AI and Data Science Journal
ISSN: 3078-7394
DOI: https://doi.org/10.61784/adsj4039

CLARITY: A CLOSED-LOOP FRAMEWORK FOR DUPLICATE-AWARE FAILURE PREDICTION AND BUDGETED AUTOMATED REMEDIATION IN CLOUD INFRASTRUCTURE

Minjae Rhee[1], JiTong Zou[2], YuXuan Qin[3*]
[1]*University of Illinois Urbana-Champaign, Champaign, IL 61820, Illinois, USA.*
[2]*Case Western Reserve University, Cleveland, OH 44106, Ohio, USA.*
[3]*Northeastern University, Boston, MA 02115, Massachusetts, USA.*
[*]*Corresponding Author: YuXuan Qin*

Abstract: Failure predictors for cloud infrastructure are often evaluated as isolated classifiers, although operators need calibrated risk, bounded actions, and feedback after remediation. This paper presents CLARITY, a closed-loop framework that couples duplicate-aware early prediction, probability calibration, a budget-constrained remediation policy, safety guardrails, and post-action feedback. The evaluation uses 575,061 labeled block-level event sequences from the public Loghub HDFS_v1 corpus. Exact duplicate sequences are grouped before splitting, and ten contradictory sequence patterns (46 occurrences) are removed rather than silently assigned. A prefix model combines normalized event frequencies and ordered transitions; XGBoost provides ranking and Platt scaling converts scores into operational probabilities. Five-fold grouped evaluation shows that observing 16 events yields an occurrence-weighted PR-AUC of 0.433±0.066 and a pattern-level PR-AUC of 0.751±0.139. Calibration reduces occurrence-weighted Brier score from 0.467 to 0.019 without changing ranking. In a trace-driven counterfactual policy, a 2% action budget captures 49.26% of anomalous remaining work and produces 11.10% normalized net savings under an explicitly stated 0.8 action efficacy and three-event action cost. Under controlled template-ID churn, feedback raises post-drift pattern PR-AUC from 0.190 to 0.470. The remediation results are policy simulations, not production intervention measurements. The study provides an implementable framework and a reproducible boundary between measured evidence and counterfactual assumptions.
Keywords: Cloud infrastructure; Failure prediction; Log anomaly detection; Automated remediation; Probability calibration; Closed-loop AIOps

1 INTRODUCTION

Large cloud platforms combine high component counts, heterogeneous workloads, shared dependencies, and strict latency objectives. The same scale that improves utilization also creates broad failure surfaces: a small number of slow or unhealthy components can amplify tail latency, trigger retries, and consume recovery capacity [1-3]. Operational telemetry is abundant, but converting it into safe automated action remains difficult. Logs are attractive because they encode control-flow and component-state evidence unavailable in aggregate metrics; however, they are noisy, duplicated, imbalanced, and affected by parser evolution [4-9].

Most log-based studies optimize detection accuracy for a completed session. DeepLog, LogAnomaly, LogRobust, PLELog, and LogBERT illustrate substantial progress in sequence modeling [10-14]. Yet four gaps remain relevant to remediation. First, a detector that waits for the full execution cannot prevent the remaining work. Second, a ranking score is not automatically a probability suitable for comparing expected benefit with action cost. Third, identical traces can cross random partitions and inflate apparent generalization. Fourth, offline evaluation usually ends at a confusion matrix, while production systems receive delayed labels and changed templates after interventions. The empirical review by Le and Zhang further shows that grouping, class distribution, and early-detection protocol can materially change conclusions on HDFS [15].

CLARITY addresses these gaps as one auditable loop. It predicts failure from a fixed event prefix, calibrates risk on a disjoint fold, selects actions under a hard occurrence budget, enforces operational gates, records outcomes, and updates a lightweight online model when the event vocabulary changes. The framework is intentionally conservative: it does not claim that a classifier alone repairs HDFS, and it does not convert a counterfactual cost model into a production availability claim. Actions are abstract remediation primitives—retry, quarantine, or deferred escalation—whose effectiveness and cost remain explicit variables.

The paper makes four contributions. (1) It introduces a duplicate-aware protocol that groups 18,373 exact full-sequence patterns and removes contradictory labels before splitting. (2) It develops and ablates a compact early prefix representation, retaining normalized event counts and ordered transitions without requiring a large neural model. (3) It formulates automated remediation as budgeted risk ranking with trace-grounded remaining work, transparent cost/efficacy assumptions, and a safety state machine. (4) It demonstrates a delayed-feedback mechanism under controlled template-ID churn, using labels only after each streaming batch. Code, checksums, fold-level metrics, policy sensitivity data, and a reference-verification table accompany the paper.

2 RELATED WORK

2.1 Log Parsing and Anomaly Detection

Early work showed that console logs can reveal large-scale system problems when free text is converted into structured event features [4]. Subsequent empirical work established reusable benchmarks and demonstrated that classical methods can remain competitive depending on grouping and window construction [6]. Drain made online template extraction practical with a fixed-depth tree [7], while Loghub consolidated heterogeneous public datasets and benchmark metadata [5]. The broader reliability-engineering survey identifies parsing, anomaly detection, failure prediction, and diagnosis as connected stages rather than independent tasks [8].

