GAS Journal of Engineering and Technology (GASJET)

Volume 3 | Issue 7, 2026 Homepage: https://gaspublishers.com/gasjet-home/ ISSN: 3048-5800

Evidence-Constrained Log Anomaly Detection and LLM-Ready Failure Narrative Generation with Selective Refusal on LogEval 2024

Victor Cui

Computer Science, University of California, Davis, Davis, CA, USA

Received: *20.06.2026* | ***Accepted:*** *25.07.2026* | ***Published:*** *27.07.2026*

***Corresponding Author:** Victor Cui

DOI: *10.5281/zenodo.21623891*

Abstract **Original Research Article**

This paper presents ECLA, an evidence-constrained, LLM-ready pipeline for LogEval 2024-style log intelligence on the HDFS-2k corpus. The study links four tasks: log parsing, window-level anomaly detection, root-cause ranking, and deterministic failure-narrative generation with selective refusal. The corpus contains 2,000 HDFS log lines, 14 structured event templates, six components, 1,994 block identifiers, and 80 warning lines. Rather than asking a language model to infer causes from raw text, ECLA extracts parser templates, block and node evidence, and window-level event signals; it then verbalizes a diagnosis only when the evidence packet supports the claim. Evaluation used a fixed seed, non-overlapping 10-line windows, a stratified train/test split for anomaly detection, and a separate anomalous-window protocol for root-cause ranking. The ECLA parser achieved perfect mapped parsing accuracy while retaining near-gold template compactness. For anomaly detection, ECLA achieved F1 = 0.926 and PR-AUC = 0.994, outperforming the learned TF-IDF baselines. For root-cause ranking, ECLA obtained Top-1 = 0.935 and Top-2 = 1.000. The selective narrative layer answered 89 actionable windows and refused 111 non-actionable windows; under the study's rule-based claim verifier, it produced no unsupported diagnoses. These results support selective refusal as a reproducible control for evidence-constrained log explanations, while not constituting an evaluation of a deployed LLM or human operator trust.

Keywords: Log anomaly detection, root-cause analysis, evidence-grounded generation, selective refusal, HDFS.

Introduction

Large distributed systems rely on logs as the most persistent record of runtime behavior. HDFS logs are especially useful because a single block operation can leave traces across DataNode, FSNamesystem, packet responder, dataset, and scanner components. Public log collections such as LogHub make this style of evaluation repeatable across systems and failure patterns (Zhu et al., 2023), while log parsing benchmarks establish the common practice of turning raw log messages into event templates before detection or diagnosis (Zhu et al., 2019). The broader reliability-engineering literature treats log analysis as a pipeline that spans parsing, compression, anomaly detection, failure prediction, and diagnosis (He et al., 2021). Earlier studies on supercomputer and distributed-system logs showed that operational failures often appear as rare message types or unusual event sequences rather than as a single obvious error token (Oliner & Stearley, 2007; Xu et

al., 2009; Fu et al., 2009).

Recent work confirms that the value of log intelligence lies in the chain rather than in a single score. Retrieved normal prototypes have been used to explain failure-injection anomalies in OpenStack logs (Xin, 2025), while cost-aware routing has been studied across LogEval parsing, detection, diagnosis, and summarization tasks (Li et al., 2026). Self-supervised LogBERT-style modeling provides a complementary sequence-learning view on a reproducible HDFS-like benchmark (Xin, 2026b). These studies motivate a design in which representation, scoring, evidence packaging, and explanation are evaluated separately.

The LogEval 2024 framing is valuable because it joins tasks that are often studied separately. A parser that looks accurate may produce features that are inconvenient for anomaly detection; an anomaly detector may flag a window without explaining the root cause; and a natural-language generator may sound convincing while adding facts that do not appear in the logs. Treating these tasks as one chain forces every stage to pass evidence forward. This paper keeps that chain explicit so that the final narrative can be traced back to concrete lines and templates.

Parsing remains the first step because raw logs mix constants and variable parameters. Classical methods include frequent-pattern clustering in SLCT (Vaarandi, 2003), iterative partitioning in IPLoM (Makanju et al., 2009), LCS-based streaming parsing in Spell (Du & Li, 2016), and fixed-depth tree parsing in Drain (He et al., 2017). These parsers established the key idea that a template should retain stable message words while abstracting block identifiers, IP addresses, ports, paths, and numeric values. However, parsing accuracy alone can hide over-splitting. A parser that produces many redundant clusters may still map each line to the correct event family. For this reason, the present evaluation reports mapped parsing accuracy together with ARI, NMI, template count, and runtime.

After parsing, log anomaly detection can be formulated as sequence modeling, semantic template modeling, or supervised classification. DeepLog used recurrent prediction over log keys for anomaly detection and diagnosis (Du et al., 2017). LogAnomaly added semantic template representations to address sequential and quantitative anomalies (Meng et al., 2019). LogRobust used attention over semantic log representations to handle unstable logs (Zhang et al., 2019), and LogBERT used self-supervised BERT-style objectives for normal-sequence modeling (Guo et al., 2021). Experience reports also show that lightweight supervised or unsupervised baselines remain useful when the dataset is small and the anomaly definition is explicit (He et al., 2016). The present work therefore compares symbolic evidence rules, TF-IDF linear models, isolation forest, and an evidence-blended detector rather than relying on a single model family. The inclusion of a language-guided feature-selection study for intrusion detection provides an adjacent security precedent for treating feature choice as an auditable layer rather than as an invisible preprocessing step (Li & Lu, 2025).

LLMs create a new opportunity and a new risk. Transformer architectures (Vaswani et al., 2017), BERT-style contextual encoders (Devlin et al., 2019), and large generative models (Brown et al., 2020) can translate technical evidence into readable explanations. Retrieval-augmented generation reduces unsupported claims by grounding generated text in retrieved evidence (Lewis et al., 2020). Yet summarization research shows that fluent outputs can be factually inconsistent with their inputs (Maynez et al., 2020; Kryscinski et al., 2020). Budgeted multi-hop retrieval shows that evidence acquisition itself can be treated as a policy with an explicit resource limit (Su et al., 2025). For operations teams, this risk is not cosmetic: a summary that asserts an unsupported root cause can lead to a wrong repair action. ECLA therefore treats generation as a selective decision. The system refuses to produce a root-cause narrative when the detector, RCA scorer, and evidence packet do not support a definite claim. The metrics include hallucination rate, refusal rate, answer accuracy, coverage, and risk.

Evidence must also be communicated in a form that supports review. Scientific-search evidence cards emphasize visible source and ranking cues (Jin, 2025b), and work on dark-pattern microcopy shows why explanation language should be inspected rather than assumed to be benign (Xu et al., 2025). These interface studies do not validate ECLA, but they support the decision to expose evidence and refusal rather than only a generated sentence.

This paper makes three contributions. First, it defines a compact LogEval-style HDFS protocol that covers log parsing, anomaly detection, RCA ranking, and failure narrative generation on the same data split. Second, it reports empirical results from local execution across all tasks rather than relying on qualitative claims. Third, it evaluates selective refusal as a measurable control mechanism. The study uses interpretable evidence features, logistic models, SVC, naive Bayes, and isolation forest implemented with standard machine-learning tooling (Chandola et al., 2009; Pedregosa et al., 2011), and it reports evidence-grounded outputs in the spirit of local explanation and claim-verification methods (Ribeiro et al., 2016).

A further motivation is that operations teams need concise answers under time pressure, but the text of a single HDFS window may be routine, mixed, or only weakly suggestive. In such cases, forcing a language model to return a diagnosis is not neutral; it can turn missing evidence into a confident but wrong explanation. The protocol therefore separates three decisions that are often merged in log-assistant demonstrations: whether a window is actionable, which failure family is best supported, and whether the evidence is strong enough to verbalize. This separation makes refusal measurable rather than subjective, and it makes hallucination a function of claim support rather than writing style. It also avoids confusing benchmark transfer with proof of deployment transfer: domain-adaptation theory shows that approximately shared features can help across domains, but only under explicit assumptions about the shared representation (Zhong et al., 2025).

