Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Why Benign-Only Log Anomaly Detection Fails Under Drift: Two Mechanisms, Seven Designs, and a Label-Efficiency Bound

Sarita Nandal

sarita@ma.iitr.ac.in

Indian Institute of Technology Roorkee

Research Article

Keywords:

Posted Date: November 17th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11401261/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Why Benign-Only Log Anomaly Detection Fails Under Drift: Two Mechanisms, Seven Designs, and a Label-Efficiency Bound

Sarita Nandal[1]

[1]Indian Institute of Technology Roorkee, Roorkee, India, sarita@ma.iitr.ac.in

Abstract

Deep sequence models dominate log-based anomaly detection benchmarks, yet controlled re-evaluations have shown that their reported accuracy collapses under chronological evaluation—without explaining why. We dissect that collapse. Under a leak-free, benign-only protocol on HDFS, BGL, and Thunderbird—chronological and random splits side by side, ten-seed confidence intervals, DeepLog and LogBERT re-run on byte-identical splits—lightweight CPU detectors match or exceed the deep re-runs wherever the template vocabulary is stable (F1 0.92–0.97 at 0.2–0.5% false-positive rates), and on two of the three benchmarks a one-line unseen-template rule suffices (F1 0.90–0.998), a quantified form of the dataset-triviality critique; HDFS, whose anomalies are structural, is the one system that needs a learned detector. The chronological collapse has two distinct mechanisms. On BGL the template vocabulary is replaced mid-trace: identity evidence is destroyed, the only transferable signal is discriminative fault semantics, and benign-only learning cannot exploit it—six designs without external semantic priors fail concordantly, a frozen pretrained-embedding prior lifts ranking above chance without becoming deployable, and full period-causal supervision yields a ranking (PR-AUC 0.722) whose operating point does not transfer (F1 0.44 causal versus 0.76 oracle). On Thunderbird the signal survives but the calibration era does not: a labeled counterfactual bounds what era-matched recalibration could recover (PCA F1 0.61). Both failures are silent at the alarm level; we identify two label-free self-monitoring statistics that expose them. All code, splits, per-seed results, and re-run artifacts are released.

Keywords: log anomaly detection; concept drift; benign-only learning; evaluation methodology; lightweight models

1 Introduction

System logs remain the most universally available signal for detecting failures and attacks at runtime, and a decade of research has produced increasingly heavy detectors for them: LSTM next-key predictors (DeepLog [1]), semantically enriched sequence models (LogAnomaly [2], LogRobust [3]), probabilistic semi-supervised pipelines (PLELog [4]), transformer masked-prediction models (LogBERT [5]), and most recently large-language-model pipelines [15–17]. Three observations motivate the present study.

First, the evaluation foundations of this literature are fragile. Le and Zhang's controlled re-evaluation [6] documented that reported results depend heavily on protocol choices—vocabulary leakage from test data, random splits that hide temporal structure, inconsistent windowing and thresholds—and that deep models degrade sharply when training data is selected chronologically. Follow-up studies quantified sensitivity to the upstream parser [9], re-examined classical learners at scale [10], and critically reviewed the benchmark datasets themselves [31].

Second, a growing thread argues that heavy models may be unnecessary on these benchmarks: tuned PCA rivals deep detectors [7]; an ICSE 2024 empirical study reaches similar conclusions with simple

supervised learners [8]; and NLLog demonstrates a CPU-deployable semantic pipeline with competitive supervised accuracy [18]. What this thread has not established is how far *benign-only* lightweight detection—the constraint matching deployments where labeled anomalies do not exist in advance—can be pushed under a protocol that closes every leak class catalogued by prior re-evaluations, and precisely where and why it stops working.

Third, the failure regime itself is under-explained. It is known *that* time-based evaluation on BGL is severe [6], and online methods have been evaluated on chronologically ordered logs [26, 27]; to our knowledge no published work explains the mechanism—what signal survives the drift, what class of learner could in principle exploit it, and what labeling budget would change the answer. Prior work measured the collapse; this paper explains it.

We answer three research questions:

- **RQ1.** Under a leak-free protocol, how does a benign-only stack that trains in under two minutes on CPU compare, on identical splits, with deep baselines and their published results?
- **RQ2.** Which structural design choices determine success under benign-only calibration, independent of model capacity?
- **RQ3.** When lightweight detection fails under temporal drift, what exactly fails—representation, calibration, or the learning objective—and what would fix it?

Contributions.

1. *A mechanistic dissection of the chronological collapse (RQ3; Sec. 5).* The collapse has two distinct mechanisms. On BGL the template vocabulary is replaced mid-trace (42% of templates first appear after the training cut), template-identity evidence is destroyed, and the only transferable signal is discriminative fault semantics—which benign-only learning cannot privilege: six designs without external semantic priors fail concordantly (Table 10), a seventh with a frozen pretrained-embedding prior lifts ranking above chance (PR-AUC 0.34) without becoming deployable, and full period-causal supervision reaches PR-AUC 0.722 with an operating point that does not transfer (F1 0.44 at a causally chosen threshold, 0.76 oracle; Table 12). On Thunderbird the signal survives and the calibration era does not: a labeled counterfactual bounds what era-matched recalibration could recover (PCA F1 0.61; Table 15).

2. *A leak-free, benign-only baseline that locates where learning is needed (RQ1, RQ2; Sec. 4).* Across HDFS, BGL, and Thunderbird under chronological and random splits, an MLP autoencoder designated in advance of the held-out system trains in under two minutes on CPU and reaches F1 0.92–0.97 on three of the four vocabulary-stable conditions, matching or exceeding DeepLog and a LogBERT-style model re-run on byte-identical splits (Tables 3–5, 14; Fig. 2); a one-line unseen-template rule reaches F1 0.90/0.998 on random BGL/Thunderbird but only 0.43 on HDFS, locating the one benchmark whose anomalies require a learned detector. Supporting analyses show the primary-detector conclusions are parser-robust (Table 7) and explain structurally why isolation-based detectors are blind where novelty carries the signal (Sec. 4.2).

3. *A label-free self-monitoring channel and a complete artifact (Sec. 6).* Both chronological failures are silent at the alarm level; two statistics computable in production without labels—the unseen-template share and the rolling benign score distribution and alarm rate—expose them. Every number derives from released scripts, with exported splits, per-seed results, and archived deep-baseline outputs.

In one sentence: this paper gives the first mechanistic account of why benign-only log anomaly detection collapses under chronological evaluation—two distinct, quantified failure mechanisms, evidence across seven designs that only external semantic priors or labels address the first, a bound on what labels buy, and a label-free monitoring channel that exposes both failures in production.

2 Related Work

Deep log anomaly detection. DeepLog [1] models template streams as a language and flags windows whose next key falls outside the top-g predictions. LogAnomaly [2] adds template semantics; LogRobust [3] targets log instability with attention over semantic vectors; PLELog [4] reduces labeling cost via probabilistic label estimation; LogBERT [5] transfers masked-prediction pretraining to log keys. These systems evaluate predominantly on HDFS and BGL under random splits—which is what makes published numbers comparable to ours. LogRobust deserves particular note as the qualitative ancestor of our RQ3 finding: it argued that semantics survive instability, in a supervised setting on synthetically perturbed logs; our mechanism is the benign-only, chronologically grounded sharpening of that intuition.

Benign-only detectors with language-model semantics. LAnoBERT [22] trains a BERT masked language model from scratch on normal logs, and LogFiT [23] fine-tunes a pretrained Longformer on normal logs only; LogGPT [24] and ADALog [25] continue this line. Neither of the former evaluates under a leak-free chronological protocol, so our results neither confirm nor contradict theirs; they occupy the design cell—benign-only learning *with* external semantic priors—that our drift analysis identifies as the theoretically promising one, and that our seventh design probes with a frozen general-purpose encoder (Sec. 5.1).

LLM-based detection. LogLLM [15] fine-tunes large language models for log-based detection, notably using chronological BGL splits to mitigate leakage; a 2026 benchmark evaluates LLM families for automated diagnostics, including few-shot label-budget curves [16]; a systematic review maps the AIOps landscape in the LLM era [17]. These pipelines occupy the opposite end of the cost spectrum from ours; our results calibrate what their added cost must beat.

Online and drift-aware detection. OMLog [26] runs normal-only online meta-learning on chronologically ordered HDFS and BGL framed as evolving data; LogOnline [27] and EvLog [28] pursue online and evolving-log settings; ROEAD [29] and ADA [30] adapt detectors in deployment; temporal-decay loss weighting [19] adapts to evolving cloud logs; FAME [38], a message-level mixture-of-experts detector, reports that 79.8% of anomalous BGL test lines come from templates absent from its offline (first 85%) region—adjacent to, and consistent with, our vocabulary-replacement statistics; and CAShift [39] benchmarks benign-only detectors under normality shift in cloud logs. OMLog reports F1 0.649–0.689 on evolving BGL with normal-only training; those numbers are not comparable to our Table 10 because OMLog's protocol is not leak-free in our sense—window construction, semantic features, and threshold selection all differ—which is exactly the protocol gap Table 1 encodes. These works occupy the chronological-protocol cell but quantify neither the vocabulary replacement nor the signal that survives it; our Table 10 explains why adaptation alone—which we also implement—does not suffice without a semantic or supervised prior.

Evaluation methodology. Beyond the classical experience report of He et al. [32] and surveys [34, 35], Le and Zhang [6] catalogued protocol pitfalls and re-measured five deep systems; Khan et al. [9] quantified parser impact; Ali et al. [10] re-evaluated classical ML broadly; Wu et al. [33] examined representation choices; Albert [43] proposed temporally preceding benchmarks for unsupervised log AD; Sedláček et al. [36] independently ran nine methods, including PCA and SemPCA, on HDFS/BGL/Thunderbird under shuffled and sequential splits and observed the chronological-BGL collapse (label-tuned and without a mechanism, which is the gap we fill); Nyyssölä and Mäntylä [37] evaluated benign-only lightweight detectors including Isolation Forest and an out-of-vocabulary detector on HDFS/BGL, reporting threshold-free AUC-ROC rather than a calibrated operating point; K^4 [40] likewise criticizes unrealistic evaluation protocols and evaluates online; a 2026 study examines sequence-embedding choices [20]; a 2025/26 framework addresses collaborative evaluation [21]; and Landauer et al. [31] argue that common log datasets contain largely trivially detectable anomalies. Our RQ1 result is the quantified benign-only version of that last point—lightweight methods suffice where the vocabulary is stable—and our RQ3 result identifies the non-trivial regime their critique calls for.

To our knowledge, no published work combines all six dimensions of Table 1; the individual protocol ingredients have precedent, and the novelty we claim is the conjunction together with the mechanism and the bound.