Sequence models extend these foundations. DeepLog learns next-event behavior with recurrent networks [10]; LogAnomaly combines sequential and quantitative anomalies [11]; LogRobust uses semantic representations to tolerate unstable logs [12]; PLELog estimates probabilistic labels for semi-supervised learning [13]; and LogBERT applies self-supervised transformer objectives [14]. These methods motivate rich sequence modeling, but the present work deliberately evaluates a compact tree model because the target is a low-latency, calibratable control signal. The design also follows evidence that reported near-perfect HDFS results can be sensitive to data grouping, training selection, and whether the full sequence is visible (Figure 1) [15].

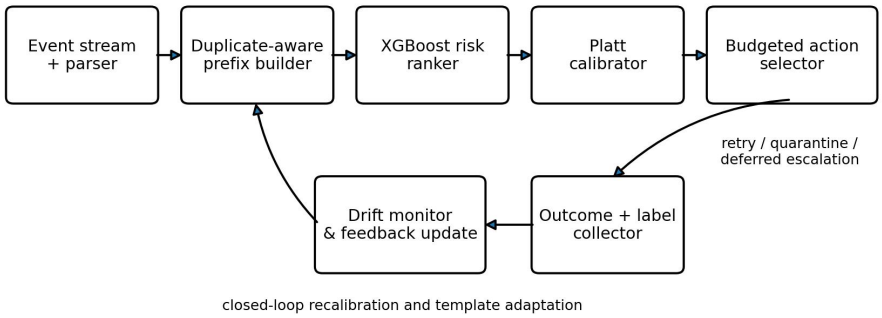

Figure 1 CLARITY Closes the Loop from Event Prefixes to Budgeted Actions and Feedback-Driven Adaptation

2.2 Failure Prediction and AIOps

Cloud failure prediction has used multivariate node telemetry, temporal features, and historical incidents. Lin et al. predicted node failures in a production cloud service system and highlighted the importance of advance warning [16]. AIOps research places prediction within a wider lifecycle of data quality, diagnosis, human trust, and operational integration [17]. Large-scale cluster managers and workload studies demonstrate why a predictor must coexist with resource policies rather than operate in isolation [1-2, 18-19]. Seer similarly connects telemetry analysis with performance diagnosis in microservices [20]. CLARITY differs by evaluating the prediction-to-action connection directly under a bounded budget.

2.3 Calibration and Production Feedback

Tree ensembles are strong tabular rankers, but their outputs are not guaranteed to be calibrated probabilities [21]. Post-hoc calibration methods evaluate whether predicted probabilities match empirical frequencies [22-23]. This property is operationally important when an action is taken only if expected avoided work exceeds cost. Precision–recall analysis is emphasized because anomalies are rare [24]. Production ML literature also warns that feedback loops, data dependencies, and missing monitoring create hidden technical debt [25], while readiness rubrics call for explicit tests and monitoring [26]. CLARITY operationalizes these ideas through separated calibration data, drift measurements, checksum-controlled inputs, and post-batch feedback.

3 PROBLEM FORMULATION AND FRAMEWORK

3.1 Event-Prefix Failure Prediction

A block execution is represented by an event-template sequence $s_i=(e_{i1},\ldots,e_{iL_i})$, a binary failure label y_i, and an occurrence multiplicity w_i after exact duplicate grouping. At decision horizon K, only $s_{i,1:K}$ is visible and records with $L_i \leq K$ are excluded because no future work remains to prevent. The predictor produces a score $z_i=f_\varphi(x_{i,K})$, where $x_{i,K}$ contains normalized unigram counts and normalized ordered bigram counts. With 29 original event IDs, the deployed representation contains 870 sparse features. Five regularity statistics and a one-hot last event are retained as a 34-feature auxiliary ablation, but are excluded from the deployed ranker after they reduce occurrence-weighted PR-AUC. The auxiliary statistics are unique-event fraction, normalized entropy, maximum run fraction, event-change rate, and repeat fraction.

The ranking model is XGBoost, trained with $\log(1+w_i)$ multiplicity weights and class-balanced effective weight [21]. Log weighting prevents one common exact pattern from fully determining the fitted model, while raw multiplicity

remains available for operational evaluation. A logistic-regression baseline uses the same features and weights. The raw tree score is transformed by Platt scaling fitted on a disjoint calibration fold:

$$p_i = \sigma(a \cdot \text{logit}(z_i) + b). \quad (1)$$

The transform is monotone, so CLARITY and the uncalibrated tree have identical rank metrics within a deployment fold. Their Brier and calibration errors can differ substantially, which separates "who should be acted on first" from "how much benefit should be expected."

3.2 Duplicate-Aware Evaluation

The HDFS corpus contains many repeated control-flow traces. Randomly splitting occurrences can place identical completed sequences in training and testing, obscuring whether a model generalizes beyond memorized paths. CLARITY therefore groups exact full sequences before any fold assignment. If a pattern appears with both labels, all its occurrences are removed and listed. Remaining patterns are greedily assigned to five folds while separately balancing normal and anomalous multiplicity. For each rotation, three folds train the model, the next fold calibrates probabilities and selects the F1 threshold, and the final fold is untouched testing. Results are reported both per unique pattern and with raw occurrence weights (Table 1).

