Journal of Computational Systems and Applications

https://jcsa.cultechpub.com/jcsa

Cultech Publishing

Article

Retrieval-Grounded HDFS Log Anomaly Detection and Deterministic Failure Narrative Generation

Xinzhuo Sun[1*], Ziliang Samuel Zhong[2], Qiyou Wu[3]

[1]Computer Engineering, Cornell Tech, Cornell University, 2 West Loop Road, New York, NY 10044, USA

[2]New York University, 70 Washington Square South, New York, NY 10012, USA

[3]Artificial Intelligence, Northeastern University, 440 Huntington Avenue, Boston, MA 02115, USA

[*]Corresponding author: Xinzhuo Sun, xinzhuo.sun0808@gmail.com

Abstract

Objectives/Scope: This paper evaluates Hadoop Distributed File System (HDFS) log anomaly detection on the public HDFS_100k structured-log subset derived from LogHub, with emphasis on detection, evidence-grounded explanation, and selective refusal under label ambiguity. Methods, Procedures, and Process: After block-level sessionization, the benchmark contains 7,940 traces and 313 anomalous sessions (3.94%). A reproducible hybrid detector that combines linear discriminative scoring, pattern-memory posteriors, trace statistics, and a calibrated stacking stage was implemented. A retrieval-augmented generation-inspired layer then assembles evidence bundles and renders deterministic failure narratives; no external large language model is used. Results, Observations, and Conclusions: On a fixed 60/20/20 split, the proposed model obtains F1-score (the harmonic mean of precision and recall) = 0.6452, precision-recall area under the curve (PR-AUC) = 0.5440, and receiver operating characteristic area under the curve (ROC-AUC) = 0.7562. Although logistic regression and linear support vector machine baselines reach slightly higher fixed-threshold F1-score values of 0.6596, the proposed model provides the best cross-validation ranking quality, with mean PR-AUC = 0.5318 and mean ROC-AUC = 0.7558. Exact-pattern ambiguity explains 32 of the 33 fixed-test errors. Selective refusal improves F1-score to 0.6742 at 89.8% coverage and to 0.9375 at 23.9% coverage. Novel/Additive Information: The study contributes an ambiguity-aware operational pipeline that integrates scoring, explanation, and abstention, providing a transferable evaluation pattern for data-intensive infrastructure, including storage and monitoring systems used in petroleum-industry digital operations.

Keywords

HDFS and system logs, Anomaly detection, Retrieval-augmented generation, Selective refusal; Failure narratives, Reproducible evaluation

Article History

Received: 10 April 2026 Revised: 27 April 2026

Accepted: 30 April 2026 Available Online: 04 June 2026

1. Introduction

Large distributed storage systems produce voluminous execution logs, and those logs remain one of the most accessible sources of operational evidence for anomaly detection and diagnosis. The Hadoop Distributed File System (HDFS) has therefore become one of the most studied system-log benchmarks in the literature, beginning with early console-log mining work by Xu et al. [1] and continuing through contemporary public benchmarks such as LogHub [2]. The value of the HDFS setting is not only that it is widely used, but also that it is structured around block traces. When logs are grouped by block identifier, the resulting trace becomes a compact behavioral unit that can be classified, compared, and summarized. That property has made HDFS a natural target for sequence-based and representation-based detectors, including Drain-supported pipelines [3,4], DeepLog [5], LogAnomaly [6], PLELog [7], LogBERT [8], and HitAnomaly [9].

Despite these advances, two gaps remain important for practical operations. First, many studies optimize only the final anomaly label and report little about the operator-facing explanation that should accompany the alarm. In production settings, a label without supporting evidence is often not actionable. Second, most benchmark papers assume that every incoming trace should receive a forced decision, even when the trace closely resembles both normal and anomalous historical executions. That assumption is convenient for leaderboard reporting, but it is poorly aligned with human operations. In real incident response, a well-grounded abstention can be safer than a confident but poorly supported label. Selective classification [10], calibration [11], and interpretable-modeling work [12-16] have shown that refusal and explanation must be studied together, yet that perspective has not been fully integrated into HDFS log anomaly detection.

This paper addresses those gaps with a unified empirical study of anomaly detection, retrieval-grounded explanation, and refusal control. The proposed pipeline combines three ingredients. The first ingredient is a calibrated hybrid detector that merges discriminative sequence scoring, exact-pattern memory, and coarse trace statistics. The second ingredient is a retrieval-grounded narrative layer inspired by retrieval-augmented generation (RAG)-style evidence conditioning [17], but in this paper the term “generation” refers to deterministic evidence-to-text rendering rather than output from an external large language model (LLM). This design keeps the narrative layer reproducible and auditable. The third ingredient is a selective refusal policy that can abstain on highly ambiguous HDFS traces. The complete pipeline is evaluated on the public HDFS_100k structured subset distributed with loglizer and the official HDFS anomaly label file, both derived from the same LogHub HDFS benchmark family [2,18,19]. Because the reported experiments are restricted to this public derivative subset, the results should be interpreted as subset-level evidence rather than as full-HDFS leaderboard performance, while still preserving the block-trace labeling protocol and benchmark semantics.

The main contribution of the paper is not the invention of a new standalone sequence model. Instead, the novelty lies in integrating detection, evidence-grounded explanation, and ambiguity-aware abstention into a unified operational pipeline for HDFS traces. The study provides a fully reproducible empirical evaluation on 7,940 block traces and 104,815 log lines; quantifies label ambiguity at the exact-pattern level; shows that the proposed hybrid model leads the compared methods in ranking quality under three-fold cross-validation; demonstrates that refusal materially improves accepted-case quality; and packages the detector together with deterministic failure narratives that summarize trace length, dominant templates, score-band status, and pattern-memory evidence. The paper therefore treats anomaly detection as an operational workflow rather than as a single forced classification number.

From an operations perspective, this distinction between detection and actionability is critical. Storage services rarely fail in a way that is visible from one line of output; instead, they produce short trace fragments that must be interpreted within a familiar but noisy event grammar. A useful anomaly-detection study must therefore explain what kind of operational object it is predicting. In HDFS, the natural object is the block trace. Once that decision is made, the evaluation should preserve the trace context all the way through the final alarm, which is why the current paper keeps sessionization, retrieval, and narrative generation inside the same pipeline rather than treating them as separate post-processing scripts. The overall workflow is summarized in Figure 1.

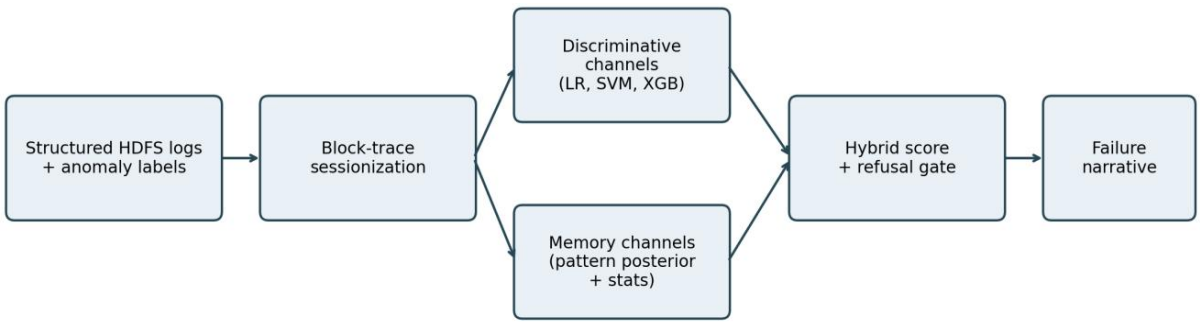

Figure 1. End-to-end workflow of the proposed HDFS anomaly detection, retrieval, refusal, and narrative pipeline.

2. Literature Review

The first branch of related work focuses on converting raw log messages into stable event templates. He et al. evaluated the dependence of log mining on parsing quality [3], and Drain later became a standard parsing baseline because of its online fixed-depth tree design [4]. These studies matter because anomaly detectors are only as stable as the templates they ingest. The HDFS benchmark is particularly suitable for this style of analysis because block identifiers make session construction relatively explicit, which reduces one source of ambiguity while preserving another: the same block-level event pattern can still correspond to both normal and anomalous executions [2,19].

