datasets and industrial systems.

System	Avg. Log Length (tokens)	# Unique Templates	Total Log Count	Source
BGL	197	1,531	4,347,984	LogHub
Thunderbird	205	2,732	2,878,984	LogHub
Industrial-1	342	4,508	8,123,410	Confidential
Industrial-2	376	5,121	9,504,822	Confidential
Industrial-3	428	6,744	10,274,915	Confidential
Industrial-4	393	5,612	7,893,143	Confidential
Industrial-5	411	6,039	8,681,777	Confidential

Despite these evident challenges, the majority of existing studies still benchmark log analysis models solely on LogHub datasets, overlooking the more stringent and heterogeneous demands of industrial software environments. In contrast to cloud or web-based systems, industrial software is characterized

979-8-3315-6653-1/25/$31.00 ©2025 IEEE
DOI 10.1109/APSEC66846.2025.00054

by tightly coupled hardware-software interactions, domain-specific error codes, and highly context-dependent execution behaviors. These factors collectively exacerbate the difficulty of anomaly detection, necessitating advanced techniques that can handle both structural diversity and semantic ambiguity with greater robustness.

To address this research gap, we collaborated with industrial partners to collect, preprocess, and anonymize logs from three large-scale industrial software systems. These logs were curated into benchmark-quality datasets, designed to preserve the complexities observed in real-world industrial environments. We then conducted a series of experiments to evaluate both parsing and anomaly detection performance across the three industrial datasets and one representative LogHub dataset. By conducting these cross-domain comparisons, we aim to assess the generalizability of current models and identify limitations that hinder their deployment in safety-critical industrial settings. In summary, our primary contributions can be summarized in three key points:

- **We present the first comprehensive empirical study of log analysis in modern industrial software systems.** Unlike prior work focused primarily on open-source or legacy systems, our study investigates real-world logs collected from three large-scale industrial environments in manufacturing, process automation, and energy monitoring. These logs exhibit greater structural variability, semantic complexity, and operational coupling, providing a more realistic basis for evaluating log analysis models in high-stakes industrial contexts.
- **We conduct a multi-dataset evaluation of state-of-the-art log parsing and anomaly detection techniques.** Using standardized, representative subsets from both industrial and benchmark datasets, we assess the generalizability and robustness of rule-based, supervised, semi-supervised, and LLM-based approaches. Our results reveal substantial performance degradation when models trained on traditional datasets are applied to industrial logs, highlighting critical gaps in current model adaptability.
- **We identify key challenges and design implications for building industrial-grade log analysis systems.** Through quantitative error analysis and expert-informed case studies, we expose frequent parsing failures, semantic omissions, and false detections caused by industrial-specific log characteristics. These findings underscore the need for adaptive, context-aware, and low-supervision log analysis frameworks to support reliable risk detection and operational decision-making in real-world industrial software systems.

II. Study Design

A. Scenarios and Dataset Construction

To ensure a realistic and comprehensive evaluation of log parsing and anomaly detection models in industrial software environments, we constructed four datasets that reflect diverse system architectures, deployment contexts, and logging practices. These datasets cover both well-established benchmarks and real-world industrial systems from multiple sectors, enabling the assessment of model generalizability across traditional and modern log sources. In particular, the industrial datasets were collected in collaboration with domain experts and capture a wide range of operational characteristics, such as hardware-software interaction, dynamic logging templates, and domain-specific terminologies. This diversity is critical for understanding how existing techniques perform under the complexities of industrial log data.

- **LHSB**: A curated subset of log data selected from the LogHub platform [10], widely recognized as a benchmark for log analysis research. This dataset serves as a controlled baseline for evaluating model performance under standard conditions.
- **FALL**: Derived from a low-resource industrial assembly line system, this dataset captures logging behaviors in high-precision manufacturing environments. It is characterized by relatively sparse log frequency, irregular message formatting, and high variability—conditions common in embedded or cost-constrained deployments.
- **PIMS**: Collected from a modern process industry management system in the chemical sector, this dataset contains high-volume logs with deeply nested event sequences and frequent interactions between software modules and physical sensors. These logs exhibit rich temporal dynamics and highly domain-specific terminologies.
- **SCADA-X**: Obtained from a supervisory control and data acquisition (SCADA) system used in energy infrastructure. This dataset includes logs from distributed control nodes and illustrates the complexity of multi-source, real-time system monitoring in safety-critical environments. Logs are heterogeneous in structure and often contain timestamp synchronization issues and context-sensitive event patterns.

Table II summarizes the key characteristics of each dataset, including log volume, average sequence length, template diversity, and domain application. For each dataset, we also constructed a structurally representative subset, which serves two primary purposes: (1) facilitating efficient log parsing by reducing redundant or low-information entries, and (2) providing a unified and balanced training set for subsequent anomaly detection models. These subsets are curated to preserve the structural and semantic diversity of the full datasets while enabling fair and consistent evaluation across different log analysis tasks.

1) LHSB Dataset: For the LHSB dataset, our objective was to construct a raw log corpus that accurately reflects the structural complexity and event distribution characteristic of real-world system operations. While the LogHub platform provides 2,000 manually curated log entries per dataset, these subsets often exclude rare anomalies, particularly those occurring within short intervals or under limited conditions. To overcome

TABLE II: Summary of constructed datasets.

Dataset	Abbr.	Type	Total Logs	Event Template	Sub. Logs	Anom.	Avg. Len (Chars)
LogHub Subset Benchmark	**LHSB**	Benchmark	**1,437,705** (6.87% anom.)	2,472	9,040	964 (10.7%)	172.4
Factory Assembly Line Logs	**FALL**	Industrial	**32,615** (9.56% anom.)	2,290	7,190	992 (13.8%)	227.5
Process Industry Mgmt. Sys.	**PIMS**	Industrial	**871,243** (9.93% anom.)	4,508	14,382	1,249 (8.7%)	312.6
Supervisory Control Logs	**SCADA-X**	Industrial	**902,377** (12.37% anom.)	6,039	15,110	1,013 (6.7%)	351.1

this limitation, we selected five LogHub [10] datasets—HDFS [15], Hadoop [16], OpenStack [17], BGL [18], and Thunderbird [18]—all of which include full log traces accompanied by ground-truth anomaly labels. To facilitate anomaly detection and log parsing evaluation under realistic conditions, we constructed a representative anomaly-centric subset through a two-stage strategy: (1) anomaly interval-based extraction and (2) dataset integration with a controlled anomaly ratio.

In the first stage, we extracted all log entries that occurred within the labeled anomalous intervals provided in the LogHub benchmark. For each dataset, the associated metadata files were parsed to identify the start and end timestamps of each abnormal window. We selected all log entries that occurred within a 90-second window before and after each annotated anomaly event. All log lines falling within these intervals were retained in their original format, ensuring comprehensive coverage of both frequent and infrequent anomalies that may be underrepresented in the original benchmark subsets. This procedure resulted in a total of 1,437,705 raw log entries, which serve as the test set for anomaly detection. Furthermore, we separately extracted the specific anomalous entries within these intervals to be used in the second stage.