Table 1 Data Audit before Modeling

Item	Value
Raw block sequences	575,061
Normal / anomalous	558,223 / 16,838
Exact full-sequence patterns	18,373
Contradictory patterns	10 (46 occurrences)
Pure patterns retained	18,363
Duplicate occurrence fraction	96.81%
Distinct event IDs	29

3.3 Budgeted Remediation

Let $r_i=\max(L_i-K,0)$ denote observed remaining events after the decision. A deployment receives an occurrence budget B and selects the highest-risk sessions until $\Sigma_i w_i a_i \le B$, where $a_i \in [0,1]$ permits fractional allocation only at a duplicate-pattern boundary. This evaluation corresponds to occurrence-level actions even though fitting is pattern-grouped. Three action families are intentionally abstract: retry a failed operation, quarantine a suspect worker or block route, and defer low-confidence cases to human escalation. The framework does not infer which concrete action is safe from HDFS logs alone.

For a counterfactual policy evaluation, q is action efficacy and c is action cost measured in event-equivalent work. Captured anomalous work is $\Sigma_{i:y_i=1} w_i a_i r_i$. Normalized net savings are

$$S(B,q,c) = [q \Sigma(y_i=1) w_i a_i r_i - c \Sigma w_i a_i] / \Sigma(y_i=1) w_i r_i. \quad (2)$$

Equation (2) is not a production recovery measurement. It is a trace-driven sensitivity model: labels and remaining lengths are observed, whereas q and c are declared assumptions. The reference setting $q=0.8$ and $c=3$ is accompanied by a grid over $q \in \{0.5,0.7,0.8,0.9\}$ and $c \in \{1,3,6\}$.

3.4 Feedback and Drift Handling

The feedback path records the selected action, eventual label, remaining work, and parser/template version. A lightweight logistic model is updated only after each labeled batch. To isolate vocabulary drift from unrelated workload changes, the stress test performs a deterministic unseen renaming of all 29 event IDs after batch 3, mapping them into IDs 30–58. Pre-drift behavior is untouched. A static model and a feedback model see the same batches; both predict before labels arrive, and only the feedback model receives four incremental passes over the completed batch. The experiment measures adaptation to template identity churn, not arbitrary semantic drift.

3.5 Safe Remediation State Machine

A calibrated probability is necessary but insufficient for write access to infrastructure. For an action a, CLARITY computes the declared expected gain $g_{i,a}=q_a p_i r_i - c_a$ and then applies four gates: $g_{i,a}$ must be positive under the configured assumptions; the action must be allowed for the resource scope and current health state; no cooldown, dependency freeze, or incident-wide circuit breaker may be active; and the action-rate budget must remain available. Low-blast-radius retries can be eligible for automatic execution, while quarantine or route changes can require canarying or human approval. The policy specification deliberately separates model evidence from authorization.

Every decision emits an immutable record containing the prefix hash, model and parser versions, calibrated probability, candidate action, gate decisions, budget consumption, eventual outcome, and rollback status. Negative or ambiguous outcomes return to the feedback store rather than being hidden as "operator overrides." This state-machine design follows production-readiness principles and makes it possible to disable automation without disabling prediction (Figure 2) [25-26].

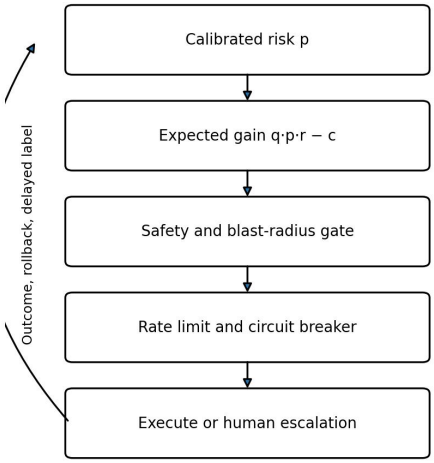

Figure 2 Safety State Machine between Calibrated Risk and an Authorized Remediation Action

4 EXPERIMENTAL METHODOLOGY

4.1 Dataset and Reproducibility

Experiments use the labeled HDFS_v1 block sequences associated with Loghub [4-5]. The original HDFS logs were generated in a 203-node private-cloud environment and labeled by block identifier. The reproducibility package downloads the AIT-AECID preprocessing files containing 5,583 training-normal sequences, 552,640 test-normal sequences, and 16,838 anomalous sequences. SHA-256 checksums are enforced. The split names are not reused as experimental partitions; all sequences are recombined, audited, grouped, and re-split under the protocol above. This avoids treating the small upstream training-normal subset as a representative supervised split.

After removing 46 contradictory occurrences, 575,015 occurrences remain. For K=8, 568,824 occurrences are eligible, including 10,623 anomalies; for K=12, the counts are 568,822 and 10,621; for K=16, 466,689 and 10,380. The decreasing population at K=16 reflects completed short normal sessions, which changes the operational prevalence and is one reason results are reported separately by horizon.

4.2 Models and Metrics

Logistic regression and XGBoost are trained on identical sparse features. XGBoost uses 320 trees, maximum depth four, learning rate 0.04, 0.85 row and feature subsampling, and regularization; all values and the random seed are in the script. CLARITY is the same ranker followed by fold-specific Platt calibration. No test fold is used to select model parameters, calibration coefficients, or thresholds.