Table 1: Protocol dimensions covered by the closest prior work (full coverage here is bought on legacy benchmarks only; Sec. 8). *BO* = benign-only training; *LF* = leak-free protocol (frozen vocabulary, benign-quantile thresholds, no test tuning); *Chr* = chronological splits on HDFS+BGL; *RR* = deep baselines re-run on identical splits; *Mech* = mechanistic account of the chronological-BGL failure; *LE* = label-efficiency bound.

Work	BO	LF	Chr	RR	Mech	LE
Le & Zhang [6]	✗	✓	✓	✓	✗	✗
Yu et al. [8]	✗	partial	✗	✓	✗	✗
OMLog [26]	✓	✗	✓	partial	✗	✗
LogLLM [15]	✗	partial	✓	partial	✗	✗
LAnoBERT [22] / LogFiT [23]	✓	✗	✗	partial	✗	✗
NLLog [18]	✗	partial	partial	✓	✗	✗
Sedláček et al. [36]	✗	✗	✓	✓	✗	✗
Nyyssölä & Mäntylä [37]	✓	✗	partial	✗	✗	✗
LLM benchmark [16]	✗	partial	partial	✓	✗	curve
This paper	✓	✓	✓	✓	✓	✓

Table 2: Benchmark statistics as prepared by the released pipeline.

Property	HDFS_v1	BGL	Thunderbird
Raw lines	11,175,629	4,747,963	10,000,000[a]
Source system	Hadoop DFS; ∼203-node Amazon EC2 cluster, 48 h [11]	BlueGene/L supercomputer, 214 days	Thunderbird supercomputer (Sandia)
Detection unit	block-id session	100-line chronological window [6]	100-line chronological window
Units	575,061	47,134[b]	99,807[b]
Anomalous units	16,838 (2.93%)	4,824 (10.24%)	38,285 (38.36%)
Ground truth	per-block labels (LogHub)	per-line alert tags; window anomalous iff any line tagged	per-line alert tags, same rule
Drain templates (full trace)	45	419	1,437
Benign-train vocabulary (chrono, 60% split)[c]	19	167	1,096

[a] The standard first-10-million-line subset of the 211,212,192-line trace, the convention of [5, 6].
[b] Of the 4,747,963 BGL lines, 4,713,400 parse into the 10-field format, yielding 47,134 non-overlapping 100-line windows; of the 10,000,000 Thunderbird lines, 9,980,700 parse, yielding 99,807 (trailing partial windows dropped).
[c] Distinct templates occurring in the benign units of the training partition; the deep-baseline re-runs use the benign train+calibration vocabulary instead (Sec. 3.4), which is 19 (HDFS chrono), 199 (BGL chrono), and 311 (BGL random).

3 Study Design

3.1 Datasets and detection units

All three datasets are the standard public LogHub distributions [12]; integrity was verified by checksum against the originals. Parsing uses our own implementation of Drain [13] (depth 4, similarity 0.4, standard masking of block ids, IPs, hex and numbers), chosen for auditability; parser sensitivity is a known threat [9] discussed in Sec. 8. A secondary BGL view for the cross-component study uses (node, 3-hour bucket) units with a 10-line minimum (49,082 units, 21.8% anomalous, 27,953 components); prevalences are reported wherever this view appears.

3.2 Leak-free protocol

Units are ordered by *first-line rank*: the index, in the raw chronological log, of the first line belonging to the unit. The *chronological* split takes the first 60% of units for training, the next 10% for calibration, and the final 30% for testing; the *random* split uses identical proportions after seeded shuffling and is reported

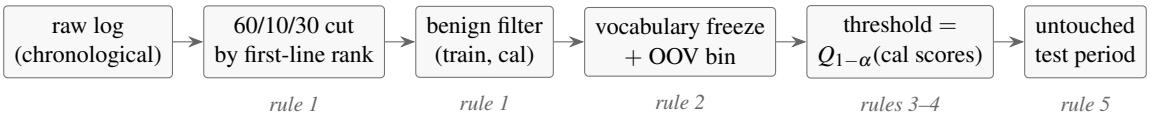


Figure 1: The leak-free protocol: each stage closes one leak class catalogued by [6]. Hyperparameters are fixed a priori and stochastic results carry multi-seed confidence intervals.

only for comparability with published numbers. Prior re-evaluations treated chronological selection as inapplicable to HDFS because block sessions interleave [6]; ordering sessions by first-line rank makes it well-defined (onset order is the deployment-relevant order), a small protocol contribution in its own right. Five rules remove the leaks catalogued by [6] (Fig. 1):

1. Training and calibration sets contain **benign units only**; anomalous units falling in those ranges are discarded, never relabeled. Under the chronological split this discards 13,115 HDFS and 3,296 BGL anomalous units from the train+calibration period.
2. The template vocabulary is **frozen on the training partition**; unseen templates map to a single out-of-vocabulary (OOV) bin.
3. The alarm threshold is the $(1-\alpha)$ quantile of **calibration** scores, $\alpha = 0.005$ fixed a priori; no threshold touches test data.
4. All hyperparameters are fixed a priori (Sec. 3.4); no per-condition tuning.
5. Stochastic detectors run with ten seeds; we report mean F1 with 95% percentile-bootstrap intervals over seeds (seed-to-seed variability, not split variability) and Cliff's δ effect sizes between stochastic detectors. Deterministic detectors are single-run by nature. The random split itself is fixed (split seed 0), so these intervals cover seed variance only (Sec. 8).

Primary-detector designation. The MLP autoencoder is the primary detector. It was designated once, on the HDFS and BGL evidence on which it dominates (Cliff's δ over OC-SVM of $+1.00$ and $+0.80$, computed on that same evidence), before the held-out system was run, and every headline number in this paper is its number—including on Thunderbird, run afterwards under the frozen protocol, where the designation costs us (Sec. 5.2). The other detectors are ablations; we never select the best detector per condition, since no benign-only criterion could make that choice in deployment.

Precision, recall, F1 and FPR are computed at the calibrated threshold; PR-AUC supplements them threshold-free and is reported in every results table. In the four vocabulary-stable conditions the empirical test FPRs (0.001–0.007) cluster around the $\alpha = 0.005$ design point, indicating that benign-quantile calibration *transfers* from the calibration to the test period when the vocabulary is stable. Because α is the one protocol constant that fixes an operating point, Sec. 4.3 reports a sensitivity sweep over $\alpha \in \{0.001, 0.005, 0.01, 0.02, 0.05\}$ from persisted scores; the sweep is diagnostic and no number outside it was chosen after seeing test labels. Environment: Python 3.11.15, scikit-learn 1.8.0, 2-vCPU container; DeepLog re-runs: PyTorch 2.8.0, Apple-silicon CPU.

3.3 Pipeline and features

The pipeline is deliberately plain, and we describe it in prose rather than as a system. A raw log stream is parsed online by Drain into template ids; a unit builder groups events into sessions (HDFS) or windows (BGL, Thunderbird); two feature branches map each unit to a vector—identity features over the frozen training vocabulary and fitting-free content-hashed features; a bank of benign-only detectors scores the unit; and the alarm decision compares the score with the benign-quantile threshold of rule 3. Training and calibration touch only benign units of their periods, and no detector ever consumes a label. Alongside

the detector, two label-free drift statistics are tracked—the unit's unseen-template share and the rolling benign score distribution and alarm rate—whose role as a self-monitoring channel Sec. 6 develops.

Identity features. Frozen-vocabulary TF–IDF template counts (Sec. 3.2) concatenated with three sequence statistics (length, distinct-template count, maximum run length). Dimensionality: 23 (HDFS) up to 1,289 (Thunderbird random).

Content features (drift study). Each template string is hashed into a 128-dimensional signed bag-of-words vector (md5 token hashing with a sign bit; parameter masks excluded); a unit's vector is the sum over its events, signed-log scaled. The map is a pure function of template text—nothing is fitted—so a template first seen at test time lands near training templates sharing its vocabulary. This is the lightweight analogue of the semantic embeddings used by LogRobust [3] and NLLog [18].

3.4 Detectors

Benign-only detectors (hyperparameters fixed a priori): PCA reconstruction with 95% retained variance, scored by squared prediction error [11]; Isolation Forest, 200 trees [14]; one-class SVM (RBF, $\nu = 0.05$, γ scale) trained on a 10,000-unit benign subsample redrawn per seed; an MLP autoencoder 64–16–64 with early stopping and ≤ 60 epochs (on 23-dimensional HDFS input the first layer is over-complete; the 16-unit bottleneck is what forces compression); and an order-2 Markov surprisal model over template transitions—our own deliberately classical sequence baseline, so the sequence-modeling idea is represented at classical cost, distinct from DeepLog. Finally, an **unseen-template rule** flags a unit iff it contains at least k events whose template is outside the frozen training vocabulary (k=1 unless stated). It has no parameters, needs no calibration, and is the leak-free form of the new-event detectors of [31, 37]; we report it in every table because any learned detector must be measured against it.

3.5 Deep baselines

DeepLog is re-implemented to match its published design (2-layer LSTM, hidden 64, window $h = 10$, top-$g = 9$, unseen-next-key windows flag; $\leq$500k training windows subsampled at fixed seed) and re-run by the authors from exported copies of the exact unit splits; the resulting JSONs are archived unmodified, and, because the top-g rule is DeepLog's only operating-point control, every re-run is scored at $g \in \{1,3,5,7,9\}$ from a single trained model (Table 5) so that the comparison cannot hinge on the published g=9. We describe this as a *protocol-matched* re-run: it holds data, splits, vocabulary discipline, and thresholds to our protocol rather than reproducing the original paper's own setup. LogBERT is represented by a protocol-matched *LogBERT-style* model rather than the official codebase: a Transformer encoder with masked-key prediction over the same frozen vocabulary, the center-loss term omitted, and detection by the masked-prediction miss ratio thresholded at the benign-calibration quantile (rule 3). It trains in 26–41 s on a free-tier Colab T4, against $\approx$30 CPU-hours per configuration for the official implementation; we therefore label it *LogBERT-style* throughout and never claim it reproduces LogBERT's training regime. Its published numbers (HDFS 0.823, BGL 0.908 [5]) are reported alongside for the three-family comparison. We additionally ran the *official* LogBERT codebase, unmodified and at its default hyperparameters (center loss included), on the exported BGL splits on a Colab T4 (80–94 minutes per configuration). Its own `predict` step sweeps the anomaly-ratio threshold on the *test* set and reports the best F1, so its numbers are test-tuned upper bounds rather than rule-3 operating points; they appear as separate rows in Table 4. The corresponding HDFS runs could not be completed—the official training-set assembly rejects HDFS's variable-length sessions under current NumPy—so on HDFS LogBERT is represented by the stand-in and its published numbers only.