In the second stage, we constructed a representative dataset by combining previously extracted anomaly logs with 10,000 randomly sampled entries from five selected LogHub benchmark subsets. To reflect the class imbalance typical in real-world environments, we reduced redundant normal logs and oversampled rare anomalies, resulting in an approximate 9:1 ratio of normal to anomalous entries. Due to missing anomaly labels, the remaining LogHub datasets were excluded. The final dataset contains 6,940 raw logs, selected without any template-based parsing or post-processing. This subset serves both as a challenging benchmark for log parsers and a unified training set for downstream anomaly detection models.

2) FALL: Low-Resource Industrial Logs from Assembly Line Systems: The FALL dataset represents a class of low-resource industrial environments that are often underrepresented in mainstream log analysis research. In such systems, log volumes are relatively limited compared to large-scale cloud infrastructure, yet the structural and semantic complexity of the logs is significantly higher due to tight integration between hardware components and real-time control mechanisms. FALL was collected from a real-world robotic assembly line operating in a high-precision manufacturing setting. The system includes multiple programmable logic controllers (PLCs), robotic arms, sensor arrays, and embedded software modules. These components collaboratively execute coordinated tasks such as material transport, positioning, and quality inspection. Logs are generated from both software control layers and physical hardware events, encompassing robotic joint positions, motion feedback, execution delays, fault alerts, and sensor threshold violations.

Over a continuous 20-day production period in October 2024, we collected a total of 32,615 raw log entries. Due to the inherent redundancy in system polling and status reports, we applied frequency-aware filtering to remove repetitive low-entropy entries, resulting in a curated subset of 7,190 log messages. These retained entries exhibit high diversity in structure, including unstructured textual errors, parameterized control signals, and hybrid-format sensor streams. Such variability poses significant challenges for conventional log parsers and anomaly detection models, particularly those trained on more uniform datasets like LogHub.

Importantly, FALL serves as a representative dataset for evaluating log analysis techniques in resource-constrained industrial deployments, such as early-stage automation systems, edge computing nodes, or embedded industrial controllers. Its inclusion allows us to explore how current approaches generalize beyond high-volume cloud logs to realistic, small-scale, heterogeneous industrial settings.

3) PIMS: Logs from Process Industry Management Systems: The PIMS dataset was collected from a large-scale industrial automation platform used in the chemical manufacturing sector. These systems typically operate continuously and are responsible for orchestrating complex process flows involving temperature regulation, chemical mixing, and pressure control, often under strict safety and quality constraints. The PIMS environment consists of a distributed architecture integrating industrial control software, batch process schedulers, human-machine interface (HMI) modules, and field-level sensor networks. Logs are generated from various layers, including control logic modules, sensor event streams, alert management subsystems, and operator interaction logs. These logs capture critical operational states such as valve openings, reactor conditions, control loop anomalies, and overridden safety procedures.

Over a one-month production window in November 2024, we collected 871,243 raw log entries from the active system. After removing low-informative periodic messages and normalizing timestamp formats, we constructed a refined dataset of 14,382 structurally representative log entries. The dataset features highly nested parameter structures, multi-source event interleaving, and extensive use of domain-specific codes, many of which are not documented in public manuals. These characteristics present unique challenges to existing parsing tools, particularly with respect to template extraction and semantic

segmentation.

The PIMS dataset represents a high-throughput, mission-critical industrial scenario in which accurate log analysis is essential for early fault detection and regulatory compliance. Its inclusion in our benchmark enables evaluation of log analysis models under conditions of high event density, overlapping system activities, and rich domain semantics.

4) SCADA-X: Logs from Supervisory Control and Data Acquisition Systems: The SCADA-X dataset originates from a Supervisory Control and Data Acquisition (SCADA) system deployed in a national-scale energy distribution network. SCADA systems are foundational to critical infrastructure sectors such as electricity, water, and gas, enabling real-time monitoring and control of geographically distributed assets, including substations, transformers, and load balancers. This particular SCADA system integrates multiple subsystems, including remote terminal units (RTUs), programmable automation controllers, historian servers, and alarm processing engines. Logs are produced through continuous telemetry, status polling, fault propagation tracking, and remote operator commands. These logs capture a diverse set of operational signals, ranging from analog voltage readings and protocol handshakes to access control violations and alarm escalations.

Between September and October 2024, we collected 902,377 raw logs during routine operation and scheduled stress testing. Due to the presence of timestamp drift, inconsistent message granularities, and asynchronous log bursts, we applied time-aligned windowing and event deduplication to extract 15,110 high-fidelity entries for downstream analysis. These logs are notable for their structural heterogeneity and temporal noise, with many sequences combining short diagnostic flags and verbose alarm reports within the same temporal window.

SCADA-X presents a highly safety-critical, real-time control environment where even small anomalies can indicate cascading risks. It is therefore well-suited for evaluating the robustness of anomaly detection models under high-noise, time-sensitive, and regulation-bound industrial contexts.

B. Log Parsing Evaluation on Industrial Datasets

Effective log parsing is a prerequisite for downstream log analysis tasks such as anomaly detection, root cause localization, and automated alert generation. To evaluate the capability of existing parsers to handle structurally diverse and domain-specific log data, we conducted a systematic parsing experiment on the curated subsets of our four datasets introduced in Table II. These subsets serve as a common ground for parser comparison, ensuring consistency across industrial and benchmark data.

Our evaluation covers both traditional rule-based log parsers and recent LLM-based approaches. Rule-based parsers typically rely on regular expression patterns or heuristic clustering strategies, offering fast and lightweight solutions but often struggling with log diversity and format drift. In contrast, LLM-based parsers leverage pretrained language models to identify and extract templates, enabling more adaptive and semantically informed parsing in complex or evolving logging environments.

Rule-based Parsers. We selected four widely used rule-based log parsers: **Drain** [19], **Spell** [20], **AEL** [21], and **IPLoM** [21]. These parsers represent distinct classes of traditional techniques. Drain utilizes a fixed-depth tree to match token sequences; Spell employs Longest Common Subsequence (LCS) for grouping; AEL is frequency-based and tuned for accuracy in structured systems; and IPLoM applies multi-stage partitioning with statistical insights. These tools have been extensively validated on LogHub datasets, making them strong baselines for testing their generalization to industrial logs.

LLM-based Parsers. To examine the capabilities of large language models in log parsing, we evaluated three representative LLM-based approaches: **DivLog** [22], **LILAC** [11], and **LUNAR** [23]. These methods leverage the contextual understanding and few-shot generalization capabilities of pretrained LLMs (e.g., GPT-style models) to extract log templates without handcrafted rules. DivLog introduces a provenance-guided prompt formulation strategy, LILAC combines clustering with in-context learning to reduce reliance on large annotated corpora, and LUNAR proposes a contrastive grouping mechanism to enhance structural pattern abstraction in high-variance log streams. We selected these models for their scalability and promising performance in recent literature.

Each parser was applied to the subset of each dataset, and its output was evaluated using standard parsing metrics. By comparing both rule-based and LLM-based approaches across industrial and benchmark logs, our experiment aims to uncover the strengths and limitations of current parsing strategies when confronted with real-world industrial log complexity.