The primary metric is occurrence-weighted area under the precision–recall curve (PR-AUC), because the eligible anomaly rate is approximately 1.87% at K=8 and K=12 and 2.22% at K=16 [24]. ROC-AUC, weighted F1, Brier score, and ten-bin expected calibration error (ECE) are secondary. Pattern-level PR-AUC is reported to expose performance across unique traces rather than frequency-dominant paths. All tables show mean and standard deviation over five held-out folds.

4.3 Research Questions

RQ1 asks how early prefix length affects failure ranking. RQ2 asks whether disjoint post-hoc calibration improves probability quality without changing rank. RQ3 evaluates how much anomalous remaining work a bounded policy captures and when action cost overwhelms benefit. RQ4 measures whether delayed feedback recovers from controlled template-ID churn.

4.4 Implementation and Computational Properties

Prefix extraction is linear in the visible prefix: at most K unigram updates and K−1 transition updates are required. The deployed global feature space contains 870 dimensions, while the ablation space contains 904; a record contains no more than O(K) nonzero event and transition terms plus, when enabled, the fixed auxiliary terms. XGBoost inference is bounded by the configured 320 trees of depth four, and Platt calibration adds one affine transform and one sigmoid. These choices avoid an unbounded history buffer or an online deep-model fine-tuning path in the action loop.

The offline audit hashes complete integer sequences to aggregate multiplicity, labels, and lengths; this grouping is performed before folds are constructed and does not delay online decisions. The package records data hashes, random seed, dependency versions, fold statistics, per-fold metrics, and all policy grid points. The end-to-end script can download the three source files, verify them, rebuild the results, and regenerate figures. The full public dataset is not duplicated in the archive.

5 RESULTS

5.1 Dataset Forensics

Exact duplication is the dominant structural property of the corpus: 96.81% of all occurrences belong to a sequence pattern observed more than once. The single most frequent completed trace accounts for 16.49% of occurrences; the five most frequent account for 42.81%, the ten most frequent for 54.83%, and the top 100 for 77.31%. Expressed from the pattern side, the top 1% of unique patterns cover 82.10% of occurrences and the top 5% cover 91.89%. Therefore, an occurrence-random split can repeatedly expose the same completed behavior to both training and testing even though the corpus contains more than half a million rows.

Multiplicity is not merely a nuisance: common normal paths are operationally important, so occurrence-weighted metrics remain useful. The audit instead prevents identity leakage and presents a second pattern-level view. Occurrence-weighted median sequence length is 19 events, whereas the median across unique patterns is 26, showing that repeated paths tend to be shorter than the pattern population. Ten exact patterns have contradictory labels across 46 occurrences. They are excluded and disclosed rather than resolved by majority vote, because either choice would inject an unverified label assumption (Figure 3).

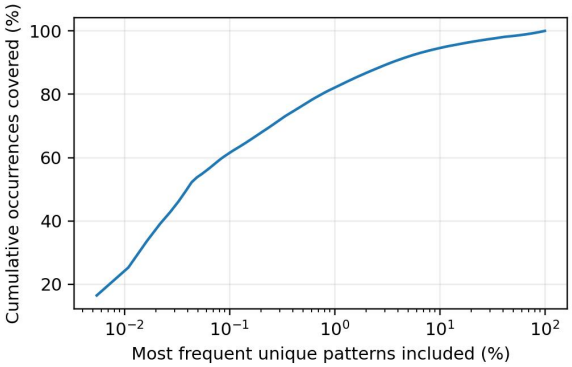

Figure 3 Cumulative Occurrence Coverage by the Most Frequent Exact Full-Sequence Patterns

5.2 RQ1: Early Prediction

Table 2 CLARITY Performance by Prefix Horizon

K	Occ. PR	Pat. PR	ROC	F1	Brier
8	0.102±0.040	0.434±0.020	0.507±0.075	0.125±0.032	0.0178±0.0004
12	0.136±0.026	0.523±0.006	0.545±0.107	0.176±0.051	0.0172±0.0005
16	0.433±0.066	0.751±0.139	0.666±0.085	0.406±0.219	0.0193±0.0006

Table 2 and Figure 4 show a clear warning-time trade-off. Eight events provide little occurrence-weighted discrimination, although performance across unique patterns is already above the corresponding prevalence. At K=12, pattern PR-AUC rises to 0.523 but operational PR-AUC remains 0.136. At K=16, occurrence-weighted PR-AUC reaches 0.433 and pattern PR-AUC reaches 0.751. The result does not imply that sixteen events are universally optimal; it is the strongest of the three tested horizons on this corpus, and it leaves a median of twelve future events among anomalous unique patterns. A production system should re-estimate this horizon against its own action latency.