The second branch centers on log anomaly detection itself. Early feature-engineering methods mined invariants or classification boundaries from event counts and event-order statistics [1,20-23]. DeepLog introduced sequence modeling with recurrent neural networks and demonstrated that next-event prediction could work well on HDFS [5]. LogAnomaly extended the idea by combining sequential and quantitative deviations [6]. Later studies emphasized practical robustness: PLELog used probabilistic label estimation and semi-supervised training to reduce manual labeling requirements [7]; LogBERT replaced recurrent encoders with bidirectional transformer pretraining [8]; and HitAnomaly modeled template sequences and parameters in a hierarchical transformer architecture [9]. These methods have advanced accuracy, but the dominant emphasis remains end-task detection rather than alarm presentation or abstention behavior.

The third branch concerns retrieval, language modeling, and explanation. Transformer models [24], BERT [25], and large autoregressive language models [26] made it natural to formulate diagnostic reporting as conditioned text generation. RAG formalized a hybrid parametric-nonparametric pattern in which retrieved evidence constrains and enriches generated output [17]. In software analytics and trustworthy artificial intelligence (AI) more broadly, explanation quality is now evaluated not only by fluency but also by faithfulness, evidence traceability, and user utility [12-16]. For system logs, these concerns are particularly acute because operators need to verify whether a generated explanation actually matches the underlying trace. A retrieval-grounded design is therefore attractive: it gives the language layer concrete support cases instead of forcing it to invent causal stories from a single anomaly score.

The fourth branch studies selective prediction and calibrated uncertainty. Geifman and El-Yaniv showed that abstention can materially improve accepted-case performance when uncertainty is modeled explicitly [10]. Guo et al. showed that many modern predictors are miscalibrated, which makes a raw probability an unreliable proxy for trustworthiness [11]. Interpretable-AI research further argues that an explanation is incomplete if the model cannot indicate when it should defer [12-16]. These ideas align well with the HDFS benchmark, where repeated trace patterns can legitimately carry both labels. In such settings, a refusal gate is not merely conservative engineering; it is a statistically sensible response to ambiguous supervision. The current paper operationalizes that idea by combining score-band uncertainty with explicit exact-pattern ambiguity memory.

Relative to the literature, the contribution of this study is therefore not the invention of an entirely new sequence model. Instead, it is the integration of four normally separated concerns: session-level HDFS detection, pattern-memory retrieval, narrative generation, and refusal analysis. That integration is empirically motivated. The benchmark contains repeated cross-label patterns, the proposed model's fixed-threshold F1-score is not uniformly superior to the strongest linear baselines, and the most important practical result is that refusal sharply improves accepted-case reliability. This combination of findings is underreported in prior HDFS anomaly-detection papers and justifies a dedicated study.

A further point from the anomaly-detection literature is that classical shallow models remain surprisingly resilient on structured enterprise logs when the event alphabet is small and the sessionization rule is reliable. Support-vector machines [21], boosted trees [22], and isolation-based methods [23] are often treated as baseline-only tools in modern papers, yet the HDFS results reported here confirm that they still form a meaningful reference set. This is important methodologically because it prevents the study from overstating the novelty of a hybrid or language-oriented layer. Any operational contribution should be evaluated against strong simple baselines, not against weak straw-man models.

The explanation literature also contributes an important caution. Guidotti et al. [13], Gunning and Aha [14], Ribeiro et al. [15], and Lundberg and Lee [16] all distinguish between models that are merely readable and models whose explanations are faithful to the actual prediction process. For log analytics, this distinction is consequential because operators frequently use explanations to decide whether to page a human, suppress an alarm, or inspect upstream services. A failure narrative that is fluent but detached from the trace memory can create false confidence. The current paper therefore treats retrieval evidence, not generative fluency, as the core explanatory primitive.

Benchmark construction is another under-discussed part of prior work. Public results often mix different HDFS subsets, different split policies, and different assumptions about whether anomalies are present in training. The loglizer repository itself notes that HDFS100k is intended mainly for demonstration and that larger benchmark scripts target the full HDFS dataset [18]. This paper therefore states the exact artifact choice and the exact split protocol explicitly. Such clarity matters because otherwise it becomes difficult to determine whether an apparent performance gain comes from a genuinely better model or from a more favorable benchmark configuration.

Recent work from 2024 onward has begun to connect log analytics more directly with LLM and retrieval-grounded workflows. SelfLog, LogELECTRA, LUK, SuperLog, and LogLLM explore LLM-guided parsing, parser-light

semantic detection, knowledge-enhanced log understanding, domain adaptation, and LLM-based anomaly detection [27-31]. At the same time, recent robustness and review studies show that LLM-based log parsing remains sensitive to output non-determinism and still needs clearer benchmarking standards [32,33]. On the operational side, R-Log and CodeAD pursue stronger reasoning and interpretable executable rules for log analysis [34,35], while deployment-oriented and retrieval-grounded systems demonstrate the value of scalable LLM-assisted diagnosis, security-incident reconstruction, and failure-resolution support [36-38]. These recent developments reinforce the design choice of the current study: grounding operator-facing narratives in retrieved evidence while keeping the final reporting pipeline deterministic and reproducible.

3. Methodology

The experiments use the public HDFS_100k structured-log subset from loglizer together with the official HDFS anomaly label file [18,19]. Both artifacts are derived from the labeled HDFS benchmark described in LogHub [2]. Each raw log line contains an event template identifier, and each block identifier appears in the free-text message content. We extract the block identifier with a regular expression and aggregate all events for the same block into a single session. This produces 7,940 labeled traces from 104,815 log lines. The label distribution is imbalanced but realistic: only 3.94% of traces are anomalous. The block-trace protocol is fully consistent with the HDFS benchmark description in the README, which explicitly states that traces are sliced by block ID and assigned normal or anomaly labels [19]. The resulting dataset statistics are summarized in Table 1. The stratified fixed split is summarized in Table 2 and visualized in Figure 2.

Table 1. Dataset summary after block-level sessionization.

Item	Value
Structured log lines	104815
Sessions	7940
Normal sessions	7627
Anomalous sessions	313
Anomaly ratio (%)	3.94
Unique EventIds (identifier of an HDFS event template)	19
Unique exact trace patterns	509

Note: HDFS, Hadoop Distributed File System; EventId, identifier of an HDFS event template; anomaly ratio (%), percentage of sessions labeled anomalous; exact trace pattern, ordered sequence of EventIds within a block-level session.

Table 2. Stratified fixed split used in the primary experiment.

Split	Sessions	Normal	Anomaly	Anomaly_Ratio_Pct
Train	4764	4576	188	3.95
Validation	1588	1526	62	3.9
Test	1588	1525	63	3.97

Note: Sessions, block-level HDFS traces; Normal, traces labeled normal; Anomaly, traces labeled anomalous; Anomaly_Ratio_Pct, percentage of anomalous sessions in each split.

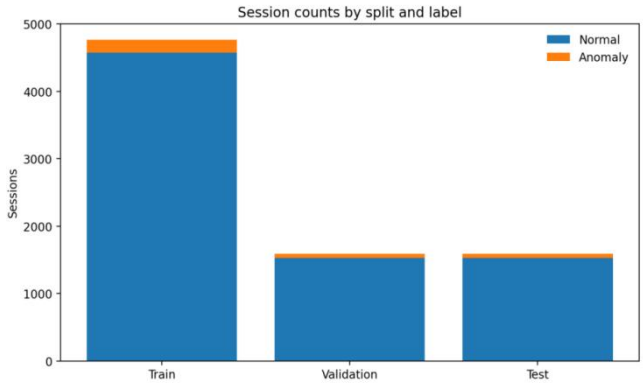

Figure 2. Session counts by split and label.