C. Log Anomaly Detection Evaluation in Industrial Settings

Anomaly detection serves as a cornerstone of intelligent monitoring systems in industrial software, providing early warnings for faults, misconfigurations, and runtime anomalies that may elude traditional rule-based diagnostic methods. To rigorously evaluate the generalization capabilities of existing detection models in the context of real-world industrial applications, we conducted a comprehensive comparison across four structurally diverse datasets introduced in Table II. These datasets span multiple industrial scenarios—including low-resource assembly lines, process control systems, and large-scale SCADA platforms—offering a realistic foundation for testing model robustness under heterogeneous log structures, varying anomaly densities, and domain-specific noise characteristics.

Our evaluation includes five representative detection models: four semi-supervised methods—**PLELog** [24], **LogBERT** [25], **SemiRALD** [12], and **SemiSMAC** [26]—and one supervised method—**LogRobust** [27]. These models were selected to cover a spectrum of design philosophies and data dependencies.

PLELog [24], [28] introduces a semi-supervised log-based anomaly detection technique that employs probabilistic label

estimation to efficiently handle unlabeled log data. This approach integrates semantic information from log events into fixed-length vectors, clusters these vectors using HDBSCAN, and refines the detection process with an attention-based GRU network. Empirical evaluations on datasets such as HDFS and BGL demonstrate PLELog's robust anomaly detection capabilities, reducing the need for extensive manual labeling.

LogBERT [25] is a BERT-based model for anomaly detection, which employs MLM and DeepSVDD losses during training. It processes log sequences refined by a Drain parser to learn normal log patterns through tasks such as masked log key prediction and hypersphere minimization. This strategy enables LogBERT to capture deep contextual relationships within log data, establishing a robust framework for anomaly detection based on deviations from learned log patterns.

SemiRALD [29] is a novel semi-supervised learning-based approach for log anomaly detection that integrates a hybrid language model combining RoBERTa with an attention-based LSTM network. This framework is specifically designed to robustly analyze log data. The system utilizes ChatGPT for efficient log parsing, ensuring the integrity of the parsed log information, and employs a semi-supervised learning paradigm to address the challenge of data scarcity. By learning the typical patterns of system behavior, SemiRALD effectively detects anomalies, making it particularly suitable for environments with limited labeled data. Extensive experiments have demonstrated its promising performance in log anomaly detection.

SemiSMAC [26] is a semi-supervised framework, combining the power of LLMs and Sequential Model-based Algorithm Configuration (SMAC) for enhanced performance. The model utilizes ChatGPT for efficient log parsing and a new log grouping approach, which requires only a small number of labeled samples to generate pseudo-labels for the rest of the data. This technique expands the training set and enables better anomaly detection. SMAC is then used to optimize the hyperparameters of the embedded models, further improving the framework's performance. The SemiSMAC-LSTM variant, using an LSTM backbone, demonstrates superior results in experiments on multiple benchmark datasets, particularly excelling in low-resource scenarios. This makes SemiSMAC a promising solution for scalable and automated anomaly detection, even in challenging real-world applications.

LogRobust [27] is a supervised model built on an attention-based Bi-LSTM architecture. It employs a specialized log parser to preprocess logs, generating TF-IDF and word semantic vectors to extract log features. Both normal and abnormal logs contribute to the training process. The inclusion of attention mechanisms strengthens the model's ability to pinpoint significant anomalies in log data, thereby improving detection accuracy and reliability across various operational contexts.

A key challenge in deploying anomaly detection in industrial environments lies in the scarcity of labeled anomaly data. Many real-world anomalies are rare, subtle, and domain-specific—requiring domain experts to interpret complex temporal patterns across multiple components. As a result, the manual annotation of logs is costly, time-consuming, and often inconsistent across systems. This makes models that require large volumes of labeled training data, such as LogRobust, difficult to apply in practice. Semi-supervised methods offer a more viable path forward, yet their performance and data requirements vary significantly. For example, PLELog assumes access to 50% of normal logs for initialization, while SemiRALD demonstrates promising results using as little as 0.1% of total logs.

To ensure a fair, reproducible, and interpretable evaluation across models and datasets, we constructed dedicated sub-training sets for each of the four datasets. These subsets were carefully curated following a unified sampling protocol, while preserving the statistical properties of the full dataset. Labeling was guided by domain-specific heuristics and expert validation, particularly for industrial logs where anomalies often exhibit subtle, system-dependent behaviors.

All models were trained and evaluated using this standardized setup, which controls for data-related confounding variables. By aligning the training data structure across models, we ensure that observed performance differences are attributable to the models' inherent detection capabilities, rather than artifacts introduced by inconsistent data volume or labeling coverage. This evaluation design allows us to fairly compare supervised and semi-supervised methods under realistic industrial conditions.

1) Sub-Dataset Validation and Processing: To ensure the reliability and experimental validity of the constructed sub-datasets, we conducted a structured multi-stage validation process involving expert review and iterative consensus-building.

Each of the four sub-datasets underwent manual annotation by four domain experts with over five years of experience in software development and system operations. These experts were divided into two independent teams, and a two-round cross-validation protocol was implemented. In the first round, each team independently labeled log entries as either *normal* or *anomalous*, and determined the correct parsing results—i.e., the *log parsing ground-truth templates*. In the second round, each team cross-validated both the anomaly labels and the parsing ground truth produced by the other team, focusing on discrepancies in either classification or structural interpretation. Entries for which the two teams could not reach agreement in either labeling or parsing output were flagged for arbitration. A round-table discussion was subsequently held involving all four experts, during which final decisions were made collaboratively. These sessions allowed for the resolution of ambiguity in both anomaly detection and log parsing, resulting in finalized and consensus-based labels and template structures for each disputed entry.

During the round-table meetings, the experts were also asked to assess the representativeness and balance of each sub-dataset. Representativeness was defined as the extent to which the sub-dataset captured the diversity of the full dataset, including typical variations in system states, workload conditions, and both common and rare anomaly patterns. The goal was to ensure that the selected logs could adequately

reflect realistic operational contexts. Balance, on the other hand, referred to the proportional distribution of normal and anomalous samples, and the reduction of excessive repetitions of near-identical logs (e.g., differing only in dynamic fields). The experts were instructed to confirm that no single log template was overrepresented to the point of inflating evaluation performance. Only when all four experts agreed that both criteria were satisfactorily met was the sub-dataset approved for use in subsequent experiments.

In total, 527 log entries across the four sub-datasets required arbitration and were resolved through expert discussion. The final labels and parsing ground truths were established for these entries through consensus. Moreover, the expert panel unanimously rated the sub-datasets as highly representative and well-balanced. The logs spanned all original datasets and no log format appeared more than six times within any sub-dataset. These validated sub-datasets were thus confirmed to be suitable for use in both log parsing evaluation and as training data for semi-supervised anomaly detection models.

D. Experimental Design and Research Questions

In this section, we present the core research questions and describe the corresponding experimental designs. Each question targets a specific aspect of log analysis in industrial settings, aiming to provide insight into parser robustness, anomaly detection generalization, and the practical implications of industrial log diversity.