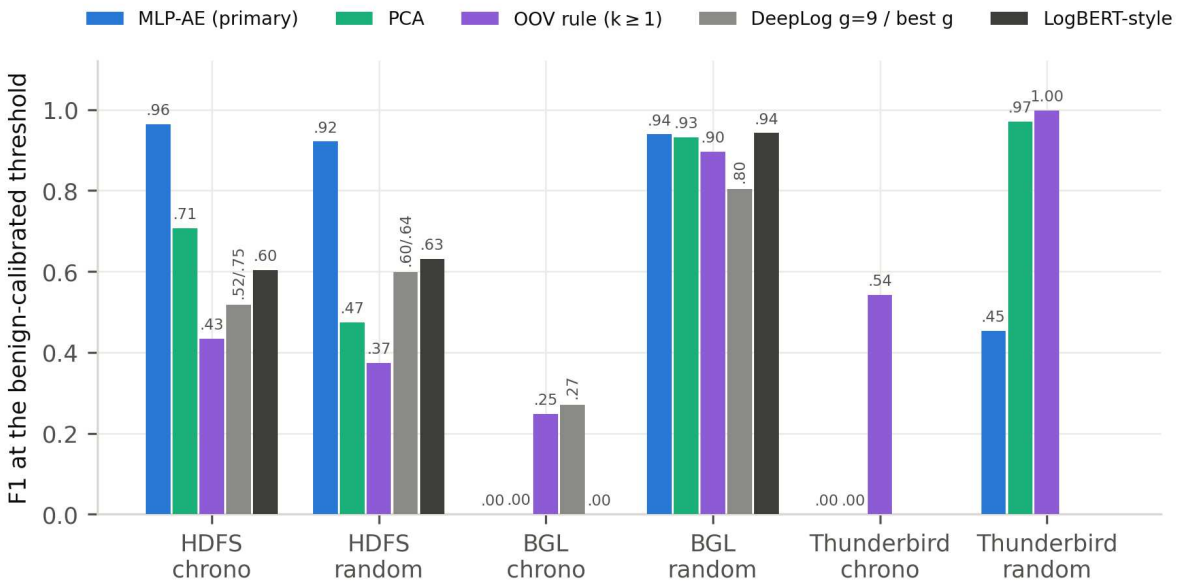


Figure 2: F1 at the benign-calibrated threshold across all six dataset–split conditions: the primary detector, PCA, the parameter-free unseen-template rule, the DeepLog re-run (label: F1 at g=9 / at its best sweep point where different), and the LogBERT-style model. DeepLog's BGL-chrono bar (0.27) and the rule's (0.25) are attained at 65% and 73% FPR and are operationally unusable; every lightweight bar above 0.4 sits at or below 0.8% FPR.

Table 3: HDFS results at the benign-calibrated threshold ($\alpha = 0.005$); 95% seed-bootstrap intervals over 10 seeds where stochastic. F1 entries are means of per-seed F1 and need not equal the F1 of the mean precision/recall. PR-AUC is threshold-free (test prevalence 2.16% chrono, 2.97% random). The primary detector is in bold; the DeepLog re-run is shown at its published g=9 and at its best sweep point (g=3 chrono, g=5 random).

Detector	Split	Precision	Recall	F1	FPR	PR-AUC
MLP-AE (primary)	chrono	0.932	1.000	**0.965 [0.956, 0.973]**	0.0016	0.996
PCA	chrono	0.830	0.617	0.708	0.0028	0.945
OC-SVM	chrono	0.942	0.817	0.875 [0.875, 0.875]	0.0011	0.973
Markov (k=2)	chrono	0.565	0.397	0.466	0.0067	0.496
Isolation Forest	chrono	0.097	0.008	0.015 [0.012, 0.018]	0.0017	0.122
OOV rule ($k \geq 1$)	chrono	1.000	0.277	0.434	0.0000	0.293
DeepLog re-run (g=9)	chrono	0.764	0.391	0.517	0.0027	—
DeepLog re-run, best g=3	chrono	0.693	0.817	0.750	0.0080	—
LogBERT-style	chrono	0.965	0.439	0.604	0.0003	0.448
MLP-AE (primary)	random	0.854	1.000	**0.921 [0.918, 0.924]**	0.0052	0.990
PCA	random	0.699	0.360	0.475	0.0047	0.692
OC-SVM	random	0.824	0.827	0.825 [0.823, 0.828]	0.0054	0.943
Markov (k=2)	random	0.698	0.394	0.503	0.0052	0.519
Isolation Forest	random	0.027	0.004	0.007 [0.006, 0.008]	0.0044	0.189
OOV rule ($k \geq 1$)	random	1.000	0.230	0.374	0.0000	0.253
DeepLog re-run (g=9)	random	0.905	0.448	0.599	0.0014	—
DeepLog re-run, best g=5	random	0.811	0.531	0.642	0.0038	—
LogBERT-style	random	0.972	0.468	0.632	0.0004	0.480

Table 4: BGL results (100-line global windows; test prevalence 10.2% random, 10.8% chrono). Conventions as in Table 3; all stochastic rows now carry ten seeds under both splits. The official-LogBERT rows are single runs of the unmodified official codebase, whose predict step tunes its threshold on the test set (test-tuned, hence optimistic); that code does not expose PR-AUC.

Detector	Split	Precision	Recall	F1	FPR	PR-AUC
MLP-AE (primary)	random	0.961	0.918	**0.939 [0.938, 0.939]**	0.0043	0.969
PCA	random	0.962	0.904	0.932	0.0041	0.983
OC-SVM	random	0.951	0.908	0.929 [0.926, 0.933]	0.0053	0.958
Markov (k=2)	random	0.949	0.882	0.914	0.0054	0.954
Isolation Forest	random	0.460	0.056	0.100 [0.094, 0.106]	0.0076	0.145
OOV rule ($k\geq1$)	random	0.988	0.819	0.896	0.0011	0.836
DeepLog re-run (g=9)	random	0.677	0.991	0.805	0.0538	—
LogBERT-style	random	0.949	0.936	0.942	0.0058	0.985
LogBERT official (test-tuned)	random	0.941	0.968	0.954	0.0069	—
MLP-AE (primary)	chrono	0.000	0.000	**0.000 [0.000, 0.000]**	0.0000	0.162
PCA	chrono	1.000	0.001	0.001	0.0000	0.125
OC-SVM	chrono	0.000	0.000	0.000 [0.000, 0.000]	0.0000	0.135
Markov (k=2)	chrono	0.031	0.002	0.004	0.0074	0.114
Isolation Forest	chrono	0.451	0.033	0.061 [0.054, 0.068]	0.0048	0.176
OOV rule ($k\geq1$)	chrono	0.141	0.993	0.248	0.7307	0.109
DeepLog re-run (g=9)	chrono	0.156	0.992	0.270	0.6484	—
LogBERT-style	chrono	0.000	0.000	0.000	0.0000	0.101
LogBERT official (test-tuned)	chrono	0.175	0.782	0.286	0.4477	—

Table 5: DeepLog: published, re-measured, and protocol-matched re-run.

Source	HDFS	BGL	Protocol
Original paper [1]	≈0.96	≈0.90–0.93 with online updating; ≈0.27–0.28 without[a]	small benign train subset, session level, own protocol
Le & Zhang [6]	configuration-sensitive; substantial degradation under controlled/time-based settings		controlled re-evaluation
Our re-run, g=9 (chrono / random)	0.517 / 0.599	0.270 / 0.805	byte-identical units and splits to Tables 3–4; h=10, 5 epochs; benign train+cal vocabulary 19 (HDFS), 199/311 (BGL chrono/random); confusion counts archived
Our re-run, sweep $g \in \{1,3,5,7,9\}$ (chrono / random)	0.39 / 0.75 / 0.65 / 0.52 / 0.52 (chrono); 0.12 / 0.34 / 0.64 / 0.61 / 0.60 (random)	0.25 / 0.26 / 0.27 / 0.27 / 0.27 (chrono); 0.44 / 0.70 / 0.75 / 0.78 / 0.80 (random)	same trained model scored at every g (F1 at g=1,3,5,7,9); the best point moves with the condition—g=3 on chronological HDFS (0.75 at 0.8% FPR), g=5 on random HDFS (0.64 at 0.4%), g=9 on random BGL—and chronological BGL floods at every g (FPR 0.65–0.73)
Our re-run on LogHub's reference HDFS parse, g=9 / g=3	0.52 / 0.67 (chrono); 0.59 / 0.66 (random; best g=5)	—	29 event types (17 in benign training); parser check of Table 7

[a] The original static-model BGL figure (≈0.27, before online updating, trained on the first 1% of normal lines) coincides with our chronological re-run (0.270) despite the very different training fraction. The agreement is arithmetic rather than a replication: once the unseen-key rule fires on most drifted windows, recall ≈ 1 and F1 $\approx 2p/\big(2p+(1-p)\,\mathrm{FPR}\big)$ at test prevalence p—0.27 at our $p = 0.108$, FPR = 0.65—so any implementation that flags the drifted majority lands near this value; both numbers describe a detector that is operationally unusable.

4 Where Lightweight Detection Suffices

4.1 RQ1: Lightweight versus deep under a leak-free protocol

Four observations. **(i)** The primary detector is the strongest learned detector in every vocabulary-stable condition except Thunderbird-random, with perfect recall on HDFS under both splits. Its four-point F1 difference between the HDFS splits (0.921 random, 0.965 chronological) is an operating-point effect, not robustness: recall is 1.000 in both, and the calibrated threshold lands at FPR 0.0016 on the chronological split against 0.0052 on the random one, at test prevalences of 2.16% and 2.97%. **(ii)** The unseen-template rule partitions the benchmarks. On random BGL and random Thunderbird it reaches F1 0.896 and 0.998 at 0.1–0.3% FPR—as good as or better than any learned detector, the quantified benign-only form of the dataset-triviality critique of [31]. On HDFS it reaches only 0.43 (recall 0.28 at zero FPR): HDFS anomalies are predominantly *structural*—missing or repeated events within a known vocabulary—which is exactly what a reconstruction model over count features captures (recall 1.000) and a novelty rule cannot. The learned stack earns its place on HDFS and nowhere else among the stable conditions. **(iii)** The DeepLog re-run trails its published HDFS figure at the published operating point (g=9: F1 0.52/0.60), but the sweep in Table 5 shows why and how far this can be pushed: over a 19-key benign vocabulary, g=9 accepts half the vocabulary at every step and caps recall at 0.39–0.45; at g=3 recall rises to 0.82 and F1 to 0.75 on chronological HDFS at 0.8% FPR, while on random HDFS the best point is g=5 (0.64 at 0.4% FPR; g=3 there costs 9% FPR). No operating point reaches the primary detector, and on chronological BGL the unseen-key rule inverts the failure into a 65–73% FPR flood at every g. On random BGL every lightweight model except Isolation Forest beats the DeepLog re-run by 11–13 F1 points at a tenth of its FPR. **(iv)** The LogBERT-style model behaves consistently: competitive on random BGL (0.942, matching its published 0.908 and our primary detector), recall-limited on HDFS (0.60–0.63 versus its published 0.823), and dead on chronological BGL. The official LogBERT run confirms the stand-in on BGL: F1 0.954 on the random split—within 1.2 points of the stand-in and above its published 0.908—and a collapse to 0.286 on the chronological split even with a test-tuned threshold. Between the two strong stochastic detectors the ordering is decisive: Cliff's δ for MLP-AE over OC-SVM is $+1.00$ (HDFS, both splits) and $+0.80$ (BGL random) over ten seeds. The simple-versus-deep *phenomenon* has independent support [7, 8, 36, 37]; what is new in RQ1 is the benign-only discipline, the primary-detector designation made in advance of the held-out system, the leak-free rule baseline that shows where learning is needed, and the same-split deep re-runs swept over their operating point.