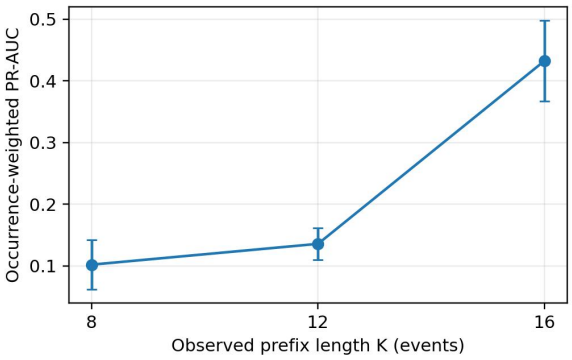

Figure 4 Occurrence-Weighted PR-AUC Rises as More Prefix Evidence Becomes Available; Error Bars Are One Standard Deviation over Folds

5.3 RQ2: Ranking Versus Calibration

Table 3 K=16 Occurrence-Weighted Model Comparison

Model	PR-AUC	F1	Brier	ECE
Logistic	0.358±0.103	0.330±0.231	0.367±0.083	0.571±0.077
XGBoost	0.433±0.066	0.409±0.222	0.467±0.122	0.636±0.102
CLARITY	0.433±0.066	0.406±0.219	0.019±0.001	0.022±0.007

XGBoost improves PR-AUC from 0.358 for logistic regression to 0.433 (Table 3), but class-balanced training makes its raw probabilities unsuitable for direct cost comparison: mean Brier score is 0.467 and ECE is 0.636. Platt scaling reduces these values to 0.019 and 0.022 while preserving PR-AUC. The nearly unchanged F1 reflects threshold selection on calibration folds; small differences arise from finite candidate thresholds. Figure 5 visualizes the separation between ranking and probability quality. This result supports calibration as an operational requirement rather than a cosmetic post-processing step [22-23].

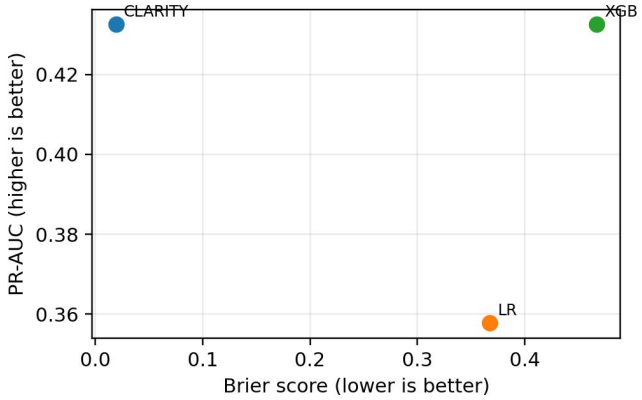

Figure 5 K=16 Ranking–Calibration Trade-Off
Note: CLARITY retains XGBoost ranking while correcting probability scale.

5.4 Feature Ablation and Parsimony

Table 4 K=16 Duplicate-Aware Feature Ablation

Feature set	Occ. PR-AUC	Pat. PR-AUC
Event counts	0.334±0.047	0.640±0.201
Ordered transitions	0.399±0.078	0.739±0.146
Counts + transitions	0.433±0.066	0.751±0.139
+ regularity + last event	0.411±0.060	0.751±0.138

The ablation provides a useful negative result (Table 4). Event counts alone obtain occurrence-weighted PR-AUC 0.334±0.047, while ordered transitions reach 0.399±0.078. Combining both raises the score to 0.433±0.066 and pattern PR-AUC to 0.751±0.139. Adding the five regularity statistics and one-hot last event leaves pattern PR-AUC essentially unchanged (0.751±0.138) but lowers occurrence-weighted PR-AUC to 0.411±0.060. CLARITY therefore deploys the simpler 870-dimensional counts-plus-transitions representation rather than retaining auxiliary features merely because they are available.

The discrepancy between pattern and occurrence views indicates that the auxiliary terms do not broadly destroy unique-pattern ranking; instead, they perturb ordering among high-multiplicity paths that dominate operational impact. This observation reinforces the duplicate-aware protocol: a feature set selected only on unique-pattern averages would appear tied, whereas the occurrence view identifies a practically relevant regression. The ablation was run with the same five folds, XGBoost configuration, and test isolation as the main comparison.

5.5 RQ3: Budgeted Remediation

Table 5 CLARITY Policy at K=16 (q=0.8, c=3 Events)

Budget	Precision	Recall	Work capture	Net savings
1%	75.69%	34.03%	37.39%	+15.76%
2%	56.18%	50.52%	49.26%	+11.10%
5%	23.00%	51.70%	50.74%	-30.16%
10%	11.50%	51.71%	50.75%	-100.91%

At a 1% budget, CLARITY acts on approximately 4,667 occurrences (Table 5), with 75.69% precision and 37.39% capture of anomalous remaining work. The 2% budget captures 49.26% of remaining anomalous work and 50.52% of anomalous occurrences. The corresponding normalized net savings are 15.76% and 11.10% under the reference assumptions. Increasing the budget to 5% or 10% adds low-risk sessions but does not increase anomaly recall in the

out-of-fold ranking; action cost then dominates and net savings become negative. This plateau is an important negative result: automated remediation should not be interpreted as "act on every score above a permissive threshold."
The random baseline captures only the budget fraction of remaining anomalous work in expectation and is negative at every tested budget. XGBoost and CLARITY have identical within-fold selections because calibration is monotone; the operational value of calibration is that $q \cdot p_i \cdot r_i$ can be compared with action cost or used to select different actions. Figure 6 shows remaining-work capture, and the supplied sensitivity file preserves all q and c combinations rather than presenting only the favorable setting.