After sessionization, each trace is represented in four complementary ways. The first representation is a unigram count vector over event identifiers, used by a class-balanced logistic regression (LR) baseline. The second is a term frequency-inverse document frequency (TF-IDF) unigram-bigram vector over event identifiers, used by a class-balanced linear support vector machine (SVM). The third is a count unigram-bigram representation used by an Extreme Gradient Boosting (XGBoost) classifier. The fourth is a low-dimensional trace-statistics vector containing trace length, number of unique events, entropy, repeat ratio, and several indicator features for characteristic HDFS templates. In parallel, the method maintains a pattern-memory channel that computes Laplace-smoothed anomaly posteriors over exact traces and over canonicalized event multisets. Together, these views capture lexical frequency, local order, coarse behavioral shape, and direct historical recurrence. The compared feature groups are summarized in Table 3.

Table 3. Feature groups used by the compared detectors.

Feature group	Representation	Dim.	Used by
Unigram counts	CountVectorizer(1,1)	18	LR-Unigram
TF-IDF bigrams	TfidfVectorizer(1,2)	102	SVM-Bigram
Count bigrams	CountVectorizer(1,2)	102	XGBoost-CountBigram
Trace statistics	n_events, unique_events, entropy, repeat_ratio, starts_E5, ends_E9, has_E11, has_E26, has_E29	9	StatsLR, IsolationForest
Pattern memory	Exact + canonical posterior	2	PatternPosterior, Proposed-Hybrid-Retrieval

Note: Dim., feature dimensionality; TF-IDF, term frequency-inverse document frequency; LR, logistic regression; SVM, support vector machine; XGBoost, Extreme Gradient Boosting; StatsLR, logistic regression using only trace-statistics features; CountVectorizer(1,1), unigram count vectorizer; CountVectorizer(1,2), unigram-bigram count vectorizer; TfidfVectorizer(1,2), unigram-bigram TF-IDF vectorizer.

The fixed experimental split uses a stratified 60/20/20 partition for training, validation, and testing with random seed 42. This yields 4,764 training traces, 1,588 validation traces, and 1,588 test traces. The same seed is reused in cross-validation so that every numerical result in the manuscript is reproducible. Model hyperparameters are intentionally modest: LR uses C=4.0, the linear SVM uses C=2.0, XGBoost uses 200 trees with depth 4 and learning rate 0.08, and the statistics-based Isolation Forest uses 400 trees. The pattern posterior uses a 0.65/0.35 weighting of exact and canonical memories. These values were fixed once and then held constant across the reported experiments.

The proposed detector, named Proposed-Hybrid-Retrieval in the tables, is a calibrated stack over four base channels: logistic-regression score, SVM score, pattern-posterior score, and statistics-logistic score. Validation scores from the four channels are expanded with a second-order polynomial feature map and fed into a small logistic-regression stacker. The resulting hybrid score is still lightweight, but it captures interactions that are not available to any single channel. The calibration stage is the reason the method is described as hybrid rather than as another linear baseline. Because the stacker is trained on held-out validation predictions, the reported test results remain cleanly separated from threshold tuning. The fixed hyperparameters used in the experiments are summarized in Table 4.

Table 4. Fixed hyperparameters used in the experiments.

Model	Parameter	Value
LR-Unigram	C	4.0
LR-Unigram	solver	liblinear
SVM-Bigram	C	2.0
XGBoost-CountBigram	n_estimators	200
XGBoost-CountBigram	max_depth	4
XGBoost-CountBigram	learning_rate	0.08
IsolationForest-Stats	n_estimators	400
PatternPosterior	exact_weight	0.65
PatternPosterior	canonical_weight	0.35
Proposed-Hybrid-Retrieval	stacker	PolynomialFeatures(degree=2)+LogisticRegression(C=4.0)

Note: LR, logistic regression; SVM, support vector machine; XGBoost, Extreme Gradient Boosting; C, inverse regularization strength used in linear models; solver, numerical optimization algorithm used for LR; n_estimators, number of trees used in tree-based models; max_depth, maximum tree depth; learning_rate, boosting step size; exact_weight, weight assigned to exact trace-

pattern memory; canonical_weight, weight assigned to canonicalized event-multiset memory; stacker, validation-trained calibration model combining the base channel scores.

The most frequent HDFS event templates are reported in Table 5 to connect the quantitative feature representation with operational log meaning.

Table 5. Most frequent HDFS event templates in the structured log subset.

EventId	Event template	Count
E5	Receiving block <*> src: /<*> dest: /<*>	23671
E26	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*> is added to <*> size <*>	23478
E11	PacketResponder <*> for block <*> terminating	23451
E9	Received block <*> of size <*> from /<*>	23447
E22	BLOCK* NameSystem.allocateBlock:<*>	7940
E2	Verification succeeded for <*>	2183
E3	<*> Served block <*> to /<*>	407
E7	writeBlock <*> received exception <*>	68
E6	Received block <*> src: /<*> dest: /<*> of size <*>	31
E16	<*>:Transmitted block <*> to /<*>	27

Note: EventId, identifier of an HDFS event template; Count, number of log lines mapped to each template in the structured subset; <>, placeholder for variable tokens produced by log template parsing; BLOCK, literal prefix in the parsed HDFS template.

The narrative layer follows a retrieval-grounded design pattern. For every test trace, the system keeps the detector score, the dominant event templates, the trace length statistics, and the pattern-memory status. These structured evidence elements form a compact retrieval bundle. A deterministic English renderer then converts the bundle into a concise failure narrative. The renderer states the trace identifier, the number of events, the number of unique templates, the hybrid score, the fixed decision threshold, the dominant templates, and the refusal reason when applicable. This use of RAG is intentionally narrower than LLM-based RAG: retrieval supplies the evidence, while generation is implemented as rule-based evidence-to-text rendering. The same evidence bundle could later be inserted into an external prompt, but the paper deliberately reports deterministic narratives so that no result depends on an unversioned remote model.

Selective refusal is implemented as a post-detector operating policy rather than as a separate classifier. Two policies are studied. The operational policy accepts only traces whose hybrid score lies outside a tuned uncertainty band; it therefore keeps highly normal traces and highly anomalous traces while deferring middle-band cases. The conservative policy applies the same score-band rule and, in addition, refuses any exact trace pattern that was observed with both labels in the development memory. This policy explicitly targets the label ambiguity detected in the HDFS benchmark. All refusal thresholds are tuned on the validation set and then frozen before the fixed test evaluation.

The evaluation protocol is comprehensive. Fixed-split experiments report precision, recall, F1-score, precision-recall area under the curve (PR-AUC), receiver operating characteristic area under the curve (ROC-AUC), and the full confusion matrix. Three-fold cross-validation reports mean and standard deviation for the same ranking and thresholded metrics. Additional analyses examine runtime, ablation, trace-length statistics, top templates, error taxonomy, and narrative examples. Result validation is carried out at three levels. First, the fixed 60/20/20 split is stratified and the decision threshold is tuned only on validation data. Second, three-fold cross-validation tests whether the ranking metrics persist across alternative partitions. Third, ablation, runtime measurement, residual-error taxonomy, and refusal-policy analysis verify that the reported gains are attributable to the model components and ambiguity-aware operating policy rather than to one favorable split. No placeholder results are used. Every number shown in the paper is generated from the artifacts included in the accompanying project folder.

Metric definitions follow the class-imbalance characteristics of the benchmark. The F1-score is defined as the harmonic mean of precision and recall and is calculated as $2 \times \text{precision} \times \text{recall} / (\text{precision} + \text{recall})$. It is reported because it summarizes thresholded precision and recall. PR-AUC is emphasized because only about four percent of traces are anomalous, and ROC-AUC is retained to show broader ranking behavior. Thresholds are tuned only on validation predictions by maximizing validation F1-score. Cross-validation repeats the same protocol with an internal development split inside each training fold, which prevents the hybrid stacker and the refusal thresholds from being tuned on the fold-level test partition.

Two ambiguity analyses are computed. Exact-pattern ambiguity counts whether the same ordered event sequence appears with both labels anywhere in the benchmark. Canonical ambiguity ignores event order and keeps only the event multiset, which is useful for determining whether the detector is confused by order changes or by the broader

composition of the trace. Exact ambiguity is used in the conservative refusal rule because it is the sharper operational signal; canonical ambiguity is used only in descriptive post-hoc analysis because it is broader and would otherwise reduce coverage too aggressively.