Finding 1 (RQ1) Under a leak-free protocol with a primary detector designated in advance of the held-out system, a benign-only CPU autoencoder matches or exceeds deep baselines wherever the template vocabulary is stable—F1 0.92–0.97 at 0.2–0.5% FPR on three of four such conditions—but so does a parameter-free unseen-template rule on random BGL and Thunderbird; only HDFS, whose anomalies are structural, needs a learned detector. Among the vocabulary-stable conditions, the split convention changes the deep baselines far more than it changes the lightweight stack.

4.2 RQ2: Structure decides—the Isolation Forest blind spot

Where does the anomaly evidence live? A single ablation answers it: retraining the primary detector on the identical features with the out-of-vocabulary (OOV) column removed. On random BGL its PR-AUC falls from 0.969 to 0.392 and F1 from 0.939 to 0.412; on random Thunderbird the drop is negligible (0.965 to 0.960) because unseen-template evidence there is redundantly encoded in the count profile; on HDFS the ablation costs nothing (PR-AUC 0.996 to 0.995 chronological, 0.990 to 0.985 random). Unseen-template activity is therefore the dominant evidence on BGL and Thunderbird—consistent with the rule's F1 of 0.90 and 0.998—and a minor one on HDFS.

That feature is constant zero on benign training data. Axis-parallel isolation trees never place a split on a constant feature, and tree structure is invariant to that dimension at test time, so no amount of OOV mass can change any path length: on BGL, Isolation Forest is blind to the primary signal *by construction*, not by capacity (F1 0.00–0.10, PR-AUC 0.03–0.32 across the four BGL/Thunderbird conditions,

Table 6: Sensitivity of F1 to the calibration level α (seed 0; scores persisted, thresholds recomputed). The last column is the best F1 at any threshold (oracle, threshold-free).

Detector	Condition	α=0.001	0.005 (used)	0.01	0.02	0.05	best
MLP-AE	HDFS chrono	0.99	0.95	0.91	0.91	0.44	1.00
MLP-AE	HDFS random	0.98	0.92	0.86	0.75	0.62	0.99
MLP-AE	BGL random	0.89	0.94	0.93	0.89	0.82	0.94
MLP-AE	Thunderbird random	0.00	0.45	0.94	0.96	0.95	0.97
MLP-AE	BGL chrono	0.00	0.00	0.00	0.00	0.05	0.25
MLP-AE	Thunderbird chrono	0.00	0.00	0.00	0.00	0.00	0.49
PCA	HDFS chrono	0.75	0.71	0.93	0.93	0.81	0.93
PCA	HDFS random	0.52	0.47	0.59	0.67	0.54	0.67
PCA	BGL random	0.92	0.93	0.93	0.90	0.83	0.93
PCA	Thunderbird random	0.29	0.97	0.98	0.97	0.95	0.98
PCA	BGL chrono	0.00	0.00	0.00	0.02	0.07	0.24
PCA	Thunderbird chrono	0.00	0.00	0.00	0.00	0.00	0.61

on features from which reconstruction models extract 0.93–0.97 wherever any detector succeeds). Reconstruction and distance models respond because OOV mass directly inflates residual error or distance; explicit rules respond trivially. On HDFS the same detector also fails, but for a different reason that the mechanism does not cover: its ranking is at chance (PR-AUC 0.11–0.19 at 2–3% prevalence) on count profiles where the autoencoder ranks near-perfectly (0.99–1.00). On Thunderbird the constant-feature argument does not bite either: the unseen-template evidence is redundantly encoded in count features the forest can see (the ablation above), so its failure there (PR-AUC 0.318 random, 0.033 chronological) belongs to the same chance-ranking regime as HDFS, not to the structural blind spot. The tree property is elementary once stated; axis-parallel limitations are exactly what motivate extended and deep isolation variants [41, 42], and Nyyssölä and Mäntylä observed the Isolation-Forest-versus-OOV gap empirically without explaining it [37]. The contribution we claim is the ablation-backed design rule of Finding 2, scoped to the regime where novelty carries the signal.

Finding 2 (RQ2) Where the anomaly evidence lives in a feature that is constant on benign training data (the OOV bin on BGL), benign-only detection must route it to score functions that respond to such features—reconstruction, distance, or an explicit rule—and never to axis-parallel split-based isolation models, which cannot learn splits on constant features. Where anomalies are structural (HDFS) or the novelty evidence is redundantly encoded in ordinary count features (Thunderbird), isolation models fail for ordinary ranking reasons, and on HDFS the rule is not a substitute.

4.3 Operating-point and parser sensitivity

The calibration level α. Every headline number uses the a-priori $\alpha = 0.005$. Table 6 re-scores the persisted per-unit scores of the primary detector and of PCA at five levels. Two regularities matter. First, on HDFS and random BGL the primary detector stays within 0.86–0.99 for $\alpha \in [0.001, 0.01]$, and its one weak headline—Thunderbird random, 0.45—is entirely an operating-point effect: 0.94 at $\alpha = 0.01$ and 0.96 at 0.02, against a threshold-free best of 0.97. PCA shows the same brittleness on chronological HDFS (0.71 at 0.005, 0.93 at 0.01). Second, on the two drift conditions no α helps: the best achievable F1 at *any* threshold is 0.25 (BGL) and 0.49–0.61 (Thunderbird), so the chronological collapse is not a calibration-level artifact. Sec. 5 dissects what it is instead, and why its diagnosis on Thunderbird requires a recalibration counterfactual rather than a sweep.

The upstream parser. Parser choice is a known threat [9]. We re-ran the full HDFS protocol on LogHub's reference parse (the released `Event_traces.csv`, 29 event types, of which 17 occur in benign training, against 45 and 19 for our Drain configuration), with the identical unit order and splits (Table 7). The primary detector is unchanged (F1 0.974 / 0.927 against 0.965 / 0.921), and so are OC-SVM, the Markov model, Isolation Forest, and the unseen-template rule—to three decimals, because the two parses agree on which sessions contain novel events. PCA is the exception: on chronological HDFS it rises from

Table 7: Parser sensitivity on HDFS: F1 / PR-AUC at $\alpha = 0.005$ with our Drain parse (as in Table 3) versus LogHub's reference parse, identical unit order and splits (autoencoder: mean over seeds 0–2 for the reference parse).

Detector	Split	Our Drain parse (45 templates)	LogHub reference parse (29)
MLP-AE (primary)	chrono	0.965 / 0.996	0.974 / 0.997
PCA	chrono	0.708 / 0.945	0.955 / 0.987
OC-SVM	chrono	0.875 / 0.973	0.875 / 0.973
Markov (k=2)	chrono	0.466 / 0.496	0.466 / 0.495
Isolation Forest	chrono	0.015 / 0.122	0.024 / 0.129
OOV rule ($k \geq 1$)	chrono	0.434 / —	0.434 / —
MLP-AE (primary)	random	0.921 / 0.990	0.927 / 0.991
PCA	random	0.475 / 0.692	0.480 / 0.752
OC-SVM	random	0.825 / 0.943	0.821 / 0.955
Markov (k=2)	random	0.503 / 0.519	0.503 / 0.518
Isolation Forest	random	0.007 / 0.189	0.008 / 0.194
OOV rule ($k \geq 1$)	random	0.374 / —	0.374 / —

Table 8: Wall-clock cost (train + score one split; seconds unless stated).

Method	HDFS (575k units)	BGL (47k units)	Hardware
PCA	2–4	1–4	2-vCPU container
Isolation Forest	4	1	2-vCPU container
OC-SVM (10k subsample)	15	7–10	2-vCPU container
MLP-AE	61–108	3–6	2-vCPU container
Markov (k=2)	8–9	6	2-vCPU container
Unseen-template rule	<1	<1	2-vCPU container
Rolling refresh (Sec. 5.1)	—	0.8–2.9 per refresh	2-vCPU container
DeepLog re-run (train, 500k windows)	42	44	Apple-silicon CPU
LogBERT-style (protocol-matched, train)	39–41	26–27	Colab T4 (free tier)
Official LogBERT codebase (train + predict)	—	4,774–5,619	Colab T4; BGL splits
Seventh design: template embedding (one-off)	—	<1 min	Colab T4

0.708 to 0.955 (PR-AUC 0.945 to 0.987), because its 95%-variance subspace is sensitive to how the count mass is partitioned among templates. The DeepLog re-run on the reference parse tells the same story as on ours (Table 5: F1 0.52 at g=9 and 0.67 at the best sweep point on chronological HDFS, against 0.52 and 0.75 with our parse), so DeepLog's recall ceiling under a frozen benign vocabulary is not a parser artifact either. Conclusions about the primary detector are therefore parser-robust; conclusions about PCA are not, and we do not draw any.

4.4 Runtime

Table 8 summarizes wall-clock costs: the stack's slowest member trains in under two minutes on commodity CPUs, every other detector in seconds, and each drift-study refresh in 0.8–2.9 s—the cost structure that makes the continual-recalibration argument of Sec. 7 operational.

5 Why Benign-Only Detection Collapses Under Chronological Drift

The two chronological conditions are not harder instances of the random ones; they fail for two different, identifiable reasons. This section dissects BGL first (vocabulary replacement), then Thunderbird (calibration-era mismatch), probes the boundary with seven benign-only designs, and bounds what supervision would buy.