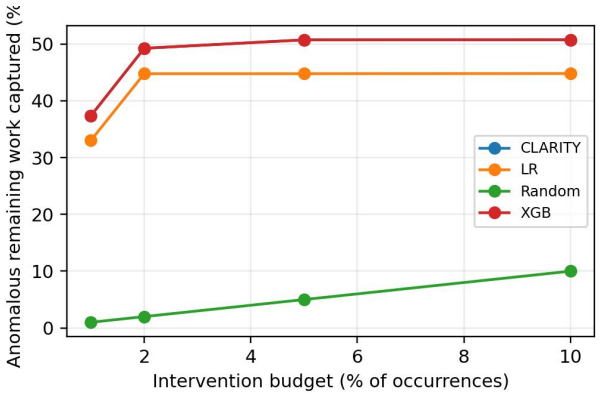

Figure 6 Captured Anomalous Remaining Work versus Occurrence Budget. Policy Evaluation Is Performed within Each Held-Out Fold and Then Aggregated

5.6 Cost and Efficacy Sensitivity

The reference policy is not universally profitable. At a 2% action budget and unit event cost, normalized net savings remain positive for every tested efficacy, ranging from 15.19% at q=0.5 to 34.90% at q=0.9. At cost c=3, the same budget is negative at q=0.5 (−3.67%) but becomes positive at q=0.7, 0.8, and 0.9 (6.18%, 11.10%, and 16.03%). At c=6, all 2% settings are negative. The safe operating region is therefore a joint property of ranking concentration, action efficacy, action cost, and budget—not a fixed probability threshold.
A smaller budget can remain viable under higher costs because it concentrates on the top-ranked sessions. At a 1% budget and c=6, net savings are −9.61% at q=0.5, −2.13% at q=0.7, +1.61% at q=0.8, and +5.35% at q=0.9. This sensitivity motivates conservative deployment: the action gate should use a validated lower bound on efficacy or require human approval when the expected gain is close to zero. Figure 7 visualizes the complete 2% slice; every numerical grid point is included in the result archive.

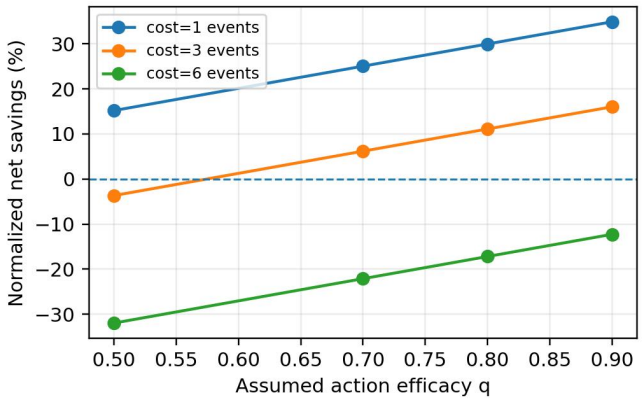

Figure 7 Normalized Net Savings at a 2% Action Budget across Declared Action Efficacy and Event-Equivalent Cost

5.7 RQ4: Feedback Under Template Churn

Before drift, the static and feedback models have similar mean pattern PR-AUC (0.674 and 0.671, respectively). Immediately after all event IDs are renamed, both models degrade; the feedback model is not protected from the first unseen batch because its labels have not arrived. Across batches 4–9, however, feedback achieves mean PR-AUC 0.470 versus 0.190 for the static model and mean Brier 0.182 versus 0.280. By the first post-label batch, feedback PR-AUC rises from 0.191 to 0.518, reaching 0.589 in the final batch. Figure 8 therefore measures recovery, not clairvoyant drift avoidance.

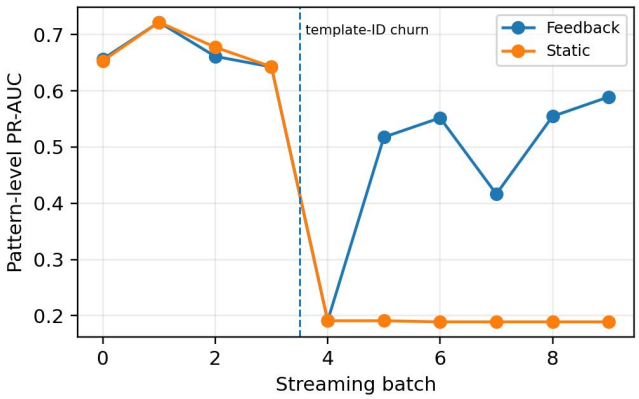

Figure 8 Static and Feedback Models under Controlled Event-Template ID Renaming after Batch 3. Labels Are Used Only after Each Batch

5.8 Interpretation

Three findings matter more than a single headline score. First, duplicate frequency is an operational fact but should not be allowed to leak identical full traces across partitions. Reporting both pattern and occurrence views distinguishes broad generalization from frequency-weighted impact. Second, calibration and ranking solve different problems. A model may prioritize useful sessions yet produce probabilities that are unusable for cost-sensitive action. Third, remediation value is non-monotonic in action volume. The strongest tested policy is deliberately selective; its value disappears when intervention volume exceeds the concentration of detectable failures.

The closed-loop contribution is therefore methodological rather than a claim of autonomous repair. Measured sequence labels support prediction and remaining-work capture; declared q and c support transparent counterfactual analysis; controlled template renaming supports a targeted stress test. Keeping these evidence classes separate is essential for trustworthy AIOps because a realistic-looking utility number can otherwise be mistaken for a live-cluster result.