The paper uses the term LLM augmentation to describe this controlled interface between evidence extraction and natural-language output. The method does not ask the language component to discover root causes from raw text alone. Instead, the parser and detectors produce an evidence packet containing the event family, component, block id, node id, and confidence signals. The narrative component then converts that packet into a concise incident statement or refuses. This design follows the same engineering intuition as retrieval-grounded generation: generation should be a final communication layer, not an unchecked source of facts. The related design-critic literature further suggests that machine-generated feedback should remain subordinate to human judgment rather than being treated as an authoritative assessment (Li et al., 2025).

Materials and Methods

Dataset and corpus validation

The evaluation used the HDFS-2k corpus as a fixed LogEval-style benchmark. Each line contains date, time, process id, level, component, content, event id, and event template. A timestamp was reconstructed from the date and time fields, and every content line was scanned for block identifiers and node IPs. The raw corpus has 2,000 lines and 14 event templates. The structured event catalog contains routine block receiving, allocation, packet responder termination, verification, and block-map updates, as well as actionable maintenance or failure evidence such as serving exceptions, namespace invalidation, local deletion, delete instructions, replication instructions, and transfer recovery. Table 1 summarizes the corpus and Table 2 lists the event catalog used by the experiment.

Several consistency checks were applied before modeling. The raw log file and structured table were required to have the same number of lines. Every row was required to contain a block identifier, which is expected for the HDFS sample because all 14 templates describe block operations. The INFO and WARN counts were checked against the event catalog, and all WARN lines mapped to the serving-exception template. These checks

connect the later metrics to the actual corpus content and prevent the method from using a task definition that is inconsistent with the data.

Window construction and actionable taxonomy

The task construction used 10-line non-overlapping windows, producing 200 windows. A window was labeled actionable when it contained at least one event from the actionable taxonomy. The window root-cause label was the dominant actionable class in the window; ties were resolved by an operational priority order: serving exception, namespace invalidation, local deletion, delete instruction, replication instruction, and transfer recovery. This construction yields a deterministic and auditable target because the label can be traced directly to log message semantics. Table 3 reports the window label distribution.

The actionable taxonomy was chosen to match the event templates present in the corpus. Serving exception corresponds to WARN DataXceiver messages that fail while serving a block. Namespace invalidation corresponds to NameSystem.delete messages that add a block to an invalidSet. Local deletion corresponds to FSDataset messages that delete a block file from the local storage path. Delete instruction, replication instruction, and transfer recovery are rarer control actions. This taxonomy separates operationally meaningful maintenance from routine block receipt, allocation, verification, and packet responder termination.

Log parsing

The parsing task compared five parsers. Numeric abstraction replaced block identifiers and numeric values. Drain-lite used an online length-bucketed clustering strategy inspired by fixed-depth streaming parsers. Drain-lite plus numeric abstraction added pre-normalization before clustering. Token abstraction replaced block ids, IPs, ports, paths, part numbers, and numeric values. The ECLA parser used HDFS-aware evidence abstraction with the same kinds of variables, then emitted templates for downstream feature extraction. Predicted templates were mapped to gold event ids by maximum bipartite matching so that parsing accuracy measured correct line family assignment. ARI and NMI measured clustering agreement without relying only on mapping.

Window-level anomaly detection

The anomaly task used the 200 windows. A stratified 70/30 split with seed 2024 produced the training and test sets. The comparison included a WARN-count rule, a deterministic event-rule detector over raw message evidence, TF-IDF IsolationForest, TF-IDF logistic regression, TF-IDF LinearSVC, and the ECLA detector. ECLA blended logistic-regression probability with normalized evidence counts for actionable log patterns and warning counts. The threshold was selected on the training split by maximizing F1 and then applied once to the held-out test split. This separation between feature constructions, learned scoring, and held-out threshold application is consistent with security work that treats language-guided feature selection as a separately inspectable stage (Li & Lu, 2025). Metrics include accuracy, precision, recall, F1, specificity, ROC-AUC, and PR-AUC.

Root-cause ranking

The RCA task used anomalous windows with enough samples for stratified splitting. The models ranked root-cause classes rather than returning only one label. Keyword evidence scored the presence of exception, invalidSet, delete, replication, and local-deletion cues. TF-IDF nearest centroid compared test windows to class centroids. TF-IDF MultinomialNB and TF-IDF logistic regression provided probabilistic text baselines. The ECLA RCA scorer blended logistic-regression probabilities with keyword evidence and reported Top-1, Top-2, and Top-3 accuracy. This ranking design reflects incident triage, where a shortlist is often more useful than a single unqualified label. Retrieval-quality prediction has likewise been framed as a supervised evaluation problem rather than as an assumption that the first retrieved item is correct (Zhang et al., 2025).

Narrative generation and selective refusal

The narrative task compared three generation policies. The always-answer policy emitted a diagnosis for every window, even when the evidence was weak. The extractive summary policy only described observed evidence. The ECLA selective narrative policy produced a root-cause narrative only when the evidence classifier returned a confidence at or above 0.72 and the predicted root cause was non-normal. Otherwise it returned a refusal sentence stating that the window lacked sufficient failure evidence. A claim verifier flagged a hallucination when a generated diagnosis named a root cause, block id, node IP, or operational impact unsupported by the window. The verifier also counted a diagnosis for a normal window as hallucinated. This design operationalizes factuality and faithfulness concerns identified in summarization research (Maynez et al., 2020; Kryscinski et al., 2020). Evidence-chain scoring provides a related sentence-level precedent for checking attribution and provenance (Jin, 2025a). The verifier also parallels evidence-ranked claim verification, where a claim is accepted only after its supporting record has been surfaced (Su, Chen, & Qian, 2026).

Metrics, implementation, and evaluation protocol

All metrics were defined before execution. Parsing accuracy is the fraction of lines whose mapped predicted cluster matches the structured event id. Macro F1 gives equal weight to rare and frequent templates, while weighted F1 reflects the line distribution. ARI and NMI evaluate cluster agreement without requiring the predicted cluster names to match the event ids. Detection F1 is the harmonic mean of actionable-window precision and recall, specificity is true-normal recall, and PR-AUC is emphasized because actionable windows are less frequent than normal windows. RCA Top-k is correct when the true root-cause family appears in the first k ranked labels. Narrative hallucination is computed only from explicit claims in the generated text, so a refusal cannot hallucinate a cause and an extractive sentence is judged by whether it introduces unsupported content.

The implementation was deliberately compact. This choice is important because operational AI systems can fail for reasons outside prediction quality. Cold-start workload forecasting shows that limited history can dominate model choice (He et al., 2025), and capacity-dashboard research shows that forecast risk must be translated into reviewable visual evidence (Liu et al., 2025). ECLA does not evaluate capacity management, but it follows the same principle of keeping intermediate evidence visible.

The implementation was deliberately compact. It used regular expressions for HDFS variable abstraction, scikit-learn vectorizers and classifiers for the statistical baselines, and fixed random seed 2024 for train/test partitioning. No external telemetry, tickets, or manual labels were added. The only semantic assumptions are the actionable taxonomy and the priority order used when a 10-line window contains more than one actionable class. Those assumptions are visible in the code and in Table 2, which keeps the experimental pipeline auditable from raw line to final narrative.

The train/test procedure was designed to keep tasks linked but not identical. Parsing used all lines because its target is a structural mapping from raw message to event family. Anomaly detection used a stratified window split because the target is binary. RCA used only anomalous windows with enough examples for stratification because normal windows have no root-cause class. Narrative evaluation used all 200 windows so that refusal on normal windows was counted directly. This separation is important: a method can detect anomalies well and still generate unsupported text if it is not evaluated at the narrative stage. The architecture is summarized in Figure 1.