RQ1: How well do representative rule-based and LLM-based log parsers generalize across diverse industrial and benchmark datasets?

In this study, we evaluated seven log parsers—Drain, Spell, AEL, IPLoM (rule-based), and DivLog, LILAC, LUNAR (LLM-based)—across the curated subsets of four datasets, including both benchmark (LHSB) and industrial systems (FALL, PIMS, SCADA-X). Each parser was applied independently to extract templates from raw logs, and the results were assessed using standard parsing metrics.

By comparing performance across datasets, we aim to quantify how well each parser adapts to structural complexity, format variation, and semantic diversity—especially in industrial logs where templates are less repetitive and often entangled with hardware-specific information. This evaluation helps identify which parsing strategies are more suitable for industrial applications, and whether recent LLM-based approaches offer significant improvements over traditional rule-based methods.

RQ2: How do semi-supervised and supervised anomaly detection models perform across industrial and benchmark datasets under standardized conditions?

In this study, we investigated the performance and robustness of five state-of-the-art log anomaly detection models: PLELog, LogBERT, SemiRALD, SemiSMAC (semi-supervised), and LogRobust (supervised). Each model was trained and evaluated using the standardized, labeled subsets we created for the four datasets. These subsets ensure fair comparison by aligning data quantity, label coverage, and structural balance.

Through this controlled experiment, we aim to evaluate model generalization to complex industrial log patterns, especially under low-label or sparse-anomaly conditions. We further analyze the trade-offs between annotation cost and detection performance, identifying which models are most practical for deployment in real-world industrial monitoring systems with limited human supervision.

RQ3: What are the key challenges of log analysis in industrial software systems, and how do they differ from those in traditional open-source environments?

To address this question, we conducted a comprehensive analysis of structural and semantic characteristics across the four datasets. Unlike traditional benchmark datasets such as LHSB, which are relatively uniform in structure and anomaly distribution, industrial logs (FALL, PIMS, SCADA-X) exhibit high variability in message format, frequent use of domain-specific error codes, nested parameter structures, and a mixture of control-layer and hardware-level events. We measured key attributes including average log length, template diversity, log entropy, and anomaly sparsity to quantify this heterogeneity. These indicators reflect real-world challenges in industrial contexts, such as noisy sensor streams, interleaved software-hardware events, and log evolution over time due to firmware or configuration changes.

To understand how these characteristics affect model performance, we performed both quantitative and qualitative evaluations. Quantitatively, we compared the performance degradation of log parsers and anomaly detection models from LHSB to industrial datasets, observing significant drops in F1-scores and parsing accuracy. Qualitatively, we conducted a case study analyzing common error types in parsing and detection outputs. We also conducted interviews with industrial maintenance engineers to validate our observations and gain insights into critical but frequently overlooked anomaly patterns—such as subtle numerical deviations, failed overrides, or domain-specific fault signatures. These findings help surface the practical limitations of current log analysis systems and motivate the development of models tailored to industrial software scenarios.

E. Evaluation Metrics

1) Log Parsing: To evaluate the performance of the ten open-source LLM-based parsers in log parsing, we follow recent studies [11], [30] and use Parsing Accuracy (PA) and F1-score of Template Accuracy (FTA) as the evaluation metrics. PA measures the ability of the parser to accurately extract templates from the log entries, which is essential for downstream tasks such as anomaly detection and log analysis. PA is defined as the proportion of correctly parsed log messages to the total number of log messages. A log message is considered correctly parsed if, and only if, all tokens of the templates and variables are correctly identified.

$$PA = \frac{\text{Number of correctly parsed logs}}{\text{Total number of logs}}$$

Where the "correctly parsed log" means that all dynamic variables are accurately identified and replaced.

FTA is a template-level metric that evaluates the correctness of identified templates in parsed logs. It is based on the precision and recall of the templates extracted from log messages. The FTA is calculated as the harmonic mean of Precision and Recall, which are computed specifically for the templates. A template is considered correctly identified if and only if the parsed log shares the same template as the ground-truth template, and all tokens of the parsed template match exactly with those of the ground-truth template.

$$Precision = \frac{\text{Number of correctly identified templates}}{\text{Number of parsed templates}}$$

$$Recall = \frac{\text{Number of correctly identified templates}}{\text{Number of ground-truth templates}}$$

$$FTA = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Where the FTA provides a balanced measure of template identification accuracy, considering both the ability to identify templates correctly (Precision) and the ability to capture all ground-truth templates (Recall).

2) Log Anomaly Detection: Log anomaly detection is treated as a classification problem, and to assess the effectiveness of models in detecting anomalies, we rely on Precision, Recall, and F1-Score metrics to evaluate model performance. The accuracy metric, which calculates the proportion of all predictions correctly made by the anomaly detection model out of the total number of samples, is not used. This is because, in most datasets, normal logs make up the majority, while anomalous logs constitute only a small portion, which is characteristic of well-qualified software. Consequently, a model could attain a high accuracy score even if it predicts all logs as normal. The definitions of each metric are as follows:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Specifically, TP (True Positive) is the count of anomalous log sequences correctly detected by the model. FP (False Positive) is the count of normal log sequences incorrectly identified as anomalies. TN (True Negative) is the count of normal logs that are correctly classified. FN (False Negative) is the count of anomalous log sequences that the model failed to detect.

III. Study Results

In this section, we present the experimental results and provide an analysis to address each research question.

In this study, we evaluated seven log parsers—Drain, Spell, AEL, IPLoM (rule-based), and DivLog, LILAC, LUNAR (LLM-based)—across the curated subsets of four datasets, including both benchmark (LHSB) and industrial systems (FALL, PIMS, SCADA-X). Each parser was applied independently to extract templates from raw logs, and the results were assessed using standard parsing metrics.

A. Effectiveness of Log Parsers (RQ1)

Table III presents the PA and FTA of seven representative log parsers evaluated across four datasets. A consistent trend is observed: while most parsers perform reasonably well on the benchmark dataset (LHSB), their effectiveness significantly degrades on the industrial datasets, especially on SCADA-X and PIMS. For instance, Drain achieves a PA of 0.815 and FTA of 0.571 on LHSB, but its FTA drops to 0.474 on SCADA-X and 0.503 on PIMS, reflecting the difficulty of generalizing rule-based strategies to structurally complex logs.

LLM-based parsers generally outperform traditional methods across all datasets. LUNAR achieves the highest parsing accuracy overall, with FTA values of 0.828 on LHSB and 0.797 on FALL. Even on SCADA-X—arguably the most challenging dataset—LUNAR maintains a strong FTA of 0.719, surpassing all other models. In contrast, rule-based methods like IPLoM perform poorly on industrial datasets, with FTA dropping as low as 0.329 on SCADA-X and 0.362 on PIMS. These results suggest that conventional parsers are sensitive to format variability and fail to adapt to evolving logging schemes.