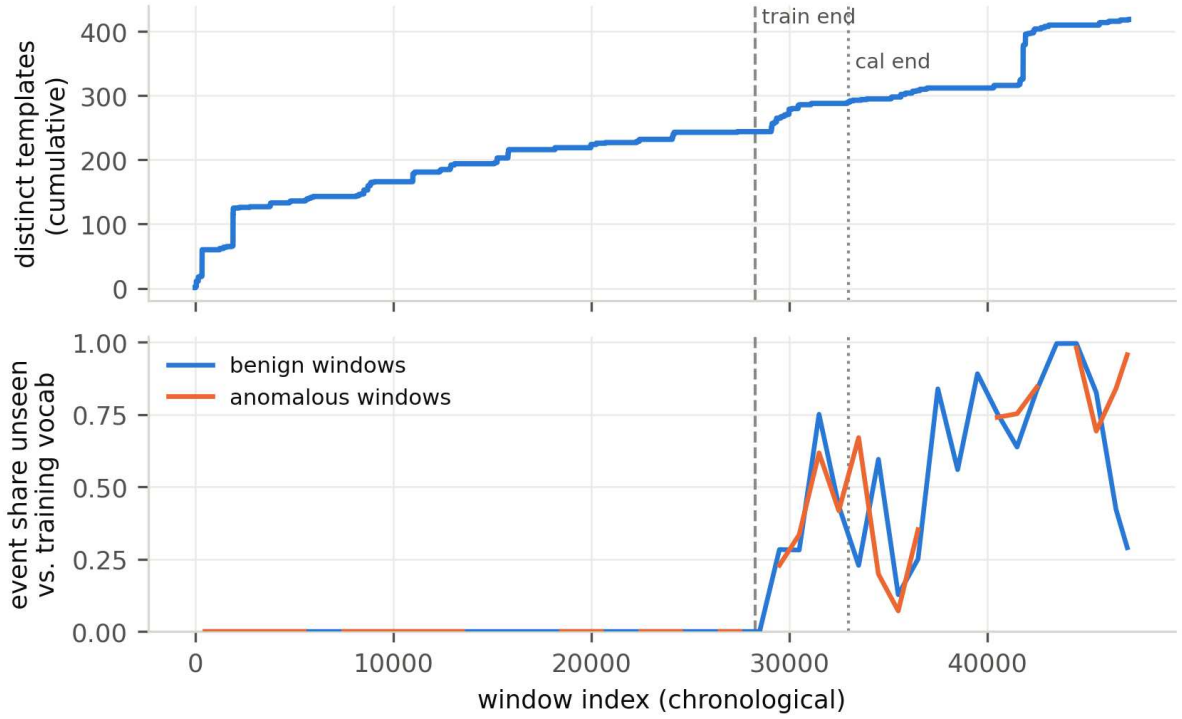


Figure 3: Top: cumulative distinct templates over the BGL trace with split cuts; the vocabulary keeps growing past the training cut (steps near windows 30k and 42k). Bottom: per-1000-window share of events unseen versus the training vocabulary—benign and anomalous windows overlap almost completely, which is the confound.

5.1 RQ3: The chronological-BGL boundary, dissected

The collapse and its two faces. All five lightweight detectors, the unseen-template rule, and both deep re-runs fail (Table 4). The failure has two opposite signatures with one cause. The lightweight stack *under-flags*: drifted benign windows score so high that the benign-quantile threshold, recalibrated on the also-drifting calibration period, rises above nearly all test mass and recall goes to zero (Fig. 4). DeepLog *over-flags*: its hard rule "unseen next key $\Rightarrow$ anomalous window" fires on the drifted benign majority (recall 0.992 at 64.8% FPR; its chronological training vocabulary holds 199 of the 311 keys that appear under the random split). The LogBERT-style re-run—the same model family but thresholded at the benign-calibration quantile under our protocol—fails *silently* instead (F1 0.000 at 0.0% FPR): the failure signature follows the thresholding discipline, not the model class. The official LogBERT run makes the same point a third way: free to tune its threshold on the test set, it lands in DeepLog's over-flagging regime (recall 0.782 at 44.8% FPR, F1 0.286)—no thresholding discipline rescues the replaced vocabulary.

The cause is vocabulary replacement. The vocabulary is effectively replaced mid-trace (Fig. 3): 42% of templates first appear after the training cutoff, and the unseen-template evidence that separates classes in every other condition is carried almost equally by benign (0.66) and anomalous (0.76) test windows (Table 9). The supervised probe—a 300-tree random forest trained, for diagnosis only, on all labeled units of the train+calibration period and evaluated on the untouched test period—confirms that identity features retain no transferable signal (PR-AUC 0.244 at 10.8% prevalence). The same probe on content-hashed features reaches 0.846: the signal survives drift *at the level of words*, with 98.4% of probe importance in content dimensions dominated by fault vocabulary (Fig. 5): *tlb, corrupt, hardware, parity-alignment, segmentation, terminated, failure*.

Why benign-only learning without semantic priors cannot use it. The surviving signal is *discriminative*: exploiting it requires knowing that fault-vocabulary directions matter, which labels (or external

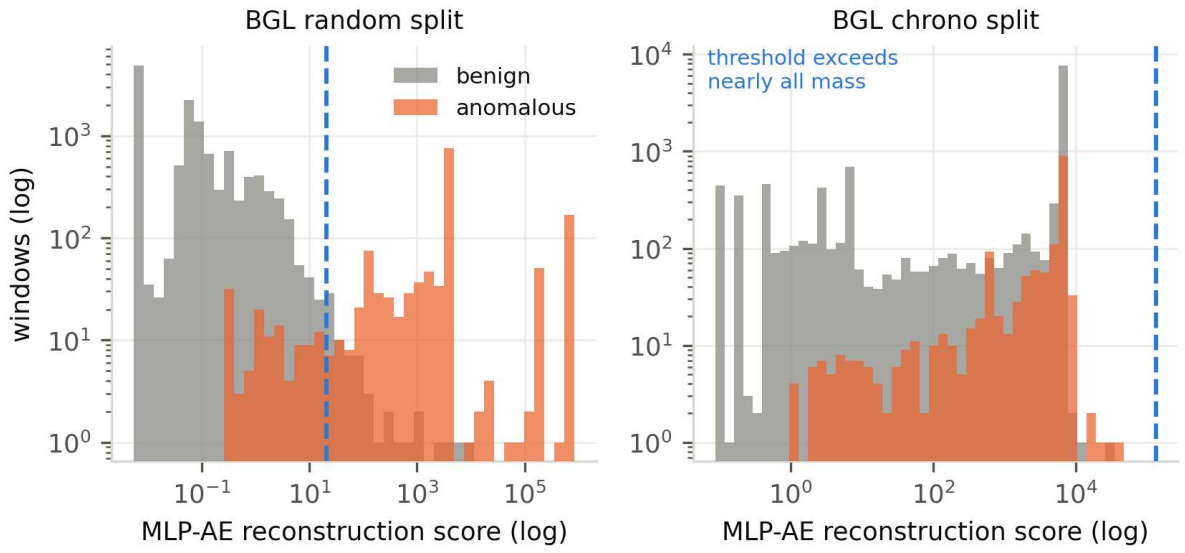


Figure 4: MLP-AE score distributions on BGL test windows. Left (random split): anomalous mass separates cleanly right of the calibrated threshold. Right (chronological): benign and anomalous distributions merge and the threshold exceeds nearly all mass.

Table 9: Drift diagnosis on chronological BGL.

Quantity	Value
Templates first appearing after the training cutoff	175 / 419 (42%)
Median benign test window: share of events from post-cutoff templates	1.00
Mean event share unseen vs. training vocabulary — benign test windows	0.66
Mean event share unseen vs. training vocabulary — anomalous test windows	0.76
Alert-bearing templates[a] also present in benign training vocabulary	150 / 369
Supervised probe on **identity** features (period-causal, diagnostic)	PR-AUC 0.244
Supervised probe on **content** features (period-causal, diagnostic)	PR-AUC 0.846

[a] Denominator: the 369 distinct templates that occur in at least one anomalous window across the full trace.

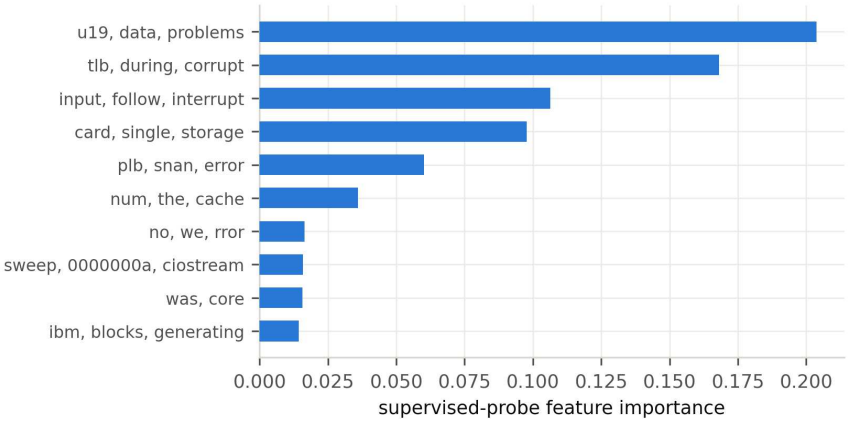


Figure 5: Top supervised-probe features on chronological BGL (content-hashed dimensions labeled with their dominant vocabulary). The transferable signal is fault semantics, not sequence shape—sequence statistics carry 1.6% of total importance. Bar labels list up to three of the shortest words hashed into each dimension; the running-text vocabulary draws on the full importance ranking. Tokens are lowercase alphanumeric runs from template text, so fragments (e.g., "rror") can appear where masking splits words.

Table 10: Six benign-only designs (no external semantic priors) on chronological BGL; best model per row.

#	Design	F1	FPR	PR-AUC	Failure mode
1	Static reconstruction, identity features	0.001	0.0000	0.17	threshold saturation (under-flagging)
2	Static reconstruction, content features	0.126	0.2427	0.17	benign drift indistinguishable from faults
3	Cross-component coalition amplifier	0.006	0.0960	0.07	amplifies drift, not anomalies (Table 13)
4	Bucket-median drift normalization	0.040	0.0048	0.10	no signal in base score ordering to normalize
5	Rolling self-cleaned retrain, identity	0.210	0.4479	0.12	vocabulary churn outpaces adaptation
6	Rolling self-cleaned retrain, content	0.185	0.5226	0.17	adapts to drift and to faults alike

Table 11: Cut-position sweep on chronological BGL (identity features). No cut position yields a deployable operating point; PR-AUC never leaves the chance band.

Cut	Post-cutoff templates	Unseen share (ben./anom.)	PCA		MLP-AE	
			F1	PR-AUC	F1	PR-AUC
50/10/40	187/419	0.59 / 0.72	0.245 @ 64.9% FPR	0.113	0.052	0.106
60/10/30	175/419	0.66 / 0.76	0.001	0.125	0.000	0.165
70/10/20	129/419	0.60 / 0.83	0.112 @ 30.7% FPR	0.128	0.071	0.170

semantic priors) teach and benign data alone cannot—benign-only scores respond to *any* deviation from recent benign content, and under drift the benign distribution keeps deviating from itself. The prediction is that no benign-only design over such fitted-on-benign features closes the gap; six designs—spanning static and adaptive, identity and content—confirm it (Table 10).