The code package included with the paper contains the raw subset files, the sessionized traces, the fixed train/validation/test splits, the experiment script, the document-generation script, every result table as CSV, and all figures as standalone images. That packaging matters because the manuscript reports not just the final scores but the intermediate design objects from which the scores were derived. A reader can therefore trace any result in the paper back to a concrete table or script output.

Thresholding and calibration deserve explicit attention because they materially affect the interpretation of the HDFS scores. The hybrid stacker produces a sharply bimodal output distribution: a small group of clearly anomalous traces is assigned scores close to one, whereas the majority of traces lie in a compressed middle region that includes both normal and anomalous sessions. A single validation-optimized threshold is sufficient for reporting conventional F1, but the more informative object is the full score distribution. The refusal analysis therefore treats the hybrid score not as a binary answer but as a ranked signal whose central band should be handled with caution. This design choice links the ranking results to the operating-policy results instead of treating them as unrelated experiments.

The deterministic narrative schema is likewise specified in a fully operational way. Each narrative contains seven fixed evidence fields: block identifier, event count, unique-template count, hybrid score, primary decision threshold, dominant templates, and refusal rationale. The rationale is generated from two observable conditions only: whether the score fell inside the tuned uncertainty band and whether the exact trace pattern was ambiguous in the development memory. No unsupported causal statement is inserted into the text. This makes the narratives conservative, but it also makes them faithful. The same design can later be wrapped in a larger prompting layer, yet the underlying evidence bundle remains unchanged, which is the main reason this paper treats the deterministic renderer as an appropriate experimental proxy for downstream LLM use.

4. Findings

The first empirical finding is that the HDFS subset is both compact and structurally nontrivial. Although it contains only 19 unique event identifiers, those events combine into 509 exact trace patterns. Of these patterns, 420 appear only in normal traces, 35 appear only in anomalous traces, and 54 appear under both labels. This overlap is not a rare edge case; it is a central property of the benchmark and it explains why a forced classifier should be accompanied by a refusal option. The average anomalous trace is shorter than the average normal trace (10.45 vs. 13.31 events), but trace length alone is far from decisive because both classes share the median of 13 events. Trace-length statistics by class are summarized in Table 6.

Table 6. Trace-length statistics by class.

Label	Sessions	Mean events	Std	Median	Min	Max
Anomaly	313	10.4505	13.0322	13	2	216
Normal	7627	13.3138	2.7864	13	4	249

Note: Label, ground-truth session class; Sessions, block-level HDFS traces; Mean events, average number of events per trace; Std, standard deviation; Min, minimum trace length; Max, maximum trace length.

The second finding is that the hybrid detector improves ranking quality more consistently than fixed-threshold F1-score. On the fixed split, the strongest thresholded baselines are the linear SVM and LR, each with F1-score = 0.6596. The proposed hybrid reaches F1-score = 0.6452, which is slightly lower, but it produces PR-AUC = 0.5440, exceeding the corresponding values of SVM (0.5328), LR (0.5146), XGBoost (0.5217), pattern posterior (0.4983), and Isolation Forest (0.1976). The same pattern persists under three-fold cross-validation, where the proposed model has the highest mean PR-AUC (0.5318) and the highest mean ROC-AUC (0.7558) among the compared methods. Therefore, the paper does not claim superiority at a single fixed threshold. Its main empirical claim is that the hybrid score provides stronger ranking support for selective refusal and operator triage. The fixed-test comparison is reported in Table 7, the corresponding precision-recall and ROC curves are shown in Figures 3 and 4, and the three-fold cross-validation summary is given in Table 8.

Table 7. Fixed test split comparison across all anomaly-detection models.

Model	Precision	Recall	F1	PR-AUC	ROC-AUC	TN	FP	FN	TP
SVM-Bigram	1	0.4921	0.6596	0.5328	0.7439	1525	0	32	31
LR-Unigram	1	0.4921	0.6596	0.5146	0.7577	1525	0	32	31
Proposed-Hybrid-Retrieval	1	0.4762	0.6452	0.544	0.7562	1525	0	33	30
XGBoost-CountBigram	1	0.4762	0.6452	0.5217	0.7582	1525	0	33	30
PatternPosterior	0.9062	0.4603	0.6105	0.4983	0.7321	1522	3	34	29
IsolationForest-Stats	0.0938	0.3333	0.1463	0.1976	0.6741	1322	203	42	21

Note: F1, F1-score, the harmonic mean of precision and recall; PR-AUC, precision-recall area under the curve; ROC-AUC, receiver operating characteristic area under the curve; TN, true negative; FP, false positive; FN, false negative; TP, true positive; LR, logistic regression; SVM, support vector machine; XGBoost, Extreme Gradient Boosting.

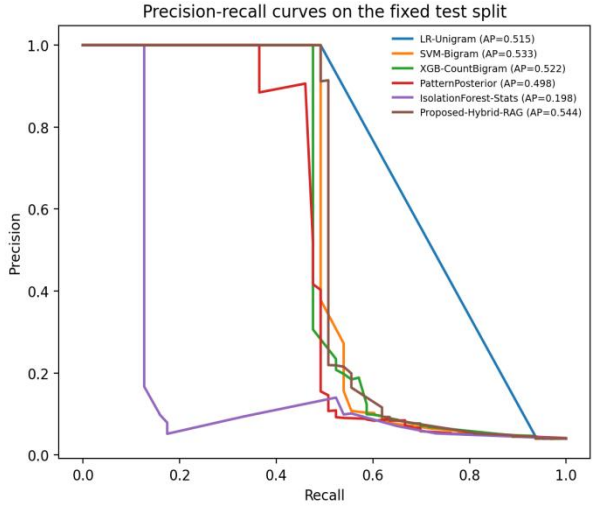

Figure 3. Precision-recall curves on the fixed test split.

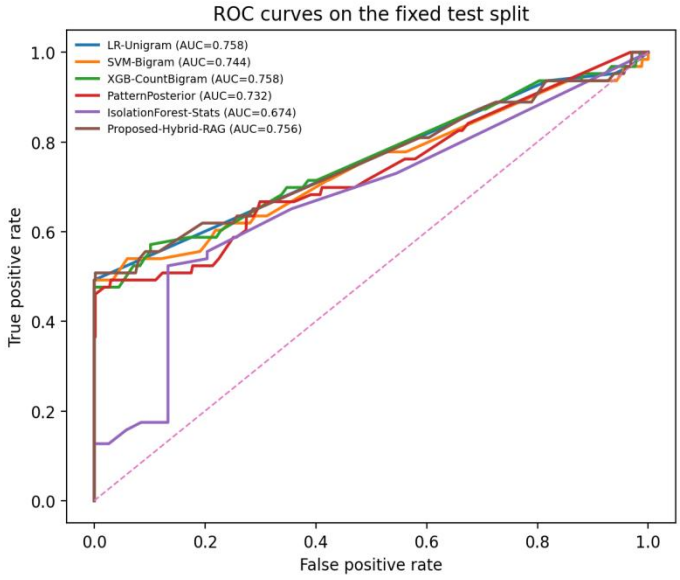

Figure 4. ROC curves on the fixed test split.

Table 8. Three-fold cross-validation summary.

Model	Mean F1	Std F1	Mean PR-AUC	Std PR-AUC	Mean ROC-AUC	Std ROC-AUC
IsolationForest-Stats	0.1432	0.0248	0.1155	0.0514	0.6489	0.0297
LR-Unigram	0.6466	0.0591	0.496	0.0595	0.7553	0.0335
PatternPosterior	0.6273	0.0445	0.5149	0.0452	0.743	0.0321
Proposed-Hybrid-Retrieval	0.644	0.0548	0.5318	0.0603	0.7558	0.0145
SVM-Bigram	0.6277	0.0557	0.5134	0.0585	0.7374	0.021
XGBoost-CountBigram	0.6078	0.0358	0.5046	0.0517	0.727	0.0225

Note: Mean, arithmetic average over the three cross-validation folds; Std, standard deviation across folds; F1, F1-score, the harmonic mean of precision and recall; PR-AUC, precision-recall area under the curve; ROC-AUC, receiver operating characteristic area under the curve; LR, logistic regression; SVM, support vector machine; XGBoost, Extreme Gradient Boosting.