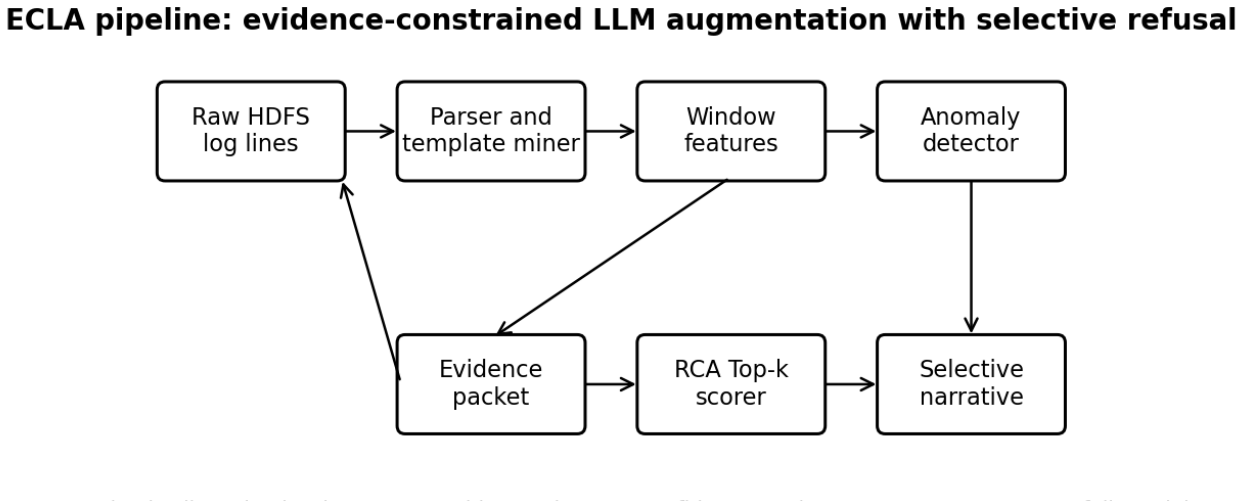

Figure 1. ECLA architecture for evidence-constrained log intelligence.

Table 1. HDFS-2k corpus statistics.

Statistic	Value
Log lines	2000
Structured templates	14
Components	6
INFO lines	1920
WARN lines	80
Unique block ids	1994
Window size	10
Windows	200
Anomalous/actionable windows	89

Table 2. Event catalog, frequencies, and actionable labels.

EventId	Count	Level	Component	EventTemplate	Actionable Label
E1	80	INFO	dfs.DataNode$DataXceiver	<*>:<*> Served block blk_<*> to /<*>	Normal
E10	311	INFO	dfs.DataNode$PacketResponder	PacketResponder <*> for block blk_<*> terminating	Normal
E11	292	INFO	dfs.DataNode$PacketResponder	Received block blk_<*> of size <*> from /<*>	Normal
E12	2	INFO	dfs.DataNode$DataXceiver	Received block blk_<*> src: /<*>:<*> dest: /<*>:<*> of size <*>	Normal
E13	292	INFO	dfs.DataNode$DataXceiver	Receiving block blk_<*> src: /<*>:<*> dest: /<*>:<*>	Normal
E14	20	INFO	dfs.DataBlockScanner	Verification succeeded for blk_<*>	Normal
E2	1	INFO	dfs.DataNode	<*>:<*> Starting thread to transfer block blk_<*> to <*>:<*>	Transfer recovery
E3	80	WARN	dfs.DataNode$DataXceiver	<*>:<*>:Got exception while serving blk_<*> to /<*>:	Serving exception
E4	5	INFO	dfs.FSNamesystem	BLOCK* ask <*>:<*> to delete blk_<*>	Delete instruction
E5	1	INFO	dfs.FSNamesystem	BLOCK* ask <*>:<*> to replicate blk_<*> to datanode(s) <*>:<*>	Replication instruction
E6	314	INFO	dfs.FSNamesystem	BLOCK* NameSystem.addStoredBlock: blockMap updated: <*>:<*> is added to blk_<*> size <*>	Normal
E7	115	INFO	dfs.FSNamesystem	BLOCK* NameSystem.allocateBlock: /<*>/part-<*>. blk_<*>	Normal
E8	224	INFO	dfs.FSNamesystem	BLOCK* NameSystem.delete: blk_<*> is added to invalidSet of <*>:<*>	Namespace invalidation
E9	263	INFO	dfs.FSDataset	Deleting block blk_<*> file /<*>/blk_<*>	Local deletion

Table 3. Window-level root-cause label distribution.

RootCause	Windows
Normal	111
Local deletion	42
Namespace invalidation	27
Serving exception	18
Transfer recovery	1
Delete instruction	1

Results and Discussion

Parsing performance

The first result is that event-family parsing is easy on this corpus, but template compactness varies. Table 4 shows perfect mapped parsing accuracy for all parsers. This does not mean that all parsers recovered the same structure. Numeric abstraction produced 46 templates, substantially more than the 14 gold event families, and its ARI and NMI were lower than the other structured parsers. Drain-lite without pre-abstraction produced 31 templates. Drain-lite plus numeric abstraction, token abstraction, and the ECLA parser produced 16 templates with near-perfect ARI and NMI. The remaining difference from 14 templates comes from rare HDFS address and path variants that preserve separate surface forms while mapping to the correct event family. Figure 4 visualizes the parsing comparison, and Figure 3 shows the underlying event-frequency skew that makes over-splitting easy to miss when only accuracy is reported.

Runtime is also consistent with the intended lightweight setting. The fastest parser processed 1,000 lines in 4.13 ms, and the ECLA parser processed 1,000 lines in 10.71 ms on the local run. This difference is negligible at the size of the corpus and remains useful because the ECLA parser produces stronger downstream evidence. The result supports a practical choice: for a compact HDFS benchmark, it is better to spend a few additional milliseconds on stable abstraction than to carry a larger set of redundant templates into anomaly detection and RCA.

Temporal and frequency analysis explains the downstream detection behavior. Figure 3 shows that the most common templates are block-map updates, packet responder termination, block receipt, block receiving, and local deletion. Figure 2 shows that WARN events occur in sparse bursts rather than throughout the whole time range. WARN count alone is therefore highly precise but low recall for the broader actionable taxonomy, because many actionable windows are INFO-level invalidation or deletion events. This justifies evaluating semantic evidence beyond warning level.

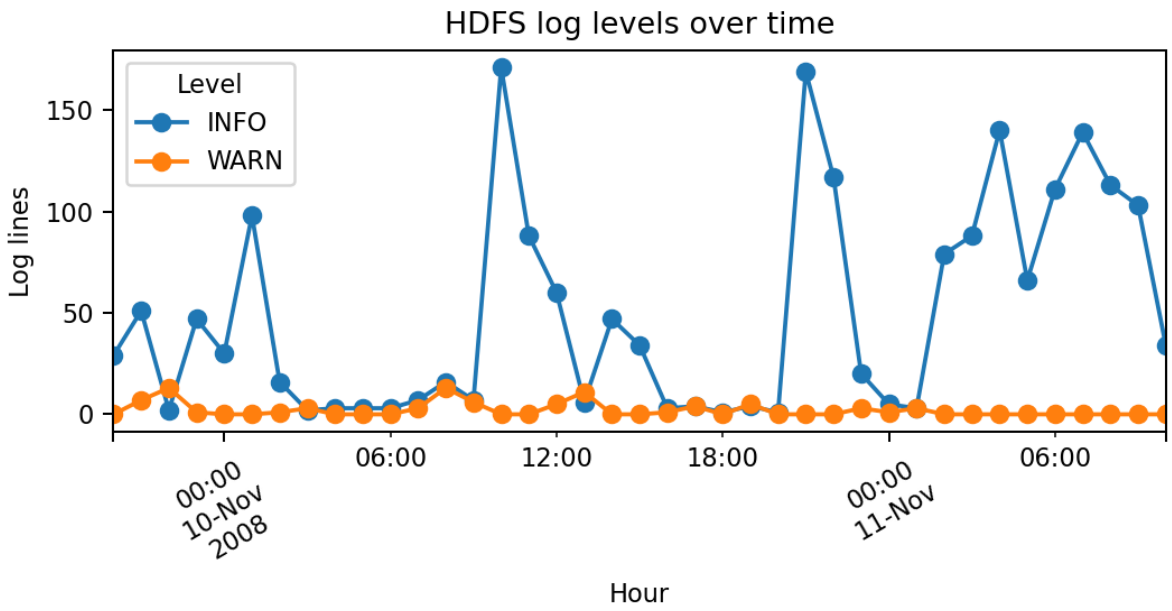

Figure 2. Temporal distribution of HDFS log levels.