Rolling refreshes cost 0.8–2.9 s each: affordability was never the obstacle; the learning objective is.

The seventh design: adding the predicted escape hatch. The analysis above predicts that an *external semantic prior* is the missing ingredient. We test that prediction directly with a seventh design: the same benign-only reconstruction machinery over frozen pretrained sentence embeddings of template text (all-MiniLM-L6-v2, embedded once, no fine-tuning). The prediction is partially confirmed—this is the first benign-only design whose ranking rises meaningfully above chance on chronological BGL (PCA base: PR-AUC 0.337 versus $\leq$0.17 for designs 1–6 and 0.108 chance; F1 0.369 at 0.15% FPR), and on random BGL the same features perform well (MLP-AE F1 0.890), confirming the pipeline is sound. But the escape is partial: recall reaches only 0.229, the autoencoder base remains threshold-saturated (F1 0.000), and the ranking stays far below the supervised content-probe ceiling of 0.846. A general-purpose sentence encoder recovers some fault semantics, not enough for deployment: the boundary moves, and does not fall.

The collapse is a regime, not a cut artifact. Because Fig. 3 shows vocabulary steps near windows 30k and 42k, the 60/10/30 cut could in principle sit at an unlucky position. Repeating the identity-feature protocol at 50/10/40 and 70/10/20 (Table 11) shows otherwise: at every cut position 31–45% of templates are post-cutoff, benign and anomalous unseen-shares overlap, and ranking stays at chance (PR-AUC 0.11–0.17); only the failure signature varies between silent threshold saturation and high-FPR flooding.

Benign-only detectors that fine-tune log-domain language models (LAnoBERT [22], LogFiT [23]) sit outside this family and have not been evaluated under this protocol; the seventh design above is the frozen-encoder end of that spectrum, and the fine-tuned end is the next experiment our analysis motivates (Sec. 7).

Even supervision converts the signal only partially. Adding period-causal labeled anomalies (never from the test period) raises ranking quality with budget: a random forest on content features reaches PR-AUC 0.722 with all 3,296 period labels (Table 12, Fig. 6), against the 0.846 of the probe of

Table 12: Label-efficiency on chronological BGL (content features; random forest; mean over 3 sampling seeds; the non-monotonic PR-AUC at N=100 reflects which fault modes the sampled labels happen to cover—per-seed values are released). $F1_{causal}$: threshold at the F1-maximum of out-of-bag predictions on the labeled period; $F1_{oracle}$: best F1 on the test period (upper bound); $F1_{bq}$: benign-quantile threshold of rule 3 (in-sample for a forest; shown for reference). The zero-label row covers designs 1–6; design 7, the benign-only frozen-embedding prior, reaches PR-AUC 0.337 (Sec. 5.1).

Labeled anomalies (causal)	PR-AUC	$F1_{causal}$	Recall / FPR (causal)	$F1_{oracle}$	$F1_{bq}$
0 (best of designs 1–6, no semantic prior)	$\leq$0.17	—	—	$\leq$0.31	$\leq$0.21
10	0.294	0.016 ± 0.011	0.009 / 0.007	0.416	0.342
25	0.433	0.255 ± 0.181	0.295 / 0.075	0.517	0.328
50	0.533	0.282 ± 0.154	0.173 / 0.000	0.560	0.402
100	0.468	0.314 ± 0.063	0.188 / 0.000	0.484	0.371
250	0.647	0.450 ± 0.044	0.649 / 0.164	0.670	0.387
all 3,296	0.722	0.438	0.908 / 0.272	0.764	0.380

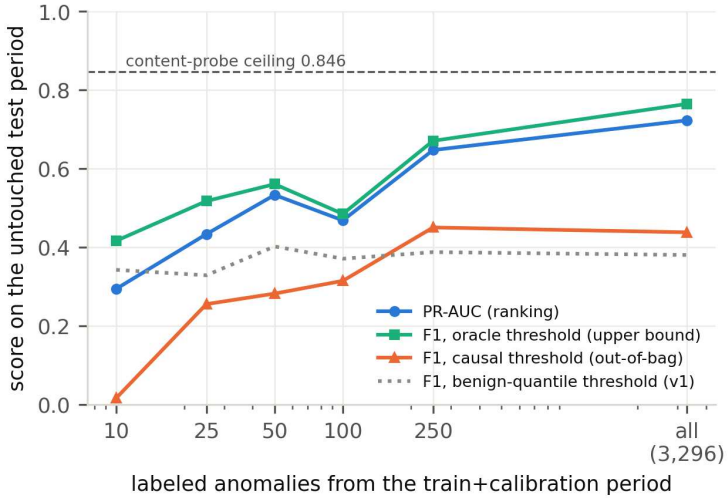


Figure 6: Label-efficiency on chronological BGL. Ranking quality (PR-AUC) climbs toward but never reaches the content-probe ceiling; the oracle F1 tracks it, while the causally chosen threshold plateaus near 0.45 because novel fault vocabulary keeps arriving after the threshold is set.

Table 9, which sees the same labels but is evaluated as a diagnostic. The operating point is the second, separate failure. Choosing the threshold causally—at the F1-maximizing point of the forest's out-of-bag predictions on the labeled period—yields F1 0.44 on the test period, whereas the oracle best-F1 on the test period is 0.76: the ranking transfers across the drift far better than any threshold set before it does, because late-period fault modes introduce new vocabulary of their own. (An earlier draft scored these supervised models at the benign-quantile threshold of rule 3, which is designed for one-class detectors and is in-sample for a forest; that gave F1 $\approx$ 0.38 at 30–40% FPR and understated what supervision achieves. Both operating points are reported.)

Negative result: coalition detection. A cross-component correlation layer—per-component score elevations aggregated in time buckets, gated to multi-component coalitions, motivated by coordinated low-and-slow behavior—*reduced* F1 wherever its base detector had signal and created none where it lacked signal (Table 13). BGL anomalies are loud and local—bursts of near-identical fault lines on affected nodes—not sub-threshold coalitions; bucket-level evidence therefore adds correlated false positives. We report this to close off a plausible-sounding design direction with data.

Table 13: Coalition-layer ablation on the (node, 3 h) BGL view.

Base detector	Split	Base F1	+Coalition F1	Δ
PCA	random	0.988	0.868	−0.120
MLP-AE	random	0.935	0.872	−0.063
OC-SVM	random	0.937	0.720	−0.217
PCA	chrono	0.028	0.006	−0.022
MLP-AE	chrono	0.018	0.000	−0.018

Table 14: Thunderbird results (100-line windows; test prevalence 38.0% random, 4.3% chrono; 95% seed-bootstrap intervals over 10 seeds where stochastic; F1 entries are means of per-seed F1). No deep re-run was executed on Thunderbird.

Detector	Split	Precision	Recall	F1	FPR	PR-AUC
MLP-AE (primary)	random	0.974	0.296	**0.454 [0.452, 0.456]**	0.0048	0.967
PCA	random	0.989	0.952	0.970	0.0062	0.990
OC-SVM	random	0.989	0.824	0.891 [0.828, 0.948]	0.0055	0.984
Markov (k=2)	random	0.000	0.000	0.000	0.0037	0.849
Isolation Forest	random	0.009	0.000	0.000 [0.000, 0.000]	0.0062	0.318
OOV rule ($k\geq1$)	random	0.996	1.000	0.998	0.0026	0.999
MLP-AE (primary)	chrono	0.000	0.000	**0.000 [0.000, 0.000]**	0.0000	0.259
PCA	chrono	0.000	0.000	0.000	0.0000	0.322
OC-SVM	chrono	0.000	0.000	0.000 [0.000, 0.000]	0.0000	0.389
Markov (k=2)	chrono	0.038	0.001	0.002	0.0009	0.234
Isolation Forest	chrono	0.000	0.000	0.000 [0.000, 0.000]	0.0157	0.033
OOV rule ($k\geq1$)	chrono	0.372	1.000	0.542	0.0764	0.420

Finding 3 (BGL: vocabulary replacement) Chronological BGL is not a harder instance but a different problem: under vocabulary replacement the only transferable signal is discriminative fault semantics, which, we argue and show empirically, benign-only learning without external semantic priors cannot privilege—six designs fail concordantly, and a seventh that adds a frozen pretrained-embedding prior lifts ranking above chance (PR-AUC 0.34) without reaching a deployable operating point—and which even full period-causal supervision converts only partially (PR-AUC 0.722), with an operating point that does not transfer across the drift (F1 0.44 causal versus 0.76 oracle).

5.2 Confirmation on a third system: Thunderbird

Thunderbird was held untouched during all design iterations and run once, at the end, under the frozen protocol—the confirmation role the Threats section reserves for it; the diagnostics added afterwards (Table 15, Fig. 7) read persisted scores and change no model. Both halves of the paper's pattern replicate (Table 14). Under the random split the lightweight stack is again competitive—PCA reaches F1 0.970 at 0.6% FPR, OC-SVM 0.891 (its wide interval reflects sensitivity of the 10k training subsample at this scale)—and the unseen-template rule reaches 0.998: on this benchmark the random-split task is novelty detection and nothing more. The primary detector illustrates the operating-point lesson in miniature: near-perfect ranking (PR-AUC 0.967) but recall 0.296 at the fixed $\alpha = 0.005$ threshold, rising to F1 0.94 at $\alpha = 0.01$ and 0.96 at $\alpha = 0.02$ (Sec. 4.3)—the ranking is right and the a-priori operating point is not, and we report the a-priori number as the headline because that is what the protocol commits to. Isolation Forest fails for the sixth condition of six (here, as on HDFS, by ranking at chance: PR-AUC 0.318). Under the chronological split every calibrated detector collapses to F1 0.000 across all ten seeds, silently, at 0.0–1.6% FPR.

A second drift mechanism, and its counterfactual. The diagnosis is not BGL's (Table 15). Only 18.8% of Thunderbird's 1,437 templates first appear after the training cutoff, benign test windows are just 1.4% unseen-template events (versus 66% on BGL), and unseen share remains *discriminative* (anoma-

Table 15: Drift diagnosis and recalibration counterfactual on chronological Thunderbird (compare Table 9 for BGL).