From a vertical perspective, we observe that every model's performance consistently declines when moving from LHSB to the industrial datasets. For example, DivLog achieves a FTA of 0.604 on LHSB but only 0.571 on PIMS and 0.588 on SCADA-X. Similarly, LILAC drops from 0.682 on LHSB to 0.566 on SCADA-X. This pattern highlights the limitations of training or evaluating log parsers solely on benchmark datasets, which may underestimate the challenges posed by real-world industrial systems with heterogeneous log formats, embedded noise, and domain-specific semantics.

Overall, the evaluation indicates that while LLM-based parsers provide improved robustness in industrial settings, there remains a notable performance gap compared to their performance on benchmark logs. This underscores the need for adaptive and industrial-aware parsing strategies, particularly for applications in safety-critical and high-variance environments.

Answer to RQ1: The results reveal a clear performance gap between benchmark and industrial datasets, with all parsers—especially rule-based methods—struggling to maintain accuracy in structurally diverse industrial logs. LLM-based approaches demonstrate superior robustness, though they too exhibit diminished performance under real-world conditions.

TABLE III: The PA and FTA of seven representative log parsers evaluated across four datasets.

	Drain		Spell		AEL		IPLoM		DivLog		LILAC		LUNAR	
	PA	FTA	PA	FTA	PA	FTA	PA	FTA	PA	FTA	PA	FTA	PA	FTA
LHSB	0.815	0.571	0.769	0.492	0.794	0.502	0.628	0.401	0.892	0.604	0.871	0.682	0.948	0.828
FALL	0.754	0.548	0.702	0.473	0.706	0.468	0.603	0.387	0.858	0.583	0.802	0.641	0.886	0.797
PIMS	0.737	0.503	0.674	0.468	0.682	0.481	0.573	0.362	0.863	0.571	0.785	0.618	0.852	0.774
SCADA-X	0.692	0.474	0.642	0.434	0.649	0.436	0.557	0.329	0.805	0.588	0.849	0.566	0.808	0.719

B. Anomaly Detection Performance (RQ2)

As shown in Table IV, detection models achieve significantly higher performance on the benchmark dataset (LHSB) compared to industrial datasets, confirming that traditional evaluation settings may overestimate model generalizability. For example, PLELog achieves an F1-score of 0.796 on LHSB but drops to 0.615 on SCADA-X and 0.634 on PIMS. Similar patterns are observed for LogBERT (F1: 0.980 on LHSB vs. 0.755 on SCADA-X) and LogRobust (F1: 0.956 on LHSB vs. 0.775 on PIMS). These results suggest that anomaly detection in industrial environments is substantially more challenging due to noisy logs, diverse templates, and subtle anomaly characteristics.

From a horizontal comparison across models, **SemiRALD consistently outperforms all baselines across all datasets**, achieving F1-scores of 0.985 (LHSB), 0.920 (FALL), 0.921 (PIMS), and 0.912 (SCADA-X). These results indicate that SemiRALD is highly robust to industrial log variability and can generalize well even in low-resource or heterogeneous settings. Notably, even in SCADA-X—arguably the most complex dataset—SemiRALD maintains its leading performance, whereas other models such as PLELog and LogBERT experience a sharp decline.

Vertically, we observe that all models exhibit performance degradation when transitioning from LHSB to industrial datasets. The average F1-score drop from LHSB to SCADA-X is approximately 20–25% for models like LogBERT and LogRobust, demonstrating that benchmark-trained models may not be reliable under industrial deployment conditions. This reinforces the necessity of evaluating detection models under realistic, diverse, and domain-specific log environments.

Overall, these results highlight two key insights: (1) industrial anomaly detection poses unique structural and semantic challenges that compromise the effectiveness of traditional models, and (2) SemiRALD demonstrates the best balance between detection accuracy and generalization, making it a promising candidate for real-world industrial log analysis systems.

Answer to RQ2: The results demonstrate that anomaly detection models perform significantly better on benchmark datasets than on real-world industrial logs, highlighting the limitations of conventional evaluation practices. Among all methods, SemiRALD exhibits the most consistent and robust performance across diverse industrial scenarios.

C. RQ3: Challenges in Industrial Log Analysis

To better understand the limitations of current models, we conducted a case study by analyzing misclassified log entries from both the log parsing and anomaly detection tasks. Table V summarizes the most frequent error types and provides representative examples. These errors were primarily identified from the three industrial datasets (FALL, PIMS, SCADA-X), which exhibited greater diversity and irregularity compared to the benchmark dataset (LHSB). In addition, we consulted with five experienced maintenance engineers who reviewed a subset of the parsed and detected results and provided feedback on practical concerns often overlooked by automated systems.

One of the most prevalent parsing issues in industrial logs was the removal of critical semantic content, such as status keywords (e.g., *fail*, *unauthorized*) and system-specific error codes. Engineers emphasized that in industrial environments, such keywords carry diagnostic meaning and are often used in standard operating procedures or linked to incident reports. For instance, in PIMS, the absence of an "unauthorized" label in the parsed template led to false negatives in access control monitoring. Another common error involved the failure to isolate meaningful variables such as usernames, IP addresses, and sensor values—especially when they appeared in compound expressions or nested structures. These omissions can significantly hinder root cause analysis and automated alerting.

In the anomaly detection task, we observed that models frequently misclassified benign logs due to superficial keyword matching. For example, control logs that contain terms like "error" or "alert" for traceability purposes were often labeled as anomalous, even though they represent expected system responses. Conversely, subtle yet critical faults—such as numerical drifts in torque, temperature, or position—were routinely ignored. Engineers pointed out that many industrial anomalies manifest through small parameter shifts rather than abrupt semantic deviations, which current models are not well-equipped to detect without context-aware mechanisms.

These findings underscore several key differences between traditional and industrial log environments. First, industrial logs are often multimodal and domain-specific, embedding physical state changes alongside software events. Second, formatting and template structures vary not only across systems but also over time within the same system, particularly in scenarios involving firmware updates or vendor component changes. Finally, industrial logs often exhibit low anomaly frequency and high class imbalance, which makes them ill-suited

TABLE IV: The performance of different models on four datasets.

Model	LHSB			FALL			PIMS			SCADA-X		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
PLELog	0.905	0.710	0.796	0.755	0.688	0.720	0.545	0.762	0.634	0.470	0.890	0.615
LogBERT	**0.990**	0.970	0.980	0.810	0.760	0.784	0.790	0.695	0.739	0.805	0.710	0.755
SemiRALD	0.975	**0.995**	**0.985**	**0.910**	**0.930**	**0.920**	**0.905**	**0.938**	**0.921**	**0.900**	**0.925**	**0.912**
SemiSMAC	0.935	0.940	0.937	0.720	0.795	0.755	0.675	0.780	0.724	0.885	0.818	0.850
LogRobust	0.950	0.963	0.956	0.765	0.890	0.823	0.750	0.802	0.775	0.795	0.780	0.787

TABLE V: Common errors in log parsing and anomaly detection observed in industrial software logs.