6 DEPLOYMENT GUIDANCE

6.1 Staged Activation

A practical rollout should begin in shadow mode, where predictions, candidate actions, gate decisions, and later outcomes are logged but no writes occur. Advisory mode can then expose ranked cases to operators and measure agreement, false-positive cost, label delay, and rollback needs. Restricted automation should start with reversible, low-blast-radius actions under canary scope and a small budget. Quarantine, traffic routing, or resource eviction should retain explicit approval until action-specific causal evidence is available.

6.2 Monitoring and Failure Containment

The predictor should be monitored by parser/model version using PR-AUC, prevalence, Brier score, and ECE; the policy should be monitored using action precision, consumed budget, estimated and observed avoided work, rollback rate, repeated-action rate, and time-to-label. A circuit breaker should trip on calibration drift, label unavailability, abnormal rollback frequency, or broad dependency incidents. Fail-open and fail-closed behavior must be action-specific: suppressing a retry may be safer than executing an uncertain quarantine, while some isolation actions may be mandatory during known integrity failures.

6.3 Validation Beyond This Study

Production validation requires outcomes unavailable in the public corpus: concrete action identity, service-level latency and availability, collateral impact, operator intervention, and counterfactual controls. Suitable designs include phased canaries, matched historical controls, or randomized trials where ethically and operationally acceptable. The policy should be re-estimated separately for each action because efficacy and cost are not transferable. This paper supplies the prediction and auditing substrate, not a substitute for action-level causal validation.

6.4 Operational Invariants

Automation should be governed by invariants that are checked independently of model score. A decision is ineligible when the parser or feature schema is unknown, the calibration monitor is outside its accepted range, the resource is already in a conflicting workflow, the action is not idempotent, or no tested rollback path exists. Budgets should be scoped by action, service, failure domain, and time window so that a globally small percentage cannot concentrate on one rack or dependency. Every action should carry a deduplication key and an expiry time to prevent repeated retries from becoming a new failure amplifier.

Label delay must also be treated as an operating state. When outcomes stop arriving, feedback updates should pause and automation should fall back to shadow or advisory mode; otherwise the system can continue acting while its measured efficacy becomes stale. Model, calibrator, parser, policy, and gate versions should be changed atomically or evaluated as an explicitly versioned combination. These invariants are framework requirements rather than measured HDFS results, but they define the conditions under which the closed loop can fail safely instead of silently drifting into unbounded control.

6.5 Reproducibility Contract

The artifact separates evidence by file and by generation path. Dataset counts and conflicts are written to dataset_audit.json and label_conflicts.csv; duplicate-aware fold composition is preserved in fold_statistics.csv; fold-level and aggregated predictive metrics are stored separately; feature_ablation.csv records every fold of Table 5; remediation_budget.csv and remediation_sensitivity.csv contain observed ranking selections together with declared q and c; and drift_feedback.csv is paired with the deterministic ID mapping used by the controlled stress test. The manifest states the random seed and the evidentiary status of policy and drift outputs.

Reproduction begins from the three public sequence files rather than bundled data. The script downloads them when absent, rejects checksum mismatches, recombines the upstream splits, removes only disclosed contradictory patterns, and regenerates all primary CSV results and plots. A reference-verification table records the bibliographic fields and verification source for every cited item. This contract cannot eliminate library- or platform-level floating-point variation, but it prevents silent input substitution and makes each reported number traceable to a result row or a declared counterfactual assumption.

7 THREATS TO VALIDITY AND LIMITATIONS

External validity is limited by one public HDFS corpus. Block-level storage logs do not cover host failures, network partitions, multi-service causal graphs, or modern container orchestration. The event sequences are pre-parsed; raw-text parser errors are not measured. Exact sequence grouping blocks the most direct duplicate leakage, but different full sequences may share the same K-event prefix, which is realistic for online prediction and can still make early prefixes ambiguous.

The remediation study is counterfactual. It uses real labels and real remaining sequence lengths, but it assumes event-equivalent action cost and efficacy rather than observing retries or quarantines in a live cluster. Consequently, the reported 15.76% and 11.10% net savings are conditional policy-model outputs, not availability, latency, or financial savings. A production deployment would require action-specific safety constraints, causal or randomized evaluation, rollback rules, and human approval for high-blast-radius actions.

The drift experiment is controlled. Renaming every template ID is severe and isolates parser identity churn, but it does not represent gradual semantic drift, correlated infrastructure upgrades, or adversarial behavior. Incremental feedback assumes labels arrive after each batch and are sufficiently reliable. Finally, five folds quantify split variability but do not establish superiority over every deep architecture. The objective is a reproducible closed-loop protocol, not a state-of-the-art claim.

8 CONCLUSION

CLARITY turns failure prediction into a bounded operational loop: audit and group duplicated traces, predict from an early prefix, calibrate risk, allocate a finite remediation budget, apply safety gates, and learn from delayed outcomes. On 575,061 real HDFS sequences, the framework exposes substantial duplicate structure, obtains its strongest tested early ranking after sixteen events, and reduces probability error by more than an order of magnitude through disjoint calibration. The trace-driven policy shows that selective action can capture 49.26% of anomalous remaining work at a 2% budget, while larger budgets can destroy net value. A controlled drift test further demonstrates that feedback materially improves recovery after template-ID churn. The key outcome is not that one model “solves” cloud failures, but that measured prediction, counterfactual action value, controlled stress testing, and operational safeguards can be combined without conflating their evidentiary status.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] Verma A, Pedrosa L, Korupolu M R, et al. Large-scale cluster management at Google with Borg. Proceedings of ACM EuroSys, 2015: 18, 1-17. DOI: 10.1145/2741948.2741964.