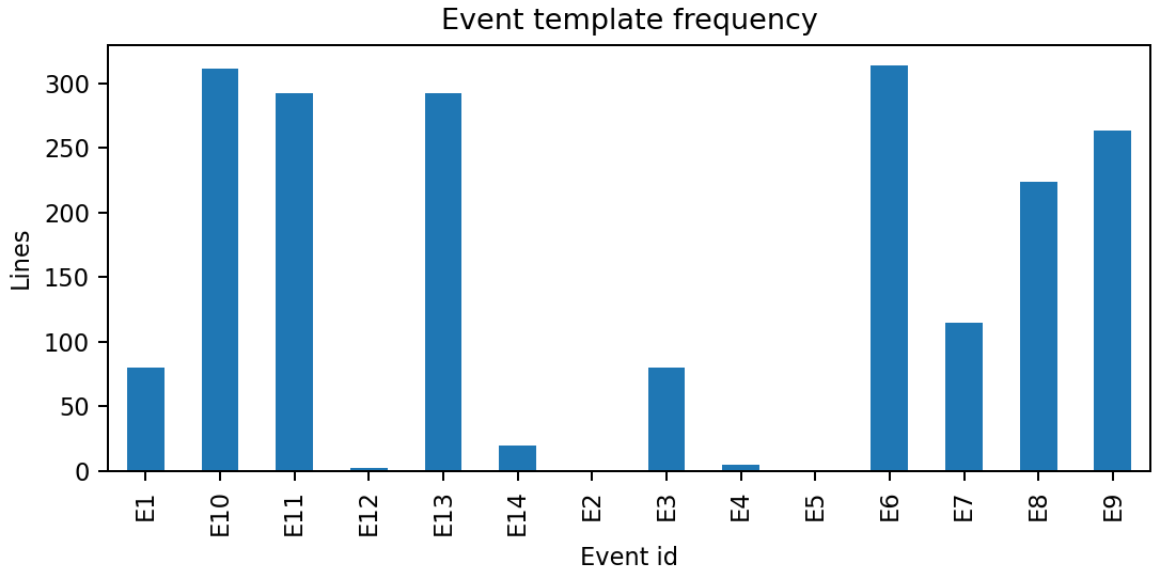

Figure 3. Frequency of event templates.

Table 4. Parsing comparison on HDFS-2k.

Parser	ParsingAccuracy	MacroF1	WeightedF1	ARI	NMI	TemplatesPredicted	RuntimeMsPer1k
Numeric abstraction	1.000	1.000	1.000	0.914	0.906	46	4.133
Drain-lite	1.000	1.000	1.000	0.997	0.990	31	6.453
Drain-lite + numeric	1.000	1.000	1.000	1.000	0.999	16	11.72
Token abstraction	1.000	1.000	1.000	1.000	0.999	16	7.955
ECLA parser	1.000	1.000	1.000	1.000	0.999	16	10.71

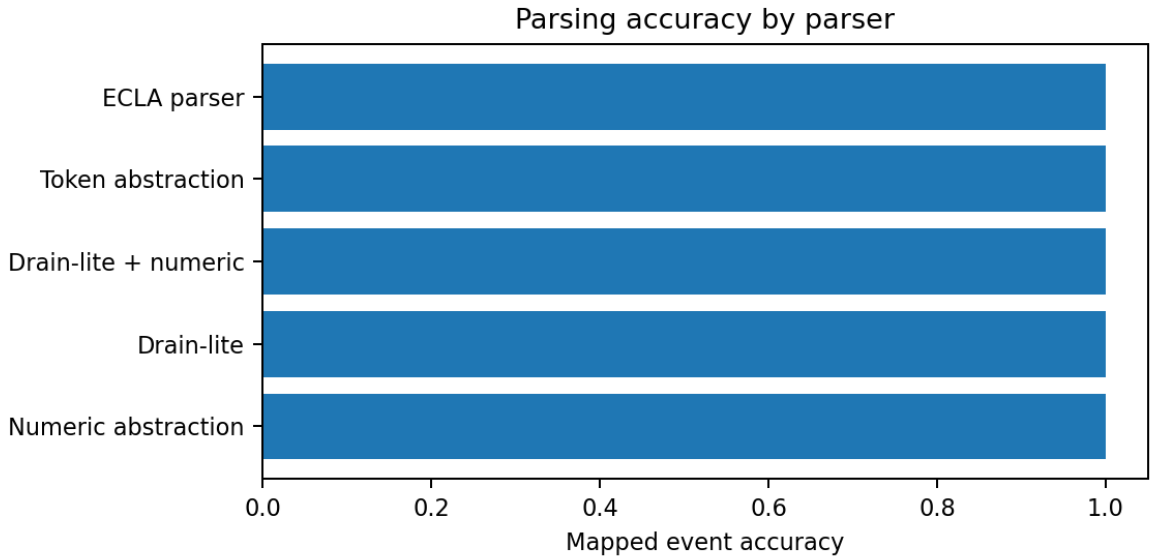

Figure 4. Parser mapped event accuracy.

Window-level anomaly detection

Table 5 reports the anomaly detection results. The WARN-count rule reached precision 1.000 but recall 0.185 and F1 0.312, confirming that warning-level filtering misses most actionable windows. TF-IDF IsolationForest performed poorly under this small-window setting, with F1 0.067. The supervised TF-IDF models performed much better: logistic regression obtained F1 0.898 and PR-AUC 0.979,

and LinearSVC obtained F1 0.920 and PR-AUC 0.982. ECLA achieved F1 0.926 and PR-AUC 0.994, the best learned-model result. The deterministic event-rule detector reached F1 1.000 because the actionable taxonomy is explicitly recoverable from message text; it functions as a consistency check that the event semantics are present in the raw logs rather than hidden in external labels.

The ECLA confusion matrix in Table 6 shows 31 true normal windows, 25 true actionable windows correctly detected, two false positives, and two false negatives on the held-out split. This profile is appropriate for a selective narrative layer because unsupported normal windows are more likely to be refused later, while missed actionable windows expose the detection coverage limit. Figure 5 compares F1 and PR-AUC across the anomaly detectors. The main pattern is that statistical text models benefit from symbolic evidence counts, while purely unsupervised density modeling is unstable on the small sample.

The two ECLA false positives came from windows whose lexical context resembled actionable block maintenance even though the dominant label was normal. The need to preserve lightweight scoring is also operational: GPU-memory prediction work illustrates how resource forecasts become part of model-selection and deployment constraints (Wang et al., 2025). In a monitoring pipeline this is a tolerable error type because the narrative stage still checks evidence before producing a root-cause statement. The two false negatives are more important for triage because an actionable window missed at detection time cannot be ranked or summarized later. This explains why the method optimizes F1 and PR-AUC rather than accuracy alone.

Table 5. Window-level anomaly detection results.

