Software Engineering and Artificial Intelligence Systems

Volume: 02 Issue: 01 (2026)

www.seais.org/

CADRE-FP: Adaptive Failure Prediction for Cloud Infrastructure under Log Concept Drift

Zixuan Luo[1], Ji-won Kim[2], Edward Clarke[3], Ruihan Zhou[1]*

[1] College of Computer Science & Software Engineering, Shenzhen University, Shenzhen, China

[2] Department of Computer Science, Hanyang University, Seoul, Republic of Korea

[3] School of Computer Science and Electronic Engineering, University of Surrey, Guildford, UK

*Co-corresponding authors: Ruihan Zhou, 839201032@qq.com

Key word: cloud infrastructure, failure prediction, log concept drift, online learning, log anomaly detection, AIOps, HDFS, concept drift.

A B S T R A C T

Abstract—Failure predictors built from system logs degrade when software evolution or parser changes alter event identities, yet indiscriminate online retraining can also erase useful stationary knowledge. This paper presents CADRE-FP, a leakage-controlled streaming framework that combines an event-identity expert, a rename-invariant canonical topology expert, stable/plastic model memories, and an unlabeled novelty–Jensen-Shannon drift controller. The canonical representation renumbers events by order of first appearance within each observed prefix, making it exactly invariant to bijective template-ID renaming while preserving local control-flow topology. We evaluate on 575,061 labeled HDFS_v1 block sequences. Exact full-sequence duplicates are grouped before splitting, ten contradictory patterns (46 occurrences) are removed, and prediction uses only the first 16 events. All labels and base sequences are real; concept-drift scenarios are deterministic, label-preserving transformations of held-out event IDs rather than claims of naturally observed temporal drift. Across five pattern-disjoint repeated splits, CADRE-FP reaches post-drift pattern PR-AUC 0.599±0.011 under complete template renaming, versus 0.541±0.008 for continuous online learning and 0.214±0.025 for a static identity model. Under gradual template splitting it matches online adaptation at 0.631±0.012 and exceeds the static model's 0.580±0.013, while the no-drift control produces zero drift activations in 60 batch evaluations and preserves the static pattern PR-AUC of 0.671±0.010. A class-conditional fault-signature drift exposes a limitation: structural drift signals are weak and occurrence-weighted PR-AUC trails continuous online learning. The results support representation-level invariance plus selective plasticity as a practical direction for adaptive cloud failure prediction, while clearly separating measured evidence from controlled stress assumptions

INTRODUCTION

Large cloud platforms expose operators to failures that emerge from complex interactions among software versions, nodes, workload phases, and shared dependencies. Logs remain a uniquely detailed source of control-flow and component-state evidence, and they have supported both anomaly detection and failure prediction research for more than a decade [2], [6], [7], [21]. In an AIOps setting, however, a predictor is useful only if it continues to rank risky executions after the logging vocabulary and execution patterns evolve [22], [25]. The problem is therefore not merely to fit a high-accuracy classifier on a static benchmark, but to preserve predictive competence under distribution change without creating a new instability through over-adaptation. The HDFS benchmark illustrates both the value and the difficulty of this problem. It provides labeled block-level traces and has become a standard test bed through Loghub [2]–[4]. At the same time, benchmark construction can dominate apparent performance: duplicates, grouping choices, imbalance, parser artifacts, and whether a model sees a complete execution can materially alter conclusions [13], [14]. A particularly valuable methodological precedent is CLARITY, which couples early prefix prediction with duplicate-aware evaluation and delayed feedback rather than treating a failure predictor as an isolated static classifier [1]. That operational discipline motivates our evaluation protocol, but the present work focuses on a different question: how should the predictor itself remain useful when log identities drift? Existing log models address instability from different directions. DeepLog and LogAnomaly model event sequences [8], [9]; LogRobust explicitly targets unstable logs [10]; probabilistic labeling and self-supervised transformers improve robustness under limited labels or richer semantics [11], [12]. Recent work goes further toward software evolution: EvLog studies anomalous logs across evolving versions [15], while METER dynamically adapts online anomaly detectors to concept changes [16]. General streaming research offers drift detectors and adaptive ensembles [17]–[20]. Yet an important gap remains for early, prefix-based failure prediction: a parser can rename event identifiers without changing the underlying execution topology, and an adaptive learner that reacts only by retraining on the new identifiers wastes the structural invariance already present in the sequence. CADRE-FP addresses this gap with a deliberately compact design. It maintains two views of the same prefix. The identity view retains failure-specific template information through hashed unigram, ordered-bigram, and position features. The canonical view discards absolute event names and renumbers distinct events by first appearance, preserving within-prefix topology. A stable identity model protects stationary knowledge; a plastic identity model receives delayed-label updates; and a canonical expert acts as a high-severity fallback. An unlabeled controller computes event novelty and Jensen-Shannon divergence before labels arrive, then chooses how much canonical evidence to mix into the current risk score. Labels from a batch are used only after all predictions for that batch are recorded.

The paper makes four contributions:

- A rename-invariant canonical log-prefix representation with a simple invariance argument: any bijective renaming of event IDs leaves the canonical sequence unchanged.
- A stable–plastic adaptation mechanism that separates routine online learning from severe syntactic drift and allows consolidation only after the stream returns to a low-drift state.
- A duplicate-aware, leakage-controlled evaluation on real HDFS_v1 labels with three deterministic drift stressors—complete renaming, gradual template splitting, and class-conditional fault-signature change—plus a no-drift control.
- An evidence-preserving result set that reports both successful adaptation and a negative boundary case, showing where unlabeled structural drift signals are insufficient.