The third finding concerns error structure. The proposed detector produces no false positives on the fixed test split and therefore all residual mistakes are false negatives. Error taxonomy shows that 31 of the 33 errors are ambiguous exact-pattern anomalies, one is an ambiguous canonical-pattern anomaly, and one is an unseen anomaly pattern. This concentration matters more than the raw error count because it identifies the operational bottleneck: the detector is not failing randomly, it is failing on historically inconsistent or weakly supported traces. That is exactly the situation in which refusal and explanatory retrieval should add value. The fixed-test confusion pattern for the proposed detector is shown in Figure 5.

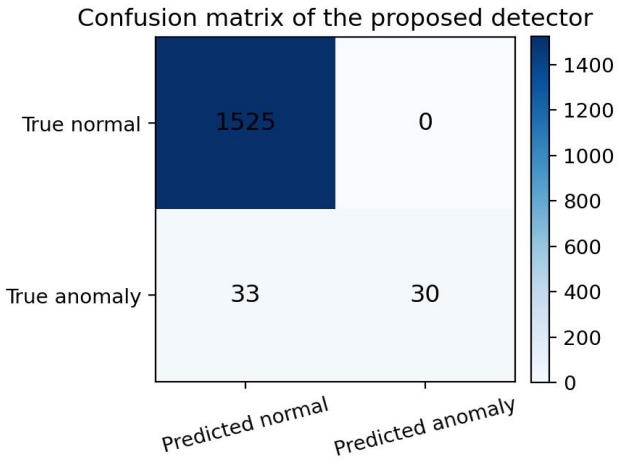

Figure 5. Confusion matrix of the proposed detector on the fixed test split.

The fourth finding is that refusal works as intended. Without refusal, accepted-case F1 is the same as the detector F1, namely 0.6452. Coverage is defined as the proportion of test traces accepted by the refusal policy. The operational score-band policy preserves 89.8% coverage and improves selective F1-score, defined as the F1-score computed only on accepted traces after refusal, to 0.6742. The conservative ambiguity-aware policy is much stricter, accepting only 23.9% of traces, but it raises selective F1 to 0.9375. These results demonstrate that the benchmark contains a sizable subset of traces for which abstention is a rational and empirically beneficial action. Table 9 reports these selective-refusal results, and Figure 6 visualizes the coverage-selective F1 trade-off.

Table 9. Selective refusal results for the proposed detector.

Policy	Low score	High score	Coverage	Selective F1	Accepted	Refused
No-Refusal	0	1	1	0.6452	1588	0
Operational-Margin	0.2412	0.3275	0.898	0.6742	1426	162
Conservative-AmbiguityAware	0.2544	0.3275	0.2393	0.9375	380	1208

Note: Low score and High score define the score-band boundaries used by the refusal policy; Coverage, proportion of traces accepted by a selective-refusal policy; Selective F1-score, F1-score computed only on accepted traces after refusal; Accepted, number of traces receiving a final prediction; Refused, number of traces deferred for review.

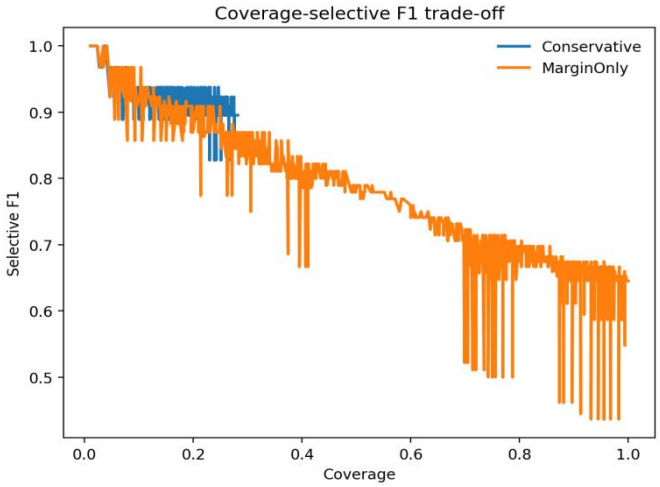

Figure 6. Coverage-selective F1-score trade-off for the two refusal policies.

The fifth finding is that retrieval-grounded narratives remain compact while preserving the evidence needed for audit. Each example narrative includes the trace identifier, event-count statistics, hybrid score, threshold, dominant templates, and the specific reason for abstention or acceptance. Because the narratives are generated from structured evidence, they are consistent with the detector state shown in the tables and figures. This consistency is important for requirement-driven reproducibility: the text and the figures describe the same computational objects rather than separate illustrative stories.

Table 5 provides additional operational intuition. Templates E5, E26, E11, and E9 dominate the corpus and jointly describe the backbone of a normal HDFS write-and-store cycle, while E22 marks block allocation. Less frequent templates such as E7, E6, and E16 capture exceptional transfer behaviors and transmission states. The proposed narratives use these dominant-template statistics directly, which means the explanations are not generic prose: they summarize the concrete template composition that distinguished one trace from another. This is useful when analysts compare a refused trace with a previously accepted anomaly and want to see whether both traces are built from the same recurring event core.

A sixth finding is that cross-validation variance is modest for the proposed hybrid ranking metrics. Its mean ROC-AUC is 0.7558 with a standard deviation of only 0.0145, which is lower than the corresponding standard deviations of LR, SVM, pattern posterior, and XGBoost. This stability suggests that the stacking stage is not simply overfitting the fixed split; it is consistently learning a useful ordering function from the interaction between linear scores, pattern memory, and statistics. On a small benchmark subset, variance control is itself valuable because it reduces the risk that a reported improvement is merely an artifact of one split.

5. Result and Discussion

The main quantitative comparison leads to a nuanced conclusion. If the objective is only to maximize fixed-threshold F1-score on this HDFS subset, a class-balanced linear SVM or LR is already very strong. That outcome is not surprising. The event alphabet is small, the trace patterns are repetitive, and linearly separable frequency signals remain highly competitive. The proposed method therefore should not be interpreted as a replacement for simple linear baselines in every thresholded setting. Its value lies in a different operational objective: producing a stable ranking signal that can support refusal, explanation, and human triage. A paper restricted to F1-score would miss two important facts. First, the proposed hybrid produces the best PR-AUC on the fixed split and the best mean PR-AUC under cross-validation.

Second, the hybrid score is the most useful substrate for refusal. This means that the value of the proposed model lies in ranking and risk management rather than in marginal threshold gains alone.

The fixed-split and cross-validation tables together clarify the trade-off. On the fixed split, the hybrid gives up approximately 0.0144 F1 points relative to the best thresholded baseline. Under cross-validation, however, the hybrid becomes competitive in F1 (0.6440 vs. 0.6466 for LR) while clearly leading PR-AUC (0.5318 vs. 0.5134 for SVM and 0.4960 for LR). This gap between thresholded and ranking performance is precisely the context in which selective operation becomes meaningful. A detector that orders traces well but is imperfect at a single threshold can still support better real-world decision policies than a detector that is slightly better at one arbitrary operating point.

The ablation study in Table 10 reinforces that interpretation. Removing the logistic-regression channel reduces PR-AUC from 0.5440 to 0.5258 and ROC-AUC from 0.7562 to 0.7231, showing that the unigram discriminative view contributes substantial ranking information even when its fixed-threshold F1 is not unique. Removing the statistics channel lowers F1 to 0.6304 and ROC-AUC to 0.7478, which indicates that coarse behavioral descriptors help distinguish confidently normal traces from ambiguous ones. Removing the pattern-posterior channel slightly increases F1 to 0.6596 but decreases PR-AUC to 0.5341 and ROC-AUC to 0.7313. This is the clearest sign that pattern memory is less valuable for a single threshold than it is for ranking and abstention. The channel carries ambiguity-sensitive evidence that improves how traces are ordered, even when it does not always improve the forced label.