Model	Accuracy	Precision	Recall	F1	Specificity	ROC_AUC	PR_AUC
WARN-count rule	0.633	1.000	0.185	0.312	1.000	0.593	0.552
Event-rule detector	1.000	1.000	1.000	1.000	1.000	1.000	1.000
TF-IDF IsolationForest	0.533	0.333	0.037	0.067	0.939	0.614	0.522
TF-IDF LogisticRegression	0.917	1.000	0.815	0.898	1.000	0.978	0.979
TF-IDF LinearSVC	0.933	1.000	0.852	0.920	1.000	0.983	0.982
ECLA detector	0.933	0.926	0.926	0.926	0.939	0.994	0.994

Table 6. ECLA detector confusion matrix on the held-out split.

Gold/Pred	Pred normal	Pred actionable
True normal	31	2
True actionable	2	25

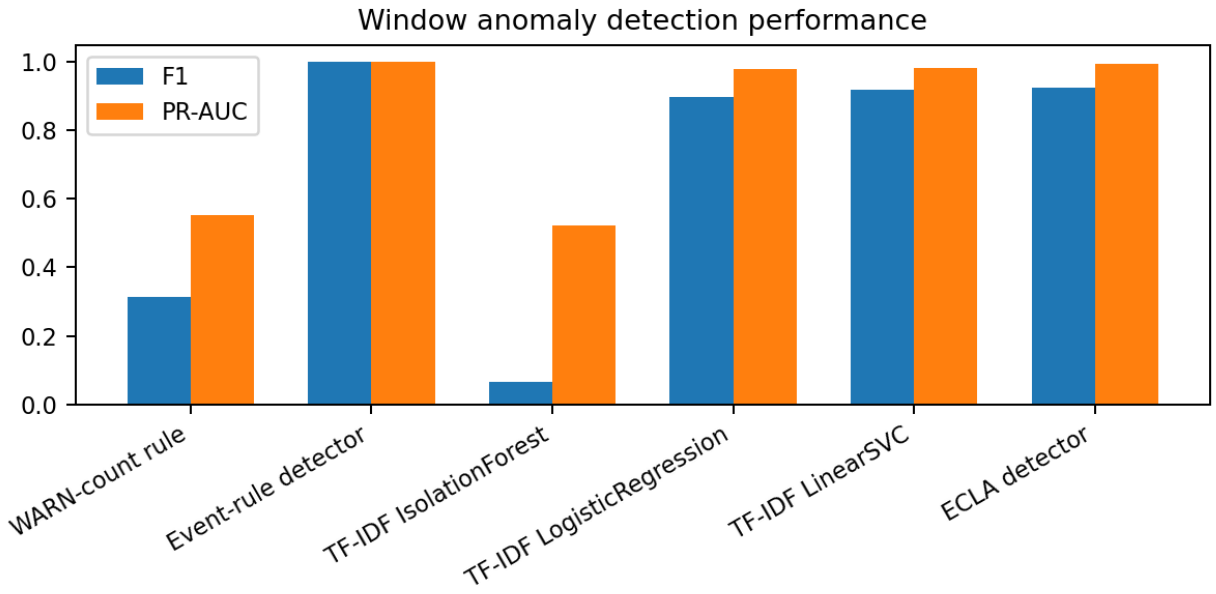

Figure 5. Anomaly detection F1 and PR-AUC.

Root-cause ranking and narrative generation

RCA results are shown in Table 7. Keyword evidence, TF-IDF nearest centroid, and TF-IDF MultinomialNB each achieved Top-1 = 0.903. Logistic regression improved Top-1 to 0.935. ECLA matched this Top-1 result and maintained Top-2 = 1.000 and Top-3 = 1.000. The ranking result is important because HDFS windows often contain multiple block operations, and a second-ranked cause can still guide the operator to the right event family. Figure 6 shows the Top-k comparison. The ECLA confidence is lower than MultinomialNB confidence because the blended scorer deliberately moderates probabilities with symbolic evidence rather than returning overconfident rankings.

The remaining RCA Top-1 errors occur in mixed maintenance contexts where deletion and invalidation evidence appear close together. These cases are not arbitrary failures: the correct label still appears by Top-2 for ECLA. In practice, this means the system can present a ranked shortlist without inventing a single unsupported answer. The Top-2 and Top-3 results are therefore not secondary decorations. They measure whether the assistant can keep the true failure family in the operator's immediate field of attention when the first score is uncertain.

The generation results in Table 8 show the value of selective refusal. This finding is consistent with conformal alert-control work in which anomaly scores are translated into controlled, auditable incident tickets rather than unconditional notifications (Xin, 2026a). The always-answer narrative answered all 200 windows but had a hallucination rate of 0.555 because it diagnosed normal windows and added unsupported impact claims. The extractive summary had zero hallucination because it only repeated evidence, but it did not make a selective root-cause decision. ECLA answered 89 windows, refused 111 windows, achieved answer accuracy 1.000, and had hallucination rate 0.000 on both answered and all-window denominators. This is the expected behavior: when evidence is absent or normal, refusal is safer than a fluent diagnosis. Figure 7 compares hallucination and refusal rates.

The extractive summary is a strong faithfulness baseline, but its usefulness differs from ECLA. It can tell an operator that a line contains an invalidSet message or a deletion message, yet it does not decide whether the window deserves a root-cause narrative. ECLA makes that decision explicitly. When the window is normal, it refuses. When the evidence is actionable, it names the supported class and includes only evidence present in the window. This distinction is why both hallucination rate and refusal rate are needed; faithfulness without diagnosis and diagnosis without refusal represent different operational tradeoffs. It also agrees with evidence-card design, where ranking transparency and visible source links are treated as separate from the prose of the explanation (Jin, 2025b).

Table 7. RCA Top-k results on anomalous windows.

RCA_Model	Top1	Top2	Top3	MeanTop1Confidence
Keyword evidence	0.903	1.000	1.000	0.792
TF-IDF nearest centroid	0.903	0.968	1.000	0.834
TF-IDF MultinomialNB	0.903	1.000	1.000	0.920
TF-IDF LogisticRegression	0.935	1.000	1.000	0.721
ECLA RCA scorer	0.935	1.000	1.000	0.736

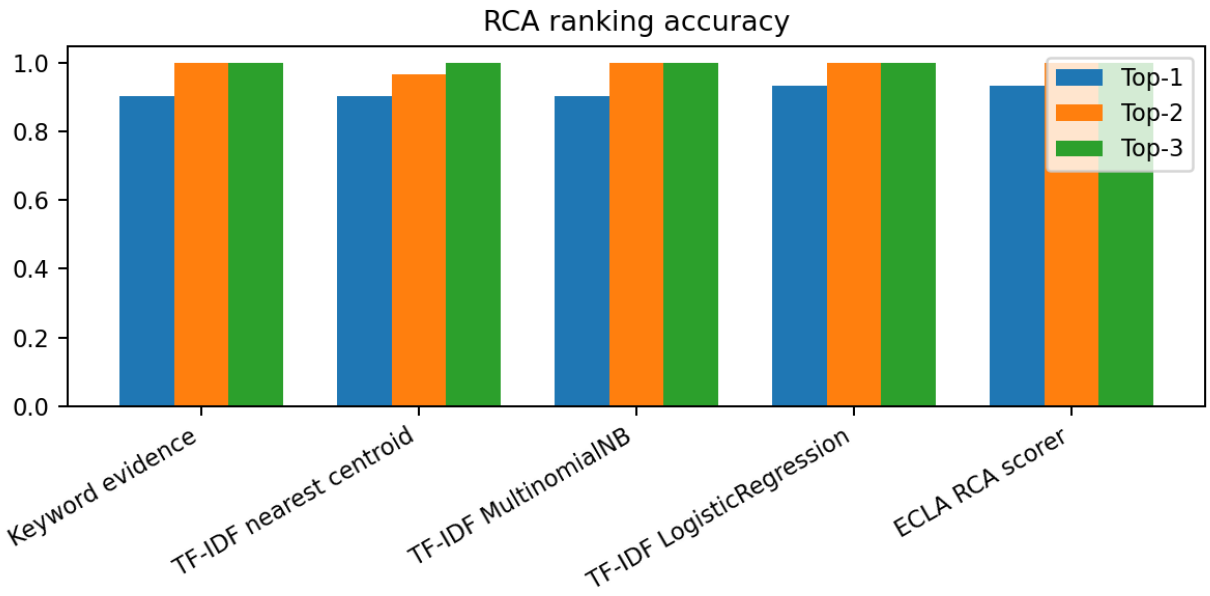

Figure 6. Root-cause ranking Top-k accuracy.