[2] Cortez E, Bonde A, Muzio A, et al. Resource Central: Understanding and predicting workloads for improved resource management in large cloud platforms. Proceedings of ACM SOSP, 2017: 153-167. DOI: 10.1145/3132747.3132772.

[3] Dean J, Barroso L A. The tail at scale. Communications of the ACM, 2013, 56(2): 74-80. DOI: 10.1145/2408776.2408794.
[4] Xu W, Huang L, Fox A, et al. Detecting large-scale system problems by mining console logs. Proceedings of ACM SOSP, 2009: 117-132. DOI: 10.1145/1629575.1629587.
[5] Zhu J, He S, He P, et al. Loghub: A large collection of system log datasets for AI-driven log analytics. Proceedings of IEEE ISSRE, 2023: 355-366. DOI: 10.1109/ISSRE59848.2023.00071.
[6] He S, Zhu J, He P, et al. Experience report: System log analysis for anomaly detection. Proceedings of IEEE ISSRE, 2016: 207-218. DOI: 10.1109/ISSRE.2016.21.
[7] He P, Zhu J, Zheng Z, et al. Drain: An online log parsing approach with fixed depth tree. Proceedings of IEEE ICWS, 2017: 33-40. DOI: 10.1109/ICWS.2017.13.
[8] He S, He P, Chen Z, et al. A survey on automated log analysis for reliability engineering. ACM Computing Surveys, 2021, 54(6): 130. DOI: 10.1145/3460345.
[9] Oliner A J, Ganapathi A, Xu W. Advances and challenges in log analysis. Communications of the ACM, 2012, 55(2): 55-61. DOI: 10.1145/2076450.2076466.
[10] Du M, Li F, Zheng G, et al. DeepLog: Anomaly detection and diagnosis from system logs through deep learning. Proceedings of ACM CCS, 2017: 1285-1298. DOI: 10.1145/3133956.3134015.
[11] Meng W. LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. Proceedings of IJCAI, 2019: 4739-4745. DOI: 10.24963/ijcai.2019/658.
[12] Zhang X. Robust log-based anomaly detection on unstable log data. Proceedings of ACM ESEC/FSE, 2019: 807-817. DOI: 10.1145/3338906.3338931.
[13] Yang L, Chen J, Wang Z, et al. PLELog: Semi-supervised log-based anomaly detection via probabilistic label estimation. Proceedings of IEEE/ACM ICSE Companion, 2021: 230-231. DOI: 10.1109/ICSE-Companion52605.2021.00106.
[14] Guo H, Yuan S, Wu X. LogBERT: Log anomaly detection via BERT. Proceedings of IJCNN, 2021: 1-8. DOI: 10.1109/IJCNN52387.2021.9534113.
[15] Le V H, Zhang H. Log-based anomaly detection with deep learning: How far are we? Proceedings of ACM/IEEE ICSE, 2022: 1356-1367. DOI: 10.1145/3510003.3510155.
[16] Lin Q. Predicting node failure in cloud service systems. Proceedings of ACM ESEC/FSE, 2018: 480-490. DOI: 10.1145/3236024.3236060.
[17] Dang Y, Lin Q, Huang P. AIOps: Real-world challenges and research innovations. Proceedings of IEEE/ACM ICSE Companion, 2019: 4-5. DOI: 10.1109/ICSE-Companion.2019.00023.
[18] Cheng Y. Analyzing Alibaba's co-located datacenter workloads. Proceedings of IEEE Big Data, 2018: 292-297. DOI: 10.1109/BigData.2018.8622518.
[19] Reiss C, Wilkes J, Hellerstein J L. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. Proceedings of ACM SoCC, 2012: 7, 1-13. DOI: 10.1145/2391229.2391236.
[20] Gan Y. Seer: Leveraging big data to navigate the complexity of performance debugging in cloud microservices. Proceedings of ACM ASPLOS, 2019: 19-33. DOI: 10.1145/3293882.3330558.
[21] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proceedings of ACM KDD, 2016: 785-794. DOI: 10.1145/2939672.2939785.
[22] Guo C, Pleiss G, Sun Y, et al. On calibration of modern neural networks. Proceedings of ICML, PMLR, 2017, 70: 1321-1330.
[23] Niculescu-Mizil A, Caruana R. Predicting good probabilities with supervised learning. Proceedings of ICML, 2005: 625-632. DOI: 10.1145/1102351.1102430.
[24] Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One, 2015, 10(3): e0118432. DOI: 10.1371/journal.pone.0118432.
[25] Sculley D. Hidden technical debt in machine learning systems. Advances in Neural Information Processing Systems, 2015: 2503-2511.
[26] Breck E, Cai S, Nielsen E, et al. The ML test score: A rubric for ML production readiness and technical debt reduction. Proceedings of IEEE Big Data, 2017: 1123-1132. DOI: 10.1109/BigData.2017.8258038.