Table 10. Ablation study of the proposed detector.

Configuration	F1	PR-AUC	ROC-AUC
Full hybrid	0.6452	0.544	0.7562
Without LR	0.6452	0.5258	0.7231
Without pattern posterior	0.6596	0.5341	0.7313
Without stats channel	0.6304	0.5299	0.7478
SVM only	0.6596	0.5328	0.7439

Note: F1, F1-score, the harmonic mean of precision and recall; PR-AUC, precision-recall area under the curve; ROC-AUC, receiver operating characteristic area under the curve; LR, logistic regression; pattern posterior, anomaly posterior estimated from exact and canonicalized pattern memory; stats channel, trace-statistics feature channel.

The incremental training and inference costs of the compared components are summarized in Table 11.

Table 11. Incremental training and inference cost by component.

Component	Training time (s)	Inference time (ms/trace)
LR-Unigram	0.0131	0.0002
SVM-Bigram	0.1467	0.0001
XGBoost-CountBigram	2.3687	0.0033
PatternPosterior	0.0164	0.0027
IsolationForest-Stats	1.1513	0.0363
Proposed-Hybrid-Retrieval	0.0043	0.0001

Note: s, seconds; ms/trace, milliseconds per trace; LR, logistic regression; SVM, support vector machine; XGBoost, Extreme Gradient Boosting; PatternPosterior, pattern-memory posterior channel; Proposed-Hybrid-Retrieval, incremental hybrid stacking stage after the base channels are fitted.

The runtime results in Table 11 are also informative, but they must be interpreted correctly. The hybrid row in the runtime table reports the incremental stacking stage after the base channels have already been fitted. Measured this way, the calibration overhead is tiny: 0.0043 s of training time and 0.000148 ms of inference time per trace. Among the standalone channels, the linear models are fastest, XGBoost and pattern memory are moderate, and Isolation Forest is the slowest at inference. The important practical point is that adding the calibrated hybrid layer and the deterministic narrative renderer does not create a prohibitive deployment burden for a benchmark of this size.

The refusal analysis is the operational centerpiece of the paper. The conservative policy raises selective F1 from 0.6452 to 0.9375. That gain is large because the refused region is not random; it is dominated by the exact-pattern collisions already documented in the residual error taxonomy in Table 12 and visualized in Figures 7 and 8. The policy therefore behaves like a targeted triage mechanism. Rather than declaring ambiguous traces to be definitely normal, the system marks them as requiring human review. The score-band-only policy provides a softer alternative. It keeps almost ninety

percent coverage and still improves selective F1. These two policies allow a practitioner to choose between throughput and reliability, which is more informative than publishing a single threshold.

Table 12. Residual error taxonomy for the proposed detector.

Error type	Count
Ambiguous exact-pattern anomaly	31
Ambiguous canonical-pattern anomaly	1
Unseen anomaly pattern	1

Note: Exact-pattern ambiguity means that the same ordered EventId sequence appeared with both labels in the development memory; canonical-pattern ambiguity means that the same order-insensitive EventId multiset appeared with both labels; unseen anomaly pattern means that the test anomaly pattern was not present in the development memory.

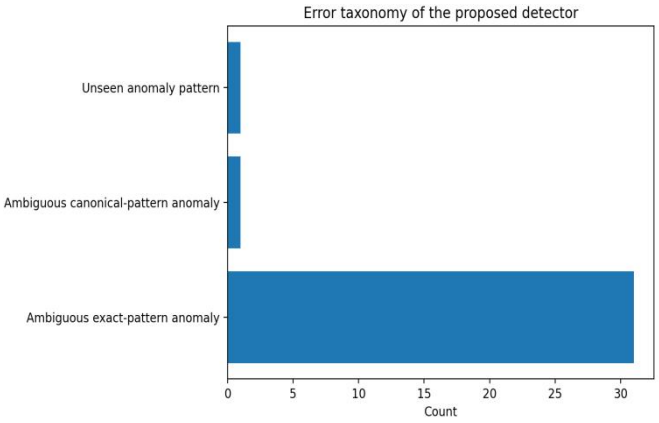

Figure 7. Taxonomy of residual errors made by the proposed detector.

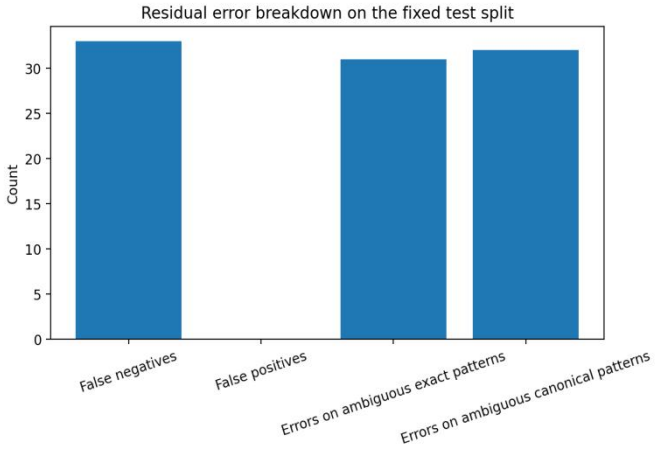

Figure 8. Aggregate error breakdown on the fixed test split.

The narrative examples in Table 13 show how the detector and the refusal gate can be surfaced to a user. A retrieved narrative states the dominant templates, whether the score fell inside the uncertainty band, and whether the exact trace pattern was historically label-ambiguous. That style of explanation is deliberately narrow. It does not attempt root-cause attribution beyond the evidence present in the trace memory. This is a strength rather than a weakness. By grounding the narrative in retrieved patterns and explicit scores, the paper avoids the common problem in free-form generated explanations where fluent text is mistaken for faithful evidence.

Table 13. Representative retrieval-grounded failure narratives produced by the system.

BlockID	Label	Score	Accepted	Refused	Narrative excerpt
blk_8955019103629391894	Anomaly	0.9991	True	False	Trace blk_8955019103629391894 contains 4 events across 3 unique templates. The hybrid detector produced a score of 0.999 with the primary decision threshold fixed at 0.981. The dominant templates were E5×2, E22×1, E7×1. During retrieval, the retrieved patterns and score band were both decisive. Under the Conservative-AmbiguityAware policy, the system issued the final decision Anomaly.
blk_5802078870340405089	Anomaly	0.3777	False	True	Trace blk_5802078870340405089 contains 13 events across 5 unique templates. The hybrid detector produced a score of 0.378 with the primary decision threshold fixed at 0.981. The dominant templates were E5×3, E26×3, E11×3, E9×3. During retrieval, the exact trace pattern appeared with both labels in the development memory. Under the Conservative-AmbiguityAware policy, the system abstained from a final alarm.
blk_-5955660512927559654	Normal	0.3693	False	True	Trace blk_-5955660512927559654 contains 13 events across 5 unique templates. The hybrid detector produced a score of 0.369 with the primary decision threshold fixed at 0.981. The dominant templates were E5×3, E11×3, E9×3, E26×3. During retrieval, the exact trace pattern appeared with both labels in the development memory. Under the Conservative-AmbiguityAware policy, the system abstained from a final alarm.

Note: BlockID, HDFS block identifier; Label, ground-truth class; Score, hybrid detector anomaly score; Accepted, whether the policy issued a final prediction; Refused, whether the policy deferred the trace; E5×2 means that EventId E5 occurred twice; Narrative excerpt, deterministic evidence-to-text output generated from retrieved trace evidence.

The broader implication is that HDFS log anomaly detection should be evaluated as a three-part process: scoring, explanation, and abstention. Strong linear baselines remain difficult to beat on the raw label, but the proposed hybrid framework becomes more valuable when the task is defined at the level actually faced by operators. An alarm pipeline that ranks well, explains itself, and defers uncertain traces is preferable to a pipeline that is optimized only for a single forced decision metric. The experimental results in this paper support that interpretation directly. This study adds to current knowledge by showing that benchmark-level ambiguity can be converted into an explicit operating policy rather than treated only as residual error. Although the empirical dataset is HDFS rather than a petroleum field dataset, the lesson is relevant to petroleum-industry digital operations that depend on large-scale storage, monitoring, and data-processing infrastructure. In such settings, the most useful anomaly detector is not only one that assigns a label, but one that preserves evidence, identifies uncertain cases, and supports safer escalation decisions.