Quantity	Value
Templates first appearing after the training cutoff	270 / 1,437 (18.8%)
Mean event share unseen vs. training vocabulary — benign test windows	0.014
Mean event share unseen vs. training vocabulary — anomalous test windows	0.169
Window prevalence, full subset → test period	38.4% → 4.3%
Unseen-template rule, $k \geq 1$ / $k \geq 5$ (no calibration)	F1 0.542 / 0.658; FPR 7.6% / 4.3%
PCA, calibration-era threshold → recalibrated on first 2,000 test units	F1 0.000 → 0.607 (FPR 5.0%)
Markov, same counterfactual	F1 0.002 → 0.429 (FPR 7.8%)
MLP-AE / OC-SVM, same counterfactual	F1 0.000 → 0.005 / 0.000
Oracle threshold from all benign test windows at $\alpha = 0.005$, any detector	F1 0.000
Best achievable F1 at any threshold (PCA / OC-SVM / MLP-AE)	0.607 / 0.546 / 0.492

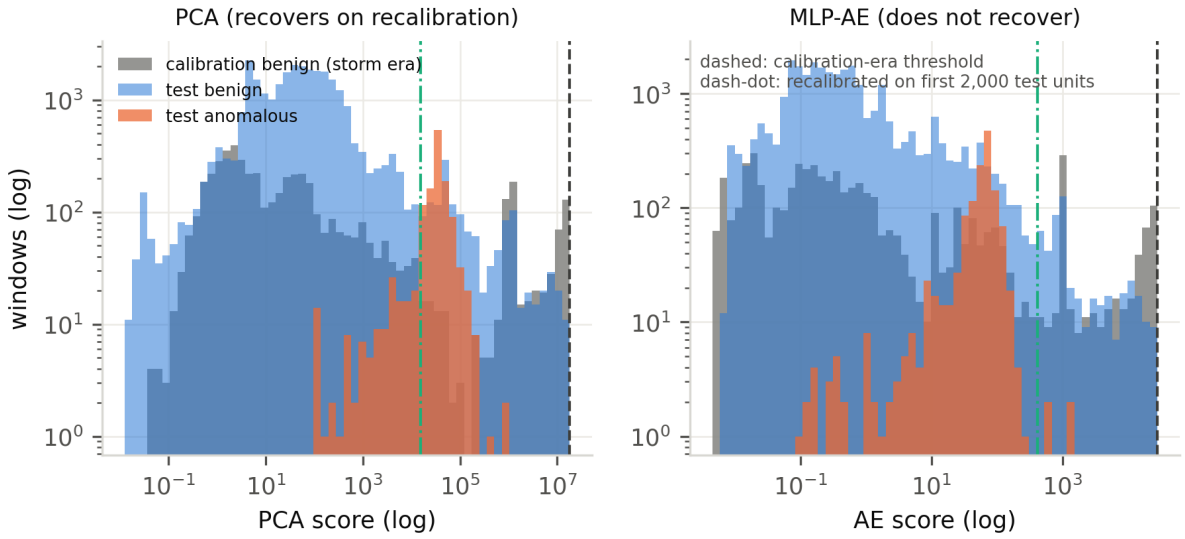


Figure 7: Chronological Thunderbird: score distributions of the calibration-period benign windows (storm era, grey), test-period benign (blue) and test-period anomalous (orange) windows. Left, PCA: the calibration-era threshold (dashed) sits above all test mass; a threshold recalibrated on the first 2,000 test-period units (dash-dot) separates the anomalous mode. Right, MLP-AE: the classes overlap in the test period, so no threshold recovers it.

lous windows: 16.9%)—so discriminative that the parameter-free rule, which needs no calibration, reaches F1 0.54 at $k \geq 1$ and 0.66 at $k \geq 5$ (recall 0.95 at 4.3% FPR) where every calibrated detector scores zero. What breaks is the calibration era: window prevalence falls from 38.4% over the full subset to 4.3% in the test period, because the chronological training and calibration partitions fall inside Thunderbird's alert-dense early life. The benign windows available for calibration are drawn from that storm era, their score distribution is heavy-tailed (Fig. 7, grey), and the $(1-\alpha)$ benign quantile lands above nearly all test-period mass. The counterfactual confirms the mechanism for PCA and bounds it: recalibrating the threshold on the benign windows among the first 2,000 test-period units (the test period's own era, labels used only to select benign units) restores PCA to F1 0.607 at 5.0% FPR and the Markov model to 0.429, whereas the autoencoder and OC-SVM do not recover (F1 $\leq$ 0.005) because their test-period score distributions no longer separate the classes at all (Fig. 7, right; PR-AUC 0.26–0.39 against 4.3% prevalence). Thunderbird therefore fails through calibration-era mismatch *plus* a weakened ranking: recalibration recovers what survives, and what survives is usable only at a few percent FPR. BGL and Thunderbird thus fail through *different* mechanisms—vocabulary replacement versus calibration-era mismatch under prevalence shift—with the same silent outcome.

Finding 4 (Thunderbird: calibration-era mismatch) The chronological collapse of benign-only detection generalizes beyond one system and beyond one cause: BGL fails by vocabulary replacement, Thunderbird by calibration-era mismatch under prevalence shift, both silently. On Thunderbird the signal survives—a calibration-free rule reaches F1 0.66 and recalibrated PCA 0.61—so drift self-monitoring must track more than unseen-template share: the benign score distribution and alarm rate are the channel that would have exposed the Thunderbird failure (Sec. 6).

6 Detecting the Collapse in Production

Both chronological failures are silent at the alarm level: every calibrated detector's recall falls to zero or near zero while its false-positive rate stays at or near the design point (Tables 4 and 14), so the detector looks healthiest exactly when it is dead. Two statistics, both computable in production without labels, expose the two mechanisms.

Channel 1: unseen-template share. The share ρ of incoming events unseen with respect to the frozen training vocabulary is the statistic behind Fig. 3 (bottom). A sustained rise signals BGL-type vocabulary replacement: on chronological BGL the mean benign test-window share reaches 0.66 (Table 9). This channel cannot see the Thunderbird failure, where benign test windows are only 1.4% unseen-template events (Table 15).

Channel 2: benign score distribution and alarm rate. The rolling score distribution and the realized alarm rate likewise require no labels. Calibration-era mismatch announces itself here: on chronological Thunderbird the strongest detectors stop alarming entirely (FPR 0.0000 with recall 0.000, against a 0.5% design rate and 4.3% test prevalence; Table 14)—a detector reporting its own death. This channel also fires on chronological BGL, so it covers both the case channel 1 sees and the one it cannot.

In deployment the monitor is a few lines of code: maintain the rolling mean of ρ and the rolling alarm rate over a window of recent units, and emit a calibration-decay warning on a sustained rise of the former or a sustained collapse of the latter below the design level α.

7 Implications

Implications for practitioners. Wherever the template vocabulary is reasonably stable—which random splits simulate, and which held for HDFS even chronologically—a benign-only CPU autoencoder occupies the same accuracy band as the deep literature at a fraction of its engineering and hardware cost, with second-scale retraining making continual recalibration operationally free. Start lower still: the parameter-free unseen-template rule is the first thing to deploy, because on BGL- and Thunderbird-like systems it is already the detector (F1 0.90 and 0.998 on the random splits, and the only thing that produced alarms on chronological Thunderbird), and its failure to fire on HDFS-like structural anomalies (recall 0.28) is the signal that a learned reconstruction model is needed. Exhaust these two before reaching for GPU-trained sequence models; this extends [7, 8] into the benign-only setting and onto chronological splits, and it is the quantified, benign-only form of the dataset-triviality critique of [31] (whose new-event detector defines novelty against the *full* benign corpus—leaky under our rule 2; our rule freezes the vocabulary at the training cut and still reaches 0.90–0.998). Set the operating point deliberately: Table 6 shows that a fixed α can cost half the F1 on a system whose ranking is near-perfect, which argues for calibrating α to a target alarm budget rather than to a universal constant. Finally, no benign-only detector should ship without the self-monitoring channels of Sec. 6: under drift, silent collapse is the default failure mode, not an edge case.

Implications for researchers. Chronological BGL should be treated as a distinct problem—detection under vocabulary replacement—whose transferable signal is fault *semantics*. Any solution must supply a prior over fault-like content that benign data cannot teach. Three candidate sources follow: learned semantic embeddings in benign-only form (LAnoBERT [22], LogFiT [23]), LLM priors [15–17] whose pretraining already encodes that "machine check parity error" is trouble, or modest labeling budgets, for which Table 12 calibrates expectations and shows diminishing returns. Our released splits

make these directly testable against this paper's numbers, and we have run the first such test: the frozen-pretrained-embedding variant of our own stack (Sec. 5.1) confirms the direction—semantic priors are the one lever that moved ranking on chronological BGL—while showing that a general-purpose sentence encoder captures too little fault semantics for deployment. Fine-tuned log-domain encoders (LAnoBERT, LogFiT) under this leak-free protocol are the natural next step.

Recommendation Report FPR beside F1 whenever drift settings are evaluated. The same drift produces silent under-flagging in calibrated detectors and alarm floods in rule-based ones: DeepLog's BGL-chrono F1 of 0.270 and the unseen-template rule's 0.248 *exceed* our primary detector's 0.000, yet all three are operationally dead—two deafening, one mute. Single-number comparisons hide exactly the failure that matters.

8 Threats to Validity

Construct. Window-level BGL and Thunderbird labels (any tagged line $\Rightarrow$ anomalous window) follow convention but blur burst boundaries; the 100-line windowing, the 3-hour node view, and all prevalences are stated, and unit definitions ship in the artifact. Ordering HDFS sessions by first-line rank makes a chronological split well defined, but sessions overlap in time, so a training session can contain lines later than a test session's first line; the split is chronological in onset, not in every event. **Internal.** The content features and adaptive variants were designed iteratively on the same benchmarks that evaluate them; we mitigate with a-priori hyperparameters, a primary detector designated before the held-out system (no per-condition selection), multi-seed intervals, full disclosure of the design path *including all failures* (Table 10), and release of every intermediate result—Thunderbird served as the untouched confirmation dataset: it was excluded from all design iterations and evaluated once under the frozen protocol; the diagnostics of Sec. 5.2 and the α sweep of Sec. 4.3 re-score persisted outputs and train nothing new. The supervised probes use labels for diagnosis only and are never presented as detectors. The cut-position sweep (Table 11) establishes that the collapse is a regime rather than an artifact of cut placement; the single-trace limitation remains. Rule 1's benign filter uses ground-truth labels to discard anomalous units from the training and calibration periods (13,115 on HDFS and 3,296 on BGL under the chronological split); a deployment cannot perform this filtering, so our results characterize benign-only detection under clean training data. The designation of the primary detector used HDFS and BGL test evidence, and the Cliff's δ values supporting it are computed on that same evidence; Thunderbird alone tests the designation out of sample. **Parser.** Sec. 4.3 quantifies parser sensitivity on HDFS: the primary detector's conclusions survive a change of parser, PCA's do not; BGL and Thunderbird were parsed once, with our Drain configuration only. **External.** Three datasets, all from batch/HPC systems; drift conclusions transfer qualitatively, not numerically, to other domains; results at 2.93%, 10.24%, and 38.36% prevalence do not extrapolate to much rarer-anomaly regimes, and the Thunderbird window prevalence is itself a consequence of the stated any-alert labeling rule at 100-line windows. **Baseline fairness.** Our DeepLog re-run diverges from its published HDFS figure at the published g=9; we therefore report the full top-g sweep, the re-run on LogHub's reference parse, archived confusion counts, and published, re-measured [6], and re-run numbers side by side. The re-implementation is ours, and a reader who trusts only the original code should read our DeepLog rows as a protocol-matched lower bound on what DeepLog can do under a frozen vocabulary; the sweep shows that bound moving from 0.52 to 0.75 on chronological HDFS, still below the primary detector. The LogBERT-style model is a protocol-matched stand-in with stated deviations (center loss omitted; benign-quantile thresholding; seconds rather than hours of training); its published numbers are reported alongside, and the random-BGL agreement (0.942, vs 0.908 published and 0.954 from our own run of the official codebase on the same split) shows the stand-in is not a weak one; on HDFS, where the official codebase could not be run, conclusions about LogBERT proper should be read with this caveat. **Conclusion validity.** Seed-bootstrap intervals (ten seeds) cover seed variance only, not split variance; the random split itself is fixed at seed 0 across detector seeds.