Table 8. Narrative generation and refusal results.

Generator	Answered	Refused	Refusal Rate	AnswerAccuracy	HallucinationRateAnswered	Hallucination RateAll	MeanRougeL
Always-answer narrative	200	0	0.000	0.445	0.555	0.555	0.058
Extractive summary	200	0	0.000	1.000	0.000	0.000	0.455
ECLA selective narrative	89	111	0.555	1.000	0.000	0.000	0.162

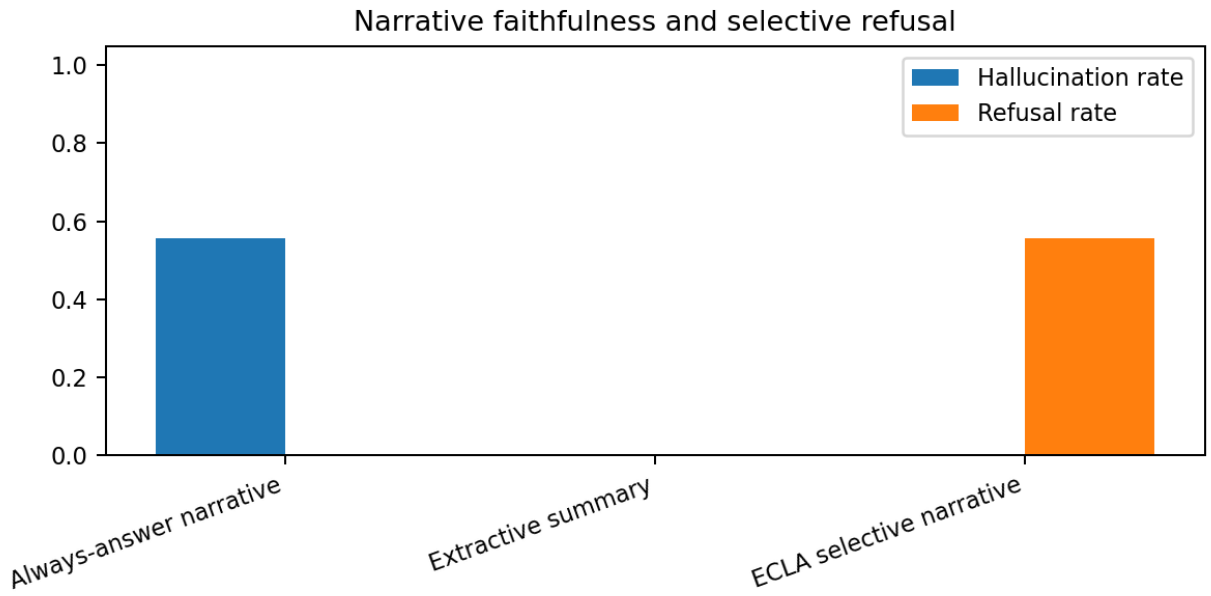

Figure 7. Narrative hallucination and refusal rates.

Ablation and coverage-risk behavior

Ablation results in Table 9 isolate the role of each component. Removing symbolic evidence reduced detection PR-AUC from 0.994 to 0.979 and F1 from 0.926 to 0.898. Rules alone reached perfect detection because the actionable definition is directly encoded in visible log phrases, but rules do not provide a ranked probabilistic RCA score or a calibrated narrative confidence. Removing selective refusal kept detection unchanged but increased narrative hallucination to 0.555. This result is the clearest evidence that refusal is not an auxiliary formatting choice; it changes the factual risk of the final user-facing narrative.

Table 10 and Figure 8 report the coverage-risk curve for ECLA refusal thresholds. At thresholds from 0.55 to 0.70, ECLA answered 44.5% of windows with zero hallucination risk. Raising the threshold reduced coverage gradually, down to 35.5% at 0.95, while maintaining zero risk under the verifier. The zero-risk curve occurs because the verifier is strict and the evidence-constrained narrative names only the dominant supported root cause. It should therefore be read as internal rule consistency, not as a general guarantee of zero hallucination. Privacy and data-integrity risk-card research makes a similar distinction between a deterministic oversight proxy and measured human behavior (Su, Rao, & Ma, 2026). In an operations setting, such a curve allows the team to choose how often the assistant should speak. Lower thresholds give more actionable summaries, while higher thresholds reserve narratives for highly confident windows.

Cross-domain implications

The overall pattern is consistent across tasks. Parsing transforms unstructured HDFS text into stable event evidence. Detection identifies whether a window contains actionable evidence. RCA ranks the most likely failure family. The narrative layer then converts this evidence into a concise explanation only when the claim is supported. The end-to-end design aligns with established log-analysis pipelines (He et al., 2021), classical anomaly-detection principles (Chandola et al., 2009), and modern retrieval-grounding ideas (Lewis et al., 2020), while adding an explicit refusal decision to control

hallucination.

The parser results also show why a single metric is insufficient. Numeric abstraction has perfect mapped accuracy because every over-split cluster still belongs to the right event family, but it predicts 46 templates and therefore loses compactness. For an alerting system, over-splitting can increase feature dimensionality and reduce robustness when a new block path or address appears. ECLA and the stronger abstraction baselines keep the template space near the 14-family target while preserving exact event mapping. This matters for RCA because the evidence packet should contain stable event names rather than a long tail of superficial variants. A parallel can be seen in SSD health-state classification, where sequence modeling and attention are useful only after the health indicators have been organized into a stable predictive representation (Wen et al., 2025).

The anomaly results show the complementarity of symbolic and statistical signals. The event-rule detector is exact for this taxonomy, but it is a narrow detector: it recognizes known actionable phrases and does not learn soft similarity. TF-IDF logistic regression and LinearSVC learn more general lexical patterns but miss some actionable windows. ECLA combines the two, which yields the best learned-model PR-AUC and a balanced F1 without relying entirely on either phrase matching or statistical coefficients. This behavior is desirable in small log benchmarks because rare windows can be too sparse for purely statistical learning, while rule-only systems can be brittle when message wording changes. Related AIOps routing results likewise caution that closed-label and retrieval policies can have limited unseen-template generalization (Li et al., 2026).

The narrative results should be read together with detection and RCA. ECLA does not attempt to summarize every line. It makes a narrower promise: when the evidence identifies a root-cause family, it produces a concise supported narrative; otherwise it refuses. The always-answer baseline demonstrates the failure mode that selective refusal avoids. It produced many fluent statements, but the verifier marked unsupported diagnoses for normal windows. The extractive baseline shows the opposite extreme: it is faithful but less diagnostic. ECLA occupies the middle position by producing root-cause text only for the 89 windows where the evidence supports a cause.

Table 9. ECLA ablation results.

Configuration	DetectionF1	DetectionPR_AUC	NarrativeHallucination	RefusalRate
Full ECLA	0.926	0.994	0.000	0.555
No symbolic evidence	0.898	0.979	0.000	0.555
Rules only	1.000	1.000	0.000	0.555
No selective refusal	0.926	0.994	0.555	0.000

Table 10. Selective refusal coverage-risk curve.