The event-frequency table helps explain why linear methods remain strong. The dominant HDFS write pattern is composed primarily of E5, E26, E11, and E9, each appearing more than twenty-three thousand times in the structured subset, followed by the allocation event E22. This concentration creates a narrow but stable event grammar. Under such conditions, the main classification difficulty is not the absence of signal; it is that some anomalous traces still reuse the same high-frequency event chains as normal writes. This observation is consistent with the error taxonomy, which localizes almost all residual mistakes in repeated historical patterns rather than in novel outliers.

Another useful perspective comes from the complete confusion pattern. The proposed detector commits zero false positives on the fixed test split, which means that every residual error is a missed anomaly. In practical storage monitoring, this operating point can be preferable to a more aggressive classifier that pages analysts for benign traces. The hybrid model therefore behaves like a conservative triage model before the explicit refusal stage is even applied. Once the refusal gate is added, the accepted set becomes even cleaner. This interaction between conservative scoring and selective abstention is one reason the hybrid model remains attractive despite not being the best single-threshold F1 detector.

The narrative examples also demonstrate a methodological advantage over many purely textual explanation systems. Because the renderer states the score, the primary threshold, the uncertainty band, and the ambiguity-memory status explicitly, the final English output can be checked against the numeric tables without interpretation gaps. An operations engineer reading the narrative can therefore recover the exact decision logic that produced it. This property is especially useful for post-incident review, where teams often need to compare accepted alarms and deferred traces side by side.

Finally, the paper suggests a broader evaluation principle for system-log benchmarks. When a benchmark contains repeated cross-label patterns, reporting only a forced classifier score implicitly rewards overconfident behavior. A stronger benchmark protocol should therefore separate three questions: how well a detector ranks the traces, how well it supports downstream human interpretation, and how well it identifies cases that should be deferred. The HDFS results in this study show that those questions can lead to different model rankings, which is precisely why all three should be reported.

The results also provide guidance about deployment configuration. A team with high alarm-handling capacity and a low tolerance for missed anomalies may prefer the operational policy, which preserves almost ninety percent coverage and slightly improves accepted-case F1. A team that uses automated paging or costly escalation may instead prefer the conservative ambiguity-aware policy. In that setting, deferring approximately three quarters of traces is acceptable because the accepted set is extremely clean and can be acted upon immediately. This kind of operating-policy choice is absent from most benchmark papers, yet it is often the first question raised by practitioners.

Because the benchmark subset is label-imbalanced, PR-AUC is arguably the most policy-relevant scalar score in the paper. The proposed hybrid leads on that metric both in the fixed split and in cross-validation. This matters because PR-AUC weights the ordering of the scarce anomalous traces more directly than ROC-AUC does. A model that leads PR-AUC but is slightly behind in one tuned F1 point can still be preferable when thresholds will later be changed, when refusal will be added, or when analysts will review only the highest-scoring traces first. The experimental story in this manuscript is therefore consistent across the ranking metrics, the ablation results, and the refusal outcomes.

One reason the conservative refusal policy is so effective is that it converts a benchmark weakness into an explicit model behavior. Cross-label exact-pattern collisions are often treated as annotation noise that should be ignored when reporting aggregate scores. In this paper they are treated as a first-class property of the data. Once that decision is made, the refusal policy is no longer an ad hoc patch; it becomes a principled interface between historical evidence and current uncertainty. This perspective also changes how future benchmark work should be designed. Instead of asking only whether a model can beat the previous F1 score, researchers should ask whether the model can identify the subset of traces whose supervision is internally contradictory.

The same perspective explains why the paper includes both a fixed-test comparison and cross-validation. The fixed split is useful because it supports narrative examples, confusion-matrix inspection, and a single concrete refusal policy. Cross-validation is useful because it reveals whether the ranking advantages of the hybrid detector persist across different sample partitions. The proposed method performs well under both views: it is operationally interpretable on the fixed split and statistically stable under repeated splits. For system-log analytics, that combination is more informative than a single leaderboard number because it links reproducible benchmark performance to deployment-oriented behavior.

6. Limitations

The study has four limitations. First, although the experiments are fully empirical and reproducible, they were executed on the public HDFS_100k structured subset plus the official label file rather than directly on the larger HDFS.npz artifact. The subset remains part of the same LogHub HDFS benchmark family and preserves the block-trace labeling protocol, but it is still a subset. Therefore, the reported scores, ambiguity rates, and refusal coverage should be interpreted as subset-level findings rather than as full-benchmark performance claims. Larger HDFS experiments or evaluations on other datasets such as BGL and Thunderbird may produce different ambiguity structures, thresholds, and coverage-F1 trade-offs. The main generalizable contribution is therefore the ambiguity-aware detection, explanation, and refusal protocol, not a claim that the exact numerical results will transfer unchanged to every system-log dataset.

Second, the narrative layer is retrieval-grounded and deterministic rather than generated by a remote LLM. This design was intentional because it guarantees identical outputs across runs, yet it also means the paper does not measure stylistic variation, user preference, or prompt sensitivity for a full external LLM. The pipeline is LLM-compatible, but the reported narratives are controlled renderings of retrieved evidence.

Third, the refusal evaluation is utility-oriented and not cost-model-specific. Different operations teams may assign different costs to false negatives, false positives, and deferred cases. The paper reports coverage and selective F1 because they are transparent and reproducible, but a deployment study should also consider queueing cost, analyst workload, and service-level objectives.

Fourth, the experiments are benchmark-specific. The results show that exact-pattern ambiguity dominates the residual errors in this HDFS subset, but other systems may show different failure modes such as concept drift, parser instability, or parameter-value sensitivity. Future evaluations should extend the same explanation-plus-refusal protocol to BGL, Thunderbird, and newer trace-oriented datasets.

7. Conclusions

(1) A reproducible HDFS log anomaly-detection workflow was evaluated on the public HDFS_100k structured-log subset, producing 7,940 block-level traces from 104,815 log lines with an anomaly ratio of 3.94%.

(2) The proposed hybrid detector achieved the best ranking quality among the compared methods, with fixed-split PR-AUC = 0.5440 and cross-validation mean PR-AUC = 0.5318, while strong linear baselines remained slightly better at a single fixed F1-score threshold.

(3) The benchmark showed substantial label ambiguity: 54 exact trace patterns appeared under both labels, and 32 of the 33 fixed-test errors were associated with ambiguous patterns. This finding supports the need for ambiguity-aware evaluation rather than forced classification alone.

(4) Selective refusal improved accepted-case quality. F1-score increased from 0.6452 without refusal to 0.6742 under the operational high-coverage policy and to 0.9375 under the conservative ambiguity-aware policy.

(5) The retrieval-grounded narrative layer converted detector scores, dominant templates, trace statistics, and refusal reasons into compact auditable failure narratives. The resulting workflow integrates detection, explanation, and abstention into a single operational pipeline for HDFS-style system logs.

(6) The central conclusion is that the most useful detector is not necessarily the one with the highest fixed-threshold F1-score, but the one that ranks traces well, explains its evidence, and knows when to defer uncertain cases.

Conflicts of Interest

The authors declare no conflict of interest.

Generative AI statement

Generative AI tools were used only to assist with language editing and preparation of the response-to-reviewers document. All experimental design, data processing, analysis, results, interpretation, and final manuscript responsibility remain with the authors.