9 Conclusion

The chronological collapse of log anomaly detection, documented by controlled re-evaluations but previously unexplained, is not one phenomenon but two. On BGL the template vocabulary is replaced mid-trace: identity evidence is destroyed, the only transferable signal is discriminative fault semantics, and benign-only learning without external semantic priors cannot privilege it—six designs fail concordantly, a frozen pretrained-embedding prior moves the boundary without removing it, and even full period-causal supervision converts the signal only partially, with an operating point that does not transfer across the drift. On Thunderbird the signal survives and the calibration era does not; a labeled counterfactual bounds what era-matched recalibration could recover. Around that core, a leak-free benign-only evaluation across three systems shows that wherever the template vocabulary is stable, an autoencoder that trains in under two minutes on CPU, designated in advance of the held-out system, matches or exceeds protocol-matched deep re-runs, and a parameter-free unseen-template rule suffices on two of the three benchmarks—leaving HDFS as the one benchmark whose anomalies require a learned detector. Both failures are silent at the alarm level, which is why every deployment recommendation in this paper is paired with the label-free self-monitoring channels of Sec. 6. Together with the released artifact, this gives the field a calibrated account of what lightweight log anomaly detection can do, what it cannot, and why.

Declarations

Data Availability

The accompanying repository contains the parsing, feature, detector, and evaluation code; every experiment script including the failed designs of Table 10; per-seed result files; the exported split files used for the deep-baseline re-runs; the archived, unmodified DeepLog result JSONs and console log; and instructions reproducing every table and figure from the raw LogHub downloads. The artifact is publicly available at `https://github.com/SaritaPhD/bold-logs` and permanently archived on Zenodo with dataset checksums (DOI: 10.5281/zenodo.21896453). Datasets are the standard public LogHub distributions [12]; Thunderbird uses the standard first-10-million-line subset.

Author Contributions

S.N. is the sole author: she conceived and designed the study, implemented the software, performed all experiments and analyses, prepared all figures and tables, and wrote and reviewed the manuscript.

Funding

The author declares that no funds, grants, or other support were received during the preparation of this manuscript.

Competing Interests

The author has no competing interests to declare that are relevant to the content of this article.

References

[1] M. Du, F. Li, G. Zheng, V. Srikumar. DeepLog: Anomaly detection and diagnosis from system logs through deep learning. In *ACM CCS*, 2017.

[2] W. Meng et al. LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In *IJCAI*, 2019.

[3] X. Zhang et al. Robust log-based anomaly detection on unstable log data. In *ESEC/FSE*, 2019.

[4] L. Yang et al. Semi-supervised log-based anomaly detection via probabilistic label estimation. In *ICSE*, 2021.

[5] H. Guo, S. Yuan, X. Wu. LogBERT: Log anomaly detection via BERT. In *IJCNN*, 2021.

[6] V.-H. Le, H. Zhang. Log-based anomaly detection with deep learning: How far are we? In *ICSE*, 2022.

[7] L. Yang et al. Try with simpler—an evaluation of improved principal component analysis in log-based anomaly detection. *ACM TOSEM*, 2024.

[8] B. Yu et al. Deep learning or classical machine learning? An empirical study on log-based anomaly detection. In *ICSE*, 2024.

[9] Z. A. Khan, D. Shin, D. Bianculli, L. Briand. Impact of log parsing on deep learning-based anomaly detection. *Empirical Software Engineering*, 29, 2024.

[10] S. Ali, C. Boufaied, D. Bianculli, P. Branco, L. Briand. A comprehensive study of machine learning techniques for log-based anomaly detection. *Empirical Software Engineering*, 30:129, 2025.

[11] W. Xu, L. Huang, A. Fox, D. Patterson, M. I. Jordan. Detecting large-scale system problems by mining console logs. In *SOSP*, 2009.

[12] J. Zhu, S. He, P. He, J. Liu, M. R. Lyu. Loghub: A large collection of system log datasets for AI-driven log analytics. In *ISSRE*, 2023.

[13] P. He, J. Zhu, Z. Zheng, M. R. Lyu. Drain: An online log parsing approach with fixed depth tree. In *ICWS*, 2017.

[14] F. T. Liu, K. M. Ting, Z.-H. Zhou. Isolation forest. In *ICDM*, 2008.

[15] W. Guan et al. LogLLM: Log-based anomaly detection using large language models. arXiv:2411.08561, 2024.

[16] D. Patel. LLM-enhanced log anomaly detection: A comprehensive benchmark of large language models for automated system diagnostics. arXiv:2604.12218, 2026.

[17] M. De la Cruz Cabello, T. Prince Sales, M. R. Machado. AIOps for log anomaly detection in the era of LLMs: A systematic literature review. *Intelligent Systems with Applications*, 28:200608, 2025.

[18] S. Ndichu, T. Ban, S. Ozawa, T. Takahashi, D. Inoue. NLLog: Lightweight, explainable SOC anomaly detection via log-to-language rewriting. arXiv:2606.04957, 2026.

[19] L. A. Jilcha, D.-H. Kim, J. Kwak. Temporal decay loss for adaptive log anomaly detection in cloud environments. *Sensors*, 25(9):2649, 2025.

[20] M. Alzahrani. Investigating the impact of log-sequence embeddings on anomaly detection: A systematic study. *Information*, 17(3):228, 2026.

[21] A. García Gómez, M. Landauer, M. Wurzenberger, F. Skopik, E. Weippl. Collaborative anomaly detection in log data: Comparative analysis and evaluation framework. *Future Generation Computer Systems*, 175:108090, 2026.

[22] Y. Lee, J. Kim, P. Kang. LAnoBERT: System log anomaly detection based on BERT masked language model. *Applied Soft Computing*, 146, 2023.

[23] C. Almodovar et al. LogFiT: Log anomaly detection using fine-tuned language models. *IEEE Transactions on Network and Service Management*, 2024.

[24] X. Han et al. LogGPT: Log anomaly detection via GPT. arXiv:2309.14482, 2023.

[25] P. Pospieszny, W. Mormul, K. Szyndler, S. Kumar. ADALog: Adaptive unsupervised anomaly detection in logs with self-attention masked language model. arXiv:2505.13496, 2025.

[26] J. Tian, M. Li, Z. Wang, L. Chen, J. Qin, R. Zhang. OMLog: Online log anomaly detection for evolving system with meta-learning. arXiv:2410.16612, 2024; IEEE journal version, 2025.

[27] X. Wang et al. LogOnline: A semi-supervised log-based anomaly detector aided with online learning mechanism. In *ASE*, 2023.

[28] Y. Huo et al. EvLog: Identifying anomalous logs over software evolution. In *ISSRE*, 2023.

[29] S. Han, Q. Wu, H. Zhang, B. Qin, J. Hu, X. Shi, L. Liu, X. Yin. Log-based anomaly detection with robust feature extraction and online learning. *IEEE Transactions on Information Forensics and Security*, 16:2300–2311, 2021.

[30] Y. Yuan et al. ADA: Adaptive deep log anomaly detector. In *IEEE INFOCOM*, 2020.

[31] M. Landauer, F. Skopik, M. Wurzenberger. A critical review of common log data sets used for evaluation of sequence-based anomaly detection techniques. *Proc. ACM Softw. Eng. (FSE)*, 2024.

[32] S. He, J. Zhu, P. He, M. R. Lyu. Experience report: System log analysis for anomaly detection. In *ISSRE*, 2016.

[33] X. Wu et al. On the effectiveness of log representation for log-based anomaly detection. *Empirical Software Engineering*, 28, 2023.

[34] S. He et al. A survey on automated log analysis for reliability engineering. *ACM Computing Surveys*, 54(6), 2021.

[35] M. Landauer et al. Deep learning for anomaly detection in log data: A survey. *Machine Learning with Applications*, 12, 2023.

[36] O. Sedláček, M. Žádník, V. Bartoš. Anomaly detection in log data: A comparative study. In *CNSM*, 2025.

[37] J. Nyyssölä, M. Mäntylä. Speed and performance of parserless and unsupervised anomaly detection methods on software logs. In *IEEE QRS*, 2024. arXiv:2312.01934.

[38] H. Wang, Z. Huang, Y. Tian, K. Dzeparoska, H.-A. Jacobsen, A. Leon-Garcia. FAME: Failure-aware mixture-of-experts for message-level log anomaly detection. arXiv:2605.22779, 2026.

[39] J. Yu, X. Xie, Q. Hu, B. Zhang, Z. Zhao, Y. Lin, L. Ma, R. Feng, F. Liauw. CAShift: Benchmarking log-based cloud attack detection under normality shift. *Proc. ACM Softw. Eng. (FSE)*, 2025. DOI 10.1145/3729346.

[40] W. Chen, V. Singh, Z. Rahmani, D. Ganguly, M. Hariri, V. Chaudhary. K^4: Online log anomaly detection via unsupervised typicality learning. In *IEEE HiPC*, 2025. arXiv:2507.20051.

[41] S. Hariri, M. Carrasco Kind, R. J. Brunner. Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1479–1489, 2021.

[42] H. Xu, G. Pang, Y. Wang, Y. Wang. Deep isolation forest for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12591–12604, 2023.

[43] R.-G. Albert. System logs anomaly detection: Are we on the right path? *Applied Artificial Intelligence*, 39(1):2440692, 2024.