II. RELATED WORK

A. Log-based anomaly detection and failure prediction

Early system-log research showed that console traces can expose large-scale failures after raw messages are converted into structured event representations [2]. Empirical studies later

demonstrated that classical statistical methods can remain competitive depending on grouping and feature construction [4], and Drain made online parsing practical through a fixed-depth tree [5]. The broader log-analysis literature now spans parsing, anomaly detection, failure prediction, and diagnosis [7]. These stages are related but not interchangeable: anomaly detection asks whether an observed sequence deviates from expected behavior, while early failure prediction asks whether a still-running execution is likely to end abnormally and therefore must operate on a prefix rather than the completed trace. Sequence learning has substantially advanced log analytics. DeepLog predicts normal next-event behavior using recurrent networks [8], LogAnomaly combines sequential and quantitative signals [9], and LogBERT uses self-supervised transformer objectives [12]. PLELog addresses scarce labels using probabilistic label estimation [11]. LogRobust is particularly relevant to evolving systems because it uses semantic representations to reduce sensitivity to unstable logs [10]. Nevertheless, empirical evaluations show that high benchmark scores can be fragile with respect to training selection, class distribution, grouping, and early-detection protocol [13]. The dataset review by Landauer et al. similarly warns that anomaly manifestations in common log benchmarks do not always require the sequence complexity assumed by a detector [14]. These findings motivate our emphasis on pattern-disjoint splits and an explicit distinction between syntactic drift and label-changing claims. Cloud failure prediction has also been studied directly from operational telemetry. MING combines temporal and spatial information to rank failure-prone cloud nodes and demonstrates the operational value of advance warning [21]. Seer connects telemetry analytics with microservice performance debugging [25]. Our scope is narrower in input modality but stricter in streaming causality: only a fixed log prefix is observed at decision time, and any labels used for adaptation arrive after a batch has been scored.

B. Software evolution and concept drift

Software evolution can alter logging statements, templates, parsers, and sequence structure. EvLog explicitly studies anomalous-log identification across software versions and highlights parsing errors, evolving events, and unstable sequences as distinct challenges [15]. METER treats online anomaly detection as a dynamic concept-adaptation problem, combining a base model with a drift-aware adaptation mechanism [16]. CADRE-FP is complementary: instead of relying only on model adaptation, it first removes a known nuisance degree of freedom—bijective template naming—from one representation. This matters when a logging or parsing update changes identifiers while leaving control-flow relationships largely intact. Concept drift is well established in streaming learning [17]. DDM monitors changes in predictive error [18], ADWIN maintains a statistically adaptive window [19], and Adaptive Random Forests combine drift-aware base learners for evolving streams [20]. These methods assume different access to labels and different notions of change. In infrastructure telemetry, labels can be delayed and expensive; moreover, some changes are immediately visible in the input vocabulary before failure outcomes are known. CADRE-FP therefore uses an unlabeled controller for syntactic novelty and distribution shift, while delayed predictive evidence only influences subsequent batches.

III. PROBLEM FORMULATION AND CADRE-FP

A. Prefix failure prediction under delayed feedback

Each HDFS block execution is represented as a template-ID sequence $s_i=(e_{i1},...,e_{iL_i})$, a binary outcome y_i in $\{0,1\}$, and an occurrence multiplicity m_i after exact duplicate grouping. At decision horizon $K=16$, the predictor observes only $s_{i,1:K}$. Sequences with $L_i<=K$ are excluded because no later event remains after the decision point. For a streaming batch t, the model outputs $p_{i,t}$ before any labels from t are available. Once the batch completes, y_i may be used to update the plastic model. This prequential order prevents label look-ahead.

We distinguish three drift notions. Syntactic identity drift changes event IDs while approximately preserving within-sequence structure. Structural drift changes the pattern of

event transitions. Conditional failure drift changes the association between features and failure labels. The first can be partly neutralized by an invariant representation; the latter two require learning. CADRE-FP is intentionally strongest for the first case and selectively plastic for the others.

B. Duplicate-aware audit and partitioning

The complete HDFS sequence is used only for offline audit and pattern identity, never as a prediction feature. Exact full-sequence duplicates are grouped before any split. Patterns observed with both labels are removed rather than resolved by majority vote. This follows the reproducibility-minded duplicate discipline demonstrated in the prior closed-loop study [1], while our subsequent train/calibration/stream partitions and drift transformations are independently regenerated. Stratification is performed at the unique-pattern level, producing 50% training, 10% calibration, and 40% streaming partitions in each of five independent seeds. The streaming partition is divided into twelve class-stratified batches.

C. Identity representation

The identity expert preserves information that can be directly associated with particular templates. For the first K events we create normalized event counts u:e, normalized ordered bigram counts b:e_a>e_b, and position tokens p:j:e. FeatureHasher maps these sparse dictionaries to 4,096 nonnegative hashed dimensions. Hashing avoids a feature-schema failure when unseen event IDs appear at deployment time: new identifiers remain representable even if they were absent during training. It does not, however, make the classifier invariant to renaming because the hashed coordinates of renamed tokens differ.

D. Canonical first-appearance representation

For a prefix s, define C(s) by replacing the first distinct event with 0, the next previously unseen event with 1, and so on. For example, (12,12,7,12,9,7) becomes (0,0,1,0,2,1). We then hash canonical position tokens, canonical bigrams, and canonical counts into 2,048 dimensions and append invariant statistics including unique-event fraction, change rate, maximum run fraction, and normalized entropy.

Proposition 1 (rename invariance). Let pi be any bijection over event identifiers. Then C(pi(s))=C(s). Proof: a bijection preserves equality and inequality relations among positions. Therefore the ordered list of first occurrences is unchanged except for the external names assigned to each equivalence class. C assigns codes solely from that first-occurrence order, so every position receives the same canonical code before and after pi. This exact invariance is the reason the canonical expert can score a completely renamed parser namespace without seeing the new IDs during training.