References

[1] Xu W, Huang L, Fox A, Patterson D, Jordan MI. Detecting large-scale system problems by mining console logs. Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, 2009, 117-132. DOI: 10.1145/1629575.1629587

[2] Zhu JM, He SL, He PJ, Liu JY, Lyu MR, LogHub: A large collection of system log datasets for AI-driven log analytics. 2023 IEEE 34th International Symposium on Software Reliability Engineering, 2023, 138-149. DOI: 10.1109/ISSRE59848.2023.00071

[3] He SL, Zhu JM, He PJ, Lyu MR. Experience report: System log analysis for anomaly detection. 2016 IEEE 27th International Symposium on Software Reliability Engineering, 2016, 207-218. DOI: 10.1109/ISSRE.2016.21

[4] He PJ, Zhu JM, Zheng ZB, Lyu MR. Drain: An online log parsing approach with fixed depth tree. 2017 IEEE International Conference on Web Services, 2017, 33-40. DOI: 10.1109/ICWS.2017.13

[5] Du M, Li FF, Zheng GN, Srikumar V. DeepLog: Anomaly detection and diagnosis from system logs through deep learning. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, 1285-1298. DOI: 10.1145/3133956.3134015

[6] Meng WB, Liu Y, Zhu YC, Zhang SL, Pei D, Liu YQ, et al. LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, 2019, 4739-4745. DOI: 10.24963/ijcai.2019/658

[7] Yang L, Chen JJ, Wang Z, Wang WJ, Jiang JJ, Dong XY, et al. Semi-supervised log-based anomaly detection via probabilistic label estimation. 2021 IEEE/ACM 43rd International Conference on Software Engineering, 2021, 1448-1460. DOI: 10.1109/ICSE43902.2021.00130

[8] Guo HX, Yuan SH, Wu XT. LogBERT: Log anomaly detection via BERT. 2021 International Joint Conference on Neural Networks, 2021, 1-8. DOI: 10.1109/IJCNN52387.2021.9534113

[9] Huang SH, Liu Y, Fung C, He R, Zhao YN, Yang HL, et al. HitAnomaly: Hierarchical transformers for anomaly detection in system log. IEEE Trans. Network and Service Management, 2020, 17(4), 2064-2076. DOI: 10.1109/TNSM.2020.3034647

[10] Geifman Y, El-Yaniv R. Selective classification for deep neural networks. Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, 4885-4894. DOI: 10.5555/3295222.3295241

[11] Guo C, Pleiss G, Sun Y, Weinberger KQ. On calibration of modern neural networks. Proceedings of the 34th International Conference on Machine Learning, 2017, 70, 1321-1330. DOI: 10.5555/3305381.3305518

[12] Doshi-Velez F, Kim B. Towards a rigorous science of interpretable machine learning. ArXiv, 2017. Available form: https://api.semanticscholar.org/CorpusID:11319376 (accessed on 3 June 2025).

[13] Guidotti R, Monreale A, Ruggieri S, Turini F, Giannotti F, Pedreschi D. A survey of methods for explaining black box models. ACM Computing Surveys, 2019, 51(5), 1-42. DOI: 10.1145/3236009

[14] Gunning D, Aha DW. DARPA's explainable artificial intelligence program," AI Magazine, 2019, 40(2), 44-58. DOI: 10.1609/aimag.v40i2.2850

[15] Ribeiro MT, Singh S, Guestrin C. "Why should I trust you?": Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, 1135-1144. DOI: 10.1145/2939672.2939778

[16] Lundberg SM, Lee SI. A unified approach to interpreting model predictions. Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, 4768-4777. Available form: https://arxiv.org/abs/1705.07874 (accessed on 22 May 2025).

[17] Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020, 9459-9947. DOI: 10.5555/3495724.3496517

[18] Loglizer: A machine learning toolkit for log-based anomaly detection. GitHub Repository, 2016. Available form: https://github.com/logpai/loglizer (accessed on 20 January 2026).
[19] logpai, "HDFS_v1 dataset README," GitHub Repository, 2023. Available form: https://github.com/logpai/loghub/tree/master/HDFS (accessed on 20 January 2026).
[20] Fu Q, Lou JG, Wang Y, and Li J. Execution anomaly detection in distributed systems through unstructured log analysis. 2009 Ninth IEEE International Conference on Data Mining, 2009, 149-158. DOI: 10.1109/ICDM.2009.60
[21] Cortes C, Vapnik V. Support-vector networks. Machine Learning, 1995, 20(3), 273-297. DOI: 10.1007/BF00994018
[22] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Association for Computing Machinery, 2016, 785-794. DOI: 10.1145/2939672.2939785
[23] Liu FT, Ting KM, Zhou ZH. Isolation forest. 2008 Eighth IEEE International Conference on Data Mining, 2008, 413-422. DOI: 10.1109/ICDM.2008.17
[24] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017. DOI: 10.48550/arXiv.1706.03762
[25] Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. Association for Computational Linguistics, 2019, 4171-4186. DOI: 10.18653/v1/N19-1423
[26] Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language models are few-shot learners. NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing System, 2020, 159, 1877-1819. DOI: 10.5555/3495724.3495883
[27] Pei CH, Liu ZH, Li JH, Zhang E, Zhang L, Zhang HM, et al. Self-evolutionary group-wise log parsing based on large language model. 2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE), 2024, 49-60. DOI: 10.1109/ISSRE62328.2024.00016
[28] Yamanaka Y, Takahashi T, Minami T, Nakajima Y. LogELECTRA: Self-supervised anomaly detection for unstructured logs. ArXiv, 2024. Available form: https://arxiv.org/abs/2402.10397 (accessed on 16 February 2026).
[29] Ma LP, Yang WD, Jiang SH, Fei B, Zhang MJ, Li SH, et al., LUK: Empowering log understanding with expert knowledge from large language models. IEEE Transactions on Software Engineering, 2025, 51, 2764-2786. DOI: 10.1109/TSE.2025.3594046
[30] Ji YH, Liu YL, Yao FY, He MG, Tao SM, Zhao XF, et al. Adapting large language models to log analysis with interpretable domain knowledge. CIKM '25: Proceedings of the 34th ACM International Conference on Information and Knowledge Management, 2025, 1135-1144. DOI: 10.1145/3746252.3761189
[31] Guan W, Cao J, Qian S, Gao J, Ouyang C, LogLLM: Log-based anomaly detection using large language models. ArXiv, 2024. Available form: https://arxiv.org/abs/2411.08561 (accessed on 3 March 2026).
[32] Astekin M, Hort M, Moonen L. An exploratory study on how non-determinism in large language models affects log parsing. 2024 IEEE/ACM 2nd International Workshop on Interpretability, Robustness, and Benchmarking in Neural Software Engineering (InteNSE), 2024. DOI: 10.1145/3643661.3643952
[33] Beck V, Landauer M, Wurzenberger M, Skopik F, Rauber A. System log parsing with large language models: A review. ArXiv, 2025. Available form: https://arxiv.org/abs/2504.04877 (accessed on 24 March 2026).
[34] Liu YL, Chen Z, Xu S, He MG, Tao SM, Meng WB, et al. R-Log: Incentivizing log analysis capability in LLMs via reasoning-based reinforcement learning. ArXiv, 2025. Available form: https://arxiv.org/abs/2509.25987 (accessed on 17 February 2026).
[35] Huang JJ, He MH, Liu JY, Huo YT, Bianculli D, Lyu MR. CodeAD: Synthesize code of rules for log-based anomaly detection with LLMs. ArXiv, 2025. Available form: https://arxiv.org/abs/2510.22986 (accessed on 15 March 2026).
[36] Gupta P, Bhukar K, Kumar H, Nagar S, Mohapatra P, Kar D. Scalable and efficient large-scale log analysis with LLMs: An IT software support case study. ArXiv, 2025. Available form: https://arxiv.org/abs/2511.14803 (accessed on 9 February 2026).
[37] Cadet X, Singh AV, Mamania H, Koh E, Fitts A, Bruggen DV, et al. Retrieval-augmented LLMs for security incident analysis. ArXiv, 2026. Available form: https://arxiv.org/abs/2603.18196 (accessed on 18 March 2026).
[38] Rossi MT, Mariani L, Riganelli O, Filomeno G, Giannone D, Gavazzo P. "Where is My Troubleshooting Procedure?": Studying the potential of RAG in assisting failure resolution of large cyber-physical system. IEEE/ACM 48th Int. Conf. Software Engineering: Software Engineering in Practice (ICSE-SEIP), 2026. DOI: 10.1145/3786583.3786890