(a) Log Parsing (Industrial Logs)

Error Type	**Freq.**	**Example (Raw Log → Parsed Template)**
Status Keyword Dropped	14.52%	*Reason=false, Service status: enabled → Reason=<*>, Service status: <*>*
User Identity Obscured	14.08%	*Login failed for user alice on host backend-prod-01 → Login failed for user <*> on host <*>*
Error Code Removed	13.36%	*ERROR CODE 502: Gateway Timeout at /api/process → ERROR CODE <*>: Gateway Timeout at <*>*
Incomplete Parameter Parsing	12.95%	*Auth: user=alice, src=192.168.0.15, method=SSH, result=failed → Auth: user=alice, src=<*>, method=SSH, result=<*>*
Truncated Request Path	12.34%	*GET /machine_status?line=3&batch=842 → GET /<*>*
Sensor Value Lost	10.87%	*Sensor reading: arm_pos=(0.22, 0.98) vs expected=(0.20, 1.00) → Sensor reading: arm_pos=<*>, expected=<*>*
Network Field Merged	7.86%	*Connect from 10.0.0.5:8080 to 172.16.4.10:443 → Connect from <*> to <*>*
Sentence Boundary Lost	7.01%	*Check at 12:00. Last OK: 11:30. Errors: 0 → Check at 12:00. Last OK: <*>*

(b) Log Anomaly Detection (Industrial Logs)

Error Type	**Freq.**	**Description and Example**
Handled Fault Misclassified (FP)	16.38%	Control-layer errors caught by internal recovery mechanisms were flagged as anomalies. (*Trigger 'error' caught and handled internally*)
Alert Verbality Misinterpreted (FP)	15.72%	Benign logs with "alert" or "warn" were falsely flagged. (*Alert: pressure cycle initialized*)
Missing Failure Indicator (FN)	14.65%	Critical keywords dropped in templates. (*Access unauthorized → Access <*>*)
Numerical Deviation Ignored (FN)	14.18%	Slight numerical changes overlooked. (*torque deviation Δ=0.012 exceeds limit*)
Integrity Check Missed (FN)	11.53%	Output mismatch due to checksum errors not flagged. (*output mismatch: checksum failure*)
Template Ambiguity (FN)	10.04%	Faulty logs matched to success templates. (*Job /welding returned status 503*)
Retry Overlooked (FN)	9.20%	Repeated retries not treated as degradation. (*retries=5, status=true*)
Scheduling Conflict Missed (FN)	8.37%	Overridden or preempted tasks not flagged. (*Task 301 preempted by Task 299*)

for models trained on homogeneous benchmark datasets. Future log analysis systems must incorporate richer semantic understanding, adaptive parsing techniques, and domain-specific heuristics to address these challenges effectively.

Answer to RQ3: Our analysis reveals that industrial log analysis presents unique challenges—such as semantic variability, subtle numerical anomalies, and inconsistent formatting—that are not adequately addressed by existing parsing or detection models. These findings highlight the need for domain-adaptive, context-aware approaches to achieve reliable performance in real-world industrial environments.

IV. Implications

Our findings have several important implications for the development and deployment of log analysis systems in industrial software environments. Unlike traditional open-source systems, industrial software generates logs that are structurally diverse, semantically dense, and tightly coupled with physical operations. This complexity demands more robust and context-aware log parsing and anomaly detection techniques, particularly in safety-critical or real-time applications. Models trained solely on benchmark datasets are insufficient for these environments and risk underperforming when deployed in the field.

Firstly, the poor generalization of rule-based parsers and some neural models on industrial datasets highlights the necessity for adaptive parsing frameworks that can handle evolving log formats and heterogeneous structures. Industrial systems often undergo frequent updates to firmware, control logic, or hardware configurations, which in turn affect log schema. As such, static or rigid template-based parsers quickly become obsolete. LLM-based methods show promise due to their contextual reasoning abilities, but further adaptation—such as domain-tuned prompts and continual learning—is needed to ensure reliability in production settings.

Secondly, our anomaly detection results emphasize the need for detection models that can operate under limited supervision and detect subtle deviations beyond keyword-based heuristics. Industrial anomalies frequently manifest as gradual shifts in numerical signals, silent misconfigurations, or protocol-level irregularities, which are easily missed by models optimized for overt textual anomalies. Semi-supervised approaches like SemiRALD performed best in these scenarios, suggesting that hybrid learning paradigms may offer a practical trade-off

between performance and labeling cost.

Finally, our case study with engineers reveals a gap between model outputs and operational interpretability. Logs in industrial systems are not only diagnostic artifacts but also serve as compliance records, audit trails, and triggers for automated maintenance. Therefore, future log analysis systems must prioritize explainability, traceability, and human-in-the-loop integration to support real-world decision-making. This entails designing output formats and interfaces that can be easily consumed by engineers, operators, and safety auditors in high-stakes environments.

V. Threats to Validity

One potential threat to validity lies in the representativeness of the industrial datasets used in this study. While we collaborated with multiple industrial partners and selected systems from distinct domains (e.g., assembly lines, process control, energy monitoring), the datasets still reflect a limited subset of industrial logging environments. As such, our findings may not generalize to all types of industrial software, particularly systems with proprietary logging formats or embedded constraints not captured in our datasets. Moreover, due to confidentiality agreements, we anonymized certain log fields, which may have impacted the semantic richness available for parsing and detection.

Another threat concerns the fairness of model comparison across diverse log datasets. Despite our efforts to construct standardized training subsets and use consistent evaluation metrics, some models may be inherently more sensitive to data volume, label sparsity, or log structure. For instance, supervised models like LogRobust benefit from dense labels, while semi-supervised models vary in their reliance on normal versus anomalous samples. Although we minimized these discrepancies through a unified evaluation protocol, subtle variations in data-label dynamics or hyperparameter tuning may still influence the observed performance rankings.

VI. Related Work

A. Log Collection

Logs in industrial software systems exhibit a high degree of heterogeneity due to varying logging standards, proprietary software architectures, and domain-specific operational contexts [30], [31]. Unlike the relatively uniform logs found in academic or open-source datasets, industrial logs are often generated by tightly integrated systems involving hardware-software co-design, real-time constraints, and component-specific logging mechanisms. This heterogeneity introduces substantial challenges for log-based anomaly detection, particularly in terms of generalizing across different industrial settings.

Although platforms such as LogHub [10] have facilitated research by providing a range of open-source log datasets, these datasets are primarily collected from legacy server-based applications and distributed computing systems. Their structural consistency and simplified formats do not adequately capture the intricacies of logs produced by contemporary industrial software. In industrial environments, logs tend to be longer, more verbose, and semantically richer, often containing embedded sensor readings, proprietary error codes, and hardware interaction traces. These characteristics make traditional log parsing and anomaly detection models less effective when directly applied to industrial data.

Moreover, industrial logs are influenced by diverse operational factors such as production line variations, maintenance procedures, and software versioning, resulting in frequent shifts in log distribution and format. These dynamic and context-sensitive characteristics necessitate more robust, adaptive log collection mechanisms. In our study, we worked closely with domain engineers to obtain real-world logs from three large-scale industrial software systems across different sectors. The collected datasets exhibit significant variability not only in log format but also in volume and frequency patterns, providing a realistic foundation for evaluating the generalizability and resilience of log analysis models in industrial settings.