Threshold	Coverage	Risk	RefusalRate
0.550	0.445	0.000	0.555
0.600	0.445	0.000	0.555
0.650	0.445	0.000	0.555
0.700	0.445	0.000	0.555
0.750	0.430	0.000	0.570
0.800	0.425	0.000	0.575
0.850	0.405	0.000	0.595
0.900	0.375	0.000	0.625
0.950	0.355	0.000	0.645

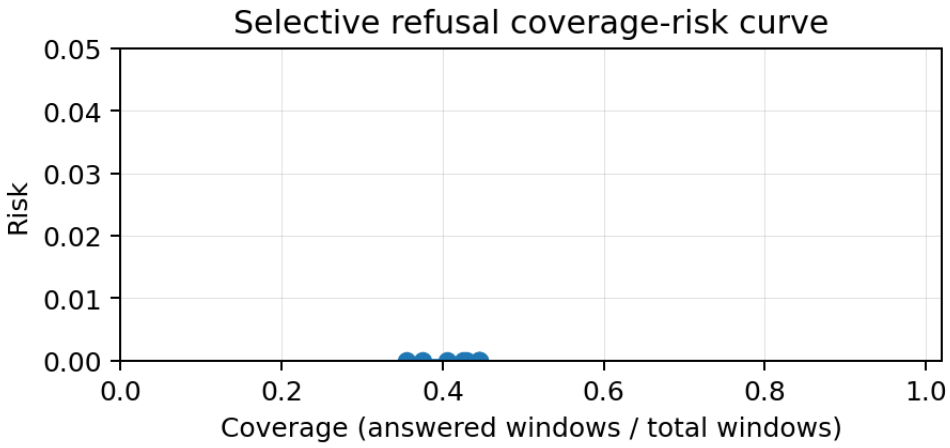

Figure 8. Coverage-risk curve for selective refusal.

Limitations

The evaluation is intentionally compact. HDFS-2k has 2,000 lines and 200 windows, which makes it well suited for transparent LogEval-style experimentation but smaller than full production HDFS logs. The actionable labels are derived from event semantics rather than from an independent incident ticket system. This makes the labels

reproducible and internally consistent, but it narrows the meaning of anomaly to failure-relevant and maintenance-relevant evidence visible in the log text.

The generation layer is evidence-constrained and deterministic. This choice makes hallucination measurement reproducible, but it does not measure the full linguistic range of a large proprietary or open-weight LLM. The result therefore demonstrates how LLM-facing evidence control and refusal should be evaluated, rather than claiming that a particular foundation model has been optimized. The same protocol can wrap a larger LLM by passing the evidence packet and enforcing the same verifier, but that extension would require separate model, prompt, decoding, latency, and human-review evaluation.

Because the verbalizer is constrained, the generated narratives are concise and sometimes less varied than a free-form LLM response. This is a deliberate tradeoff. The paper evaluates factual support, not stylistic richness. In an operations environment, a short accurate sentence that names the supported failure family is preferable to a longer explanation that speculates about cluster-wide impact. The same evidence packet could support richer wording after deployment, but the verifier should remain in place.

The window size is fixed at 10 lines. This setting is appropriate for the short corpus and provides enough windows for stratified evaluation, but other HDFS deployments may require session-level grouping by block id or longer time windows. The RCA taxonomy is also specific to the observed HDFS templates. Systems with application-level exceptions, resource saturation, or network partitions would require additional root-cause classes and corresponding evidence rules.

The claim verifier is strict but still rule-based. Trust-calibrated multilingual RAG research similarly shows that retrieval support and response confidence must be evaluated as separate quantities (Chen & Xu, 2026). It checks whether root-cause names, block identifiers, node identifiers, and impact claims are supported by the window. It does not evaluate more subtle linguistic issues such as causal hedging, operator usefulness, or whether a generated explanation is the most concise possible one. Human incident review would add qualitative value, but the present verifier gives a repeatable lower-level factuality test that can be applied automatically to every generated narrative. Agent-trajectory research similarly separates tool-use success prediction from proof that an explanation or action is acceptable to a human operator (Zhong et al., 2026).

The experiment also uses a single random seed for the reported split. The fixed seed keeps the tables reproducible and makes the prediction files easy to inspect. A larger study could report many random splits or cross-system transfer, but that would change the paper from a compact LogEval-style HDFS evaluation into a broader benchmarking exercise. The conclusions here are strongest for the tested corpus and task construction: evidence control reduces unsupported narratives, and parsing plus semantic evidence is enough to recover the main actionable windows.

Summary and implications

This study presented ECLA, an evidence-constrained, LLM-ready pipeline for LogEval 2024 style HDFS log intelligence. The same HDFS-2k corpus was used for parsing, anomaly detection, RCA ranking, and selective failure narrative generation. The experiments produced 10 tables and eight figures. ECLA achieved perfect mapped parser accuracy, learned-model anomaly F1 of 0.926, RCA Top-1 of 0.935, RCA Top-2 of 1.000, and zero hallucinated selective narratives under the study's rule-based claim verifier.

The main finding is that selective refusal materially changes the reliability of narrative generation. An always-answer policy produced fluent but unsupported diagnoses on normal windows. The ECLA narrative layer refused those windows and generated a root-cause narrative only when the parser, detector, and evidence scorer supported the claim. Within this benchmark and verifier, the result supports a practical design rule for log assistants: the model should first prove that its evidence is adequate, and only then translate the

evidence into a human-readable failure narrative.

A second finding is methodological. A LogEval-style paper should evaluate the full chain instead of reporting an isolated detection score. In this study, parsing quality affected template compactness, anomaly detection determined coverage, RCA ranking determined which cause could be named, and selective refusal determined whether the final narrative was faithful. Reporting these stages together made the tradeoffs visible: an exact symbolic detector can recover the taxonomy, learned detectors generalize lexical evidence, and refusal turns evidence sufficiency into a quantifiable metric. The conclusion remains limited to the tested HDFS-2k construction and does not establish production effectiveness or human trust.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys, 41*(3), Article 15. https://doi.org/10.1145/1541880.1541882

Chen, Y., & Xu, H. (2026). Trust-calibrated multilingual RAG for humanitarian information platforms: Empirical evaluation on OMoS-QA for migration information access. *International Journal of Graphic Design, 4*(1), 141–164. https://doi.org/10.51903/ijgd.v4i1.3552

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

Du, M., & Li, F. (2016). Spell: Streaming parsing of system event logs. In *2016 IEEE 16th International Conference on Data Mining* (pp. 859–864). IEEE. https://doi.org/10.1109/ICDM.2016.0103

Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). DeepLog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1285–1298). Association for Computing Machinery. https://doi.org/10.1145/3133956.3134015

Fu, Q., Lou, J.-G., Wang, Y., & Li, J. (2009). Execution anomaly detection in distributed systems through unstructured log analysis. In *2009 Ninth IEEE International Conference on Data Mining* (pp. 149–158). IEEE. https://doi.org/10.1109/ICDM.2009.24

Guo, H., Yuan, S., & Wu, X. (2021). LogBERT: Log anomaly detection via BERT. In *2021 International Joint Conference on Neural Networks* (pp. 1–8). IEEE. https://doi.org/10.1109/IJCNN52387.2021.9534113

He, P., Zhu, J., Zheng, Z., & Lyu, M. R. (2017). Drain: An online log parsing approach with fixed depth tree. In *2017 IEEE International Conference on Web Services* (pp. 33–40). IEEE. https://doi.org/10.1109/ICWS.2017.13

He, S., He, P., Chen, Z., Yang, T., Su, Y., & Lyu, M. R. (2021). A survey on automated log analysis for reliability engineering. *ACM Computing Surveys, 54*(6), Article 130. https://doi.org/10.1145/3460345

He, S., Li, C., & Rao, H. (2025). Few-shot cold-start workload forecasting for new AI inference tenants with time-series foundation models. *Journal of Technology Informatics and Engineering, 4*(1), 306–324. https://doi.org/10.51903/jtie.v4i1.546

He, S., Zhu, J., He, P., & Lyu, M. R. (2016). Experience report: System log analysis for anomaly detection. In *2016 IEEE 27th International Symposium on Software Reliability Engineering* (pp. 207–218). IEEE. https://doi.org/10.1109/ISSRE.2016.21

Jin, J. (2025a). Evidence-chain reliable RAG: Hallucination detection, source attribution, and deterministic provenance explanations. *Journal of Technology Informatics and Engineering, 4*(2), 520–533. https://doi.org/10.51903/jtie.v4i2.535

Jin, J. (2025b). LLM-style evidence cards for scientific search interfaces: A UI/UX design framework for retrieval transparency, ranking trust, and visual evidence hierarchy. *International Journal of Graphic Design, 3*(2), 397–414. https://doi.org/10.51903/ijgd.v3i2.3698

Kryscinski, W., McCann, B., Xiong, C., & Socher, R. (2020). Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (pp. 9332–9346). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.750

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems, 33*, 9459–9474.