B. Log Parsing

Log parsing refers to the automatic transformation of raw log messages into structured representations by identifying constant components (event templates) and variable components (parameters) [24], [32]. Research has indicated that over 50% of individual log messages contain more than three variable parameters, with approximately 20% even exceeding five variables [9]. This variability complicates precise parsing, as accurately distinguishing each parameter within these dynamically structured log entries is far from trivial [33]. Prior studies [21], [34]–[37] have demonstrated that the quality of log parsing plays a crucial role in the effectiveness of downstream tasks such as anomaly detection and root cause analysis [30], [31]. As such, log parsing is widely regarded as a foundational step in the log analysis pipeline.

Current mainstream log parsing approaches can be broadly categorized into rule-based methods (e.g., Drain [19], [20]) and LLM-based methods [9], [11], [36]. While rule-based parsers rely on manually defined patterns and heuristic rules, recent studies have shown that LLM-based parsers excel at recognizing complex and diverse textual structures, offering substantial advantages—particularly in scenarios involving highly variable or unstructured log formats. LogGPT [38] introduces a novel approach to log-based anomaly detection, leveraging ChatGPT-3.5's language interpretation capabilities. The results show promising performance and strong interpretability, providing valuable insights into the potential of prompt-based models for log data analysis. However, LogGPT's performance on the BGL dataset is suboptimal, underscoring the need for further improvements to this approach. LogPPT [36] introduces a prompt-based few-shot learning approach for log parsing. This method captures log templates and parameters using minimally labeled data, outperforming traditional techniques in both effectiveness and efficiency across several public log datasets. DivLog [9] utilizes in-context learning with LLMs to enhance log parsing by leveraging diverse, small

samples and training-free prompt construction, achieving state-of-the-art performance in accuracy and template precision across multiple large-scale datasets. Additionally, LILAC [11] integrates LLMs with an adaptive parsing cache and ICL, significantly improving template accuracy, parsing efficiency, and reducing reliance on LLM queries. Nevertheless, the practical deployment of such large models is constrained by substantial computational costs and resource requirements, limiting their applicability in real-time or low-resource scenarios.

C. Log Anomaly Detection

Currently, researchers have proposed various methods for detecting abnormal logs from large-scale log datasets, including rule-based filtering, statistical modeling, and a growing body of machine learning and deep learning approaches [39], [40]. These techniques aim to identify deviations from expected system behavior by analyzing log sequences, patterns, or semantics—making them essential tools for fault diagnosis, performance monitoring, and early risk warning [41]. Du et al. [17] introduced the DeepLog framework, which treats log information as natural language sequences, autonomously learns log patterns from normal execution sequences, and identifies anomalies by assessing whether incoming log events deviate from the predictions of a stacked LSTM model. Zhang et al. [27] introduced LogRobust, which represents log events as semantic vectors and uses an attention-based Bi-LSTM model to detect anomalies. While supervised methods are capable of accurately diagnosing system conditions, their primary limitation lies in the requirement for a substantial volume of normal and abnormal data during the training phase. In contrast, semi-supervised methods require only a small amount of labeled data for training, substantially reducing the cost of manual annotation and dependence on labeled datasets compared to fully supervised approaches. Meanwhile, they often outperform unsupervised methods in predictive performance, making them an increasingly popular focus in recent research. Yang et al. [24], [28] introduced a semi-supervised learning framework known as PLELog, which relies on probabilistic label estimation. PLELog utilizes the HDBSCAN [42] clustering method for estimating labels probabilistically on unlabeled log sequences, addressing the challenge of limited labeling. LogBERT [25] learns patterns within normal log sequences through masked log message prediction and hypersphere volume minimization, yet it cannot specifically identify semantic information in abnormal logs. SemiRALD integrates the strengths of RoBERTa and Bi-LSTM, achieving robust performance across multiple datasets and positioning itself as a more representative model.

In industrial software systems, the importance of log anomaly detection is further amplified due to the high stakes involved. Failures in such systems can lead to significant financial losses, production downtime, or even threats to safety in critical domains such as manufacturing, energy, and transportation. Logs often serve as the only non-intrusive channel to monitor internal operations in real time, especially when access to system internals is limited due to proprietary hardware or regulatory constraints. Therefore, timely and accurate detection of anomalies through log analysis is not only desirable but crucial for maintaining operational resilience in industrial contexts.

Despite the growing sophistication of anomaly detection algorithms, most existing work has focused on evaluating these methods using publicly available datasets—such as those from LogHub—which are predominantly drawn from open-source software or traditional distributed systems. These datasets do not fully reflect the complexity, scale, or dynamic behavior of modern industrial software. To date, research on log anomaly detection in industrial software remains limited. Few studies have systematically investigated the challenges unique to industrial settings, such as inconsistent log formats, limited labeled data, and rapidly changing operational conditions. This gap underscores the need for domain-adapted detection techniques and industrial-grade evaluation benchmarks—motivating the contributions of our study.

VII. Conclusion

This study presents a comprehensive empirical evaluation of log parsing and anomaly detection methods in the context of industrial software systems. By constructing and analyzing four diverse datasets—spanning benchmark and real-world industrial environments—we reveal that existing models, particularly those trained on traditional open-source logs, face significant performance degradation when applied to industrial settings. Our findings underscore the limitations of current log analysis techniques in handling the structural variability, semantic richness, and operational complexity inherent in industrial logs. Moreover, we demonstrate the value of semi-supervised and LLM-based approaches in improving robustness and generalizability. These insights motivate the development of domain-adaptive, interpretable, and low-supervision solutions tailored for industrial software monitoring and risk mitigation.

Acknowledgment

This work is partially supported by the General Research Fund of the Research Grants Council of Hong Kong and the Research Funds from the City University of Hong Kong (6000796, 9229109, 9229098, 9220103, 9229029).

References

[1] S. Pirelli, "Scalable teaching of software engineering theory and practice: An experience report," in *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training*, 2024, pp. 286–296.

[2] S. He, J. Zhu, P. He, and M. R. Lyu, "Experience report: System log analysis for anomaly detection," in *2016 IEEE 27th international symposium on software reliability engineering (ISSRE)*. IEEE, 2016, pp. 207–218.

[3] J.-G. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, "Mining invariants from console logs for system problem detection," in *2010 USENIX Annual Technical Conference (USENIX ATC 10)*, 2010.

[4] V. Garousi, G. Giray, E. Tüzün, C. Catal, and M. Felderer, "Aligning software engineering education with industrial needs: A meta-analysis," *Journal of Systems and Software*, vol. 156, pp. 65–83, 2019.

[5] X. Wu, H. Li, and F. Khomh, "On the effectiveness of log representation for log-based anomaly detection," *Empirical Software Engineering*, vol. 28, no. 6, p. 137, 2023.

[6] X. Ma, J. Keung, P. He, Y. Xiao, X. Yu, and Y. Li, "A semisupervised approach for industrial anomaly detection via self-adaptive clustering," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 2, pp. 1687–1697, 2023.