Li, C., Liu, G., & Zhao, Z. (2026). Cost-aware LLM-style routing for AIOps log analysis: Log parsing, anomaly detection, fault diagnosis, and incident summarization on LogEval task files. *Journal of Technology Informatics and Engineering, 5*(2), 91–103. https://doi.org/10.51903/jtie.v5i2.538

Li, Y., & Lu, S. (2025). Language-guided feature selection for DDoS and intrusion detection on CICIDS2017. *Journal of Technology Informatics and Engineering, 4*(1), 284–305. https://doi.org/10.51903/jtie.v4i1.531

Li, Y., Lu, S., & Zhao, L. (2025). LLM-as-design-critic: Aligning AI-generated UI feedback with human graphic design judgment. *International Journal of Graphic Design, 3*(1), 196–215. https://doi.org/10.51903/ijgd.v3i1.3661

Liu, G., He, S., & Wong, H. (2025). LLM-compatible visual brief cards for AI infrastructure capacity dashboards: A UI/UX framework for turning forecast risk into graphic design decisions. *International Journal of Graphic Design, 3*(1), 196–213. https://doi.org/10.51903/ijgd.v3i1.3723

Makanju, A. A. O., Zincir-Heywood, A. N., & Milios, E. E. (2009). Clustering event logs using iterative partitioning. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1255–1264). Association for Computing Machinery. https://doi.org/10.1145/1557019.1557154

Maynez, J., Narayan, S., Bohnet, B., & McDonald, R. (2020). On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 1906–1919). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.173

Meng, W., Liu, Y., Zhu, Y., Zhang, S., Pei, D., Liu, Y., Chen, Y., Zhang, R., Tao, S., Sun, P., & Zhou, R. (2019). LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (pp. 4739–4745). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/658

Oliner, A., & Stearley, J. (2007). What supercomputers say: A study of five system logs. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks* (pp. 575–584). IEEE. https://doi.org/10.1109/DSN.2007.103

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1135–1144). Association for Computing Machinery. https://doi.org/10.1145/2939672.2939778

Su, W., Chen, S., & Qian, E. (2026). Narrative-aware scientific claim verification agent with evidence ranking for ClimateCheck. *Journal of Technology Informatics and Engineering, 5*(1), 327–340. https://doi.org/10.51903/jtie.v5i1.549

Su, W., Chen, S., & Zhao, C. (2025). Budgeted multi-hop retrieval agent for compositional question answering: A retrieval-policy evaluation on the official MultiHop-RAG benchmark. *Journal of Technology Informatics and Engineering, 4*(3), 649–662. https://doi.org/10.51903/jtie.v4i3.543

Su, W., Rao, H., & Ma, E. (2026). Privacy and data-integrity risk cards for LLM agents: A UI/UX design framework for secure human oversight under prompt-injection attacks. *International Journal of Graphic Design, 4*(1), 186–191. https://doi.org/10.51903/ijgd.v4i1.3699

Vaarandi, R. (2003). A data clustering algorithm for mining patterns from event logs. In *Proceedings of the 3rd IEEE Workshop on IP Operations & Management* (pp. 119–126). IEEE. https://doi.org/10.1109/IPOM.2003.1251233

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*, 5998–6008.

Wang, C., Wen, Z., Zhang, R., Xu, P., & Jiang, Y. (2025). GPU memory requirement prediction for deep learning task based on bidirectional gated recurrent unit optimization transformer. In *2025 5th International Conference on Artificial Intelligence, Virtual Reality and Visualization*. IEEE. https://doi.org/10.1109/AIVRV67401.2025.11350369

Wen, Z., Zhang, R., & Wang, C. (2025). Optimization of bi-directional gated loop cell based on multi-head attention mechanism for SSD health state classification model. In *2025 6th International Conference on Electronic Communication and Artificial Intelligence* (pp. 548–552). IEEE. https://doi.org/10.1109/ICECAI66283.2025.11171441

Xin, Q. (2025). Explaining OpenStack failure-injection log anomalies with retrieved normal prototypes. *Emerging Information Science and Technology, 6*(2), 125–146. https://doi.org/10.18196/eist.v6i2.31232

Xin, Q. (2026a). Log anomaly detection with conformal alert control and evidence-grounded incident ticket generation. *Aviation Electronics, Information Technology, Telecommunications, Electricals, and Controls (AVITEC), 8*(2), 247–264. https://doi.org/10.28989/avitec.v8i2.3974

Xin, Q. (2026b). Self-supervised log anomaly detection with LogBERT-style transformers: Full empirical evaluation on a reproducible SynHDFS benchmark. *JEECS (Journal of Electrical Engineering and Computer Sciences), 11*(1), 23–35. https://doi.org/10.54732/jeecs.v11i1.3

Xu, H., Chen, Y., & Med, A. (2025). Automatic detection and explanation of dark patterns from interface microcopy: Empirical comparison of BERT-style encoders, RoBERTa-style encoders, and LLM-style decoders on the ec-darkpattern dataset. *Journal of Technology Informatics and Engineering, 4*(3), 590–612. https://doi.org/10.51903/jtie.v4i3.491

Xu, W., Huang, L., Fox, A., Patterson, D., & Jordan,

M. I. (2009). Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles* (pp. 117–132). Association for Computing Machinery. https://doi.org/10.1145/1629575.1629587

Zhang, R., Wen, Z., Wang, C., Tang, C., Xu, P., & Jiang, Y. (2025). Quality analysis and evaluation prediction of RAG retrieval based on machine learning algorithms. *arXiv*. https://doi.org/10.48550/arXiv.2511.19481

Zhang, X., Xu, Y., Lin, Q., Qiao, B., Zhang, H., Dang, Y., Xie, C., Yang, X., Cheng, Q., Li, Z., Chen, J., He, X., Yao, R., Lou, J.-G., Chintalapati, M., Shen, F., & Zhang, D. (2019). Robust log-based anomaly detection on unstable log data. In *Proceedings of the 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 807–817). Association for Computing Machinery. https://doi.org/10.1145/3338906.3338931

Zhong, Z. S., Li, C., & Rao, H. (2026). Trajectory reliability prediction for generalist AI agents: Tool-use failure analysis and success forecasting on ZClawBench. *Journal of Technology Informatics and Engineering*, *5*(1), 341–360. https://doi.org/10.51903/jtie.v5i1.539

Zhong, Z. S., Pan, X., & Lei, Q. (2025). Bridging domains with approximately shared features. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics* (Proceedings of Machine Learning Research, Vol. 258, pp. 559–567). PMLR.

Zhu, J., He, S., He, P., Liu, J., & Lyu, M. R. (2023). Loghub: A large collection of system log datasets for AI-driven log analytics. In *2023 IEEE 34th International Symposium on Software Reliability Engineering* (pp. 355–366). IEEE. https://doi.org/10.1109/ISSRE59848.2023.00071

Zhu, J., He, S., Liu, J., He, P., Xie, Q., Zheng, Z., & Lyu, M. R. (2019). Tools and benchmarks for automated log parsing. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice* (pp. 121–130). IEEE. https://doi.org/10.1109/ICSE-SEIP.2019.00021