[7] H. Dai, H. Li, C.-S. Chen, W. Shang, and T.-H. Chen, "Logram: Efficient log parsing using n n-gram dictionaries," *IEEE Transactions on Software Engineering*, vol. 48, no. 3, pp. 879–892, 2020.

[8] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *2009 ninth IEEE international conference on data mining*. IEEE, 2009, pp. 149–158.

[9] J. Xu, R. Yang, Y. Huo, C. Zhang, and P. He, "Divlog: Log parsing with prompt enhanced in-context learning," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, 2024, pp. 1–12.

[10] J. Zhu, S. He, P. He, J. Liu, and M. R. Lyu, "Loghub: A large collection of system log datasets for ai-driven log analytics," in *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2023, pp. 355–366.

[11] Z. Jiang, J. Liu, Z. Chen, Y. Li, J. Huang, Y. Huo, P. He, J. Gu, and M. R. Lyu, "Lilac: Log parsing using llms with adaptive parsing cache," *Proceedings of the ACM on Software Engineering*, vol. 1, no. FSE, pp. 137–160, 2024.

[12] Y. Sun, J. W. Keung, Z. Yang, S. Liu, and H. K. Yu, "Semirald: A semi-supervised hybrid language model for robust anomalous log detection," *Information and Software Technology*, p. 107743, 2025.

[13] B. Yu, J. Yao, Q. Fu, Z. Zhong, H. Xie, Y. Wu, Y. Ma, and P. He, "Deep learning or classical machine learning? an empirical study on log-based anomaly detection," in *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, 2024, pp. 1–13.

[14] Y. Sun, J. Keung, Z. Yang, S. Liu, and H. K. Yu, "Improving anomaly detection in software logs through hybrid language modeling and reduced reliance on parser," *Automated Software Engineering*, vol. 33, no. 1, p. 12, 2026.

[15] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, 2009, pp. 117–132.

[16] Q. Lin, H. Zhang, J.-G. Lou, Y. Zhang, and X. Chen, "Log clustering based problem identification for online service systems," in *Proceedings of the 38th International Conference on Software Engineering Companion*, 2016, pp. 102–111.

[17] M. Du, F. Li, G. Zheng, and V. Srikumar, "Deeplog: Anomaly detection and diagnosis from system logs through deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1285–1298.

[18] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," in *37th annual IEEE/IFIP international conference on dependable systems and networks (DSN'07)*. IEEE, 2007, pp. 575–584.

[19] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *2017 IEEE international conference on web services (ICWS)*. IEEE, 2017, pp. 33–40.

[20] M. Du and F. Li, "Spell: Streaming parsing of system event logs," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 859–864.

[21] J. Zhu, S. He, J. Liu, P. He, Q. Xie, Z. Zheng, and M. R. Lyu, "Tools and benchmarks for automated log parsing," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. IEEE, 2019, pp. 121–130.

[22] A. Tufek and M. S. Aktas, "On the provenance extraction techniques from large scale log files," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 15, p. e6559, 2023.

[23] J. Huang, Z. Jiang, Z. Chen, and M. Lyu, "No more labelled examples? an unsupervised log parser with llms," *Proceedings of the ACM on Software Engineering*, vol. 2, no. FSE, pp. 2406–2429, 2025.

[24] Y. Lin and Y.-Y. Chiang, "A semi-supervised learning approach for abnormal event prediction on large network operation time-series data," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 1024–1033.

[25] H. Guo, S. Yuan, and X. Wu, "Logbert: Log anomaly detection via bert," in *2021 international joint conference on neural networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[26] Y. Sun, J. W. Keung, Z. Yang, S. Liu, and Y. Liao, "Semismac: A semi-supervised framework for log anomaly detection with automated hyperparameter tuning," *Information and Software Technology*, p. 107869, 2025.

[27] X. Zhang, Y. Xu, Q. Lin, B. Qiao, H. Zhang, Y. Dang, C. Xie, X. Yang, Q. Cheng, Z. Li *et al.*, "Robust log-based anomaly detection on unstable log data," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 807–817.

[28] L. Yang, J. Chen, Z. Wang, W. Wang, J. Jiang, X. Dong, and W. Zhang, "Semi-supervised log-based anomaly detection via probabilistic label estimation," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 2021, pp. 1448–1460.

[29] Y. SUN, J. Keung, Z. Yang, S. Liu, and H. K. Yu, "Semirald: A semi-supervised hybrid language model for robust anomalous log detection," *Available at SSRN 4927951*, 2024.

[30] Z. Ma, A. R. Chen, D. J. Kim, T.-H. P. Chen, and S. Wang, "Llmparser: An exploratory study on using large language models for log parsing," 2024.

[31] Z. Li, C. Luo, T.-H. Chen, W. Shang, S. He, Q. Lin, and D. Zhang, "Did we miss something important? studying and exploring variable-aware log abstraction," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 830–842.

[32] S. He, P. He, Z. Chen, T. Yang, Y. Su, and M. R. Lyu, "A survey on automated log analysis for reliability engineering," *ACM computing surveys (CSUR)*, vol. 54, no. 6, pp. 1–37, 2021.

[33] X. Xie, S. Jian, C. Huang, F. Yu, and Y. Deng, "Logrep: Log-based anomaly detection by representing both semantic and numeric information in raw messages," in *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2023, pp. 194–206.

[34] Z. A. Khan, D. Shin, D. Bianculli, and L. Briand, "Guidelines for assessing the accuracy of log message template identification techniques," in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 1095–1106.

[35] Y. Huo, Y. Su, C. Lee, and M. R. Lyu, "Semparser: A semantic parser for log analytics," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 881–893.

[36] V.-H. Le and H. Zhang, "Log parsing with prompt-based few-shot learning," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*. IEEE, 2023, pp. 2438–2449.

[37] Y. Liu, X. Zhang, S. He, H. Zhang, L. Li, Y. Kang, Y. Xu, M. Ma, Q. Lin, Y. Dang *et al.*, "Uniparser: A unified log parser for heterogeneous log data," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1893–1901.

[38] J. Qi, S. Huang, Z. Luan, S. Yang, C. Fung, H. Yang, D. Qian, J. Shang, Z. Xiao, and Z. Wu, "Loggpt: Exploring chatgpt for log-based anomaly detection," in *2023 IEEE International Conference on High Performance Computing & Communications, Data Science & Systems, Smart City & Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*. IEEE, 2023, pp. 273–280.

[39] X. Ma, H. Zou, P. He, J. Keung, Y. Li, X. Yu, and F. Sarro, "On the influence of data resampling for deep learning-based log anomaly detection: Insights and recommendations," *IEEE Transactions on Software Engineering*, 2024.

[40] 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 Transactions on Software Engineering*, 2025.

[41] Y. Sun, J. Keung, H. K. Yu, S. Liu, Y. Liao, and J. Zhang, "Beyond log parsers: A scalable ai-driven framework for efficient log anomaly detection in software engineering," in *2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2025, pp. 1382–1387.

[42] L. McInnes and J. Healy, "Accelerated hierarchical density based clustering," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2017, pp. 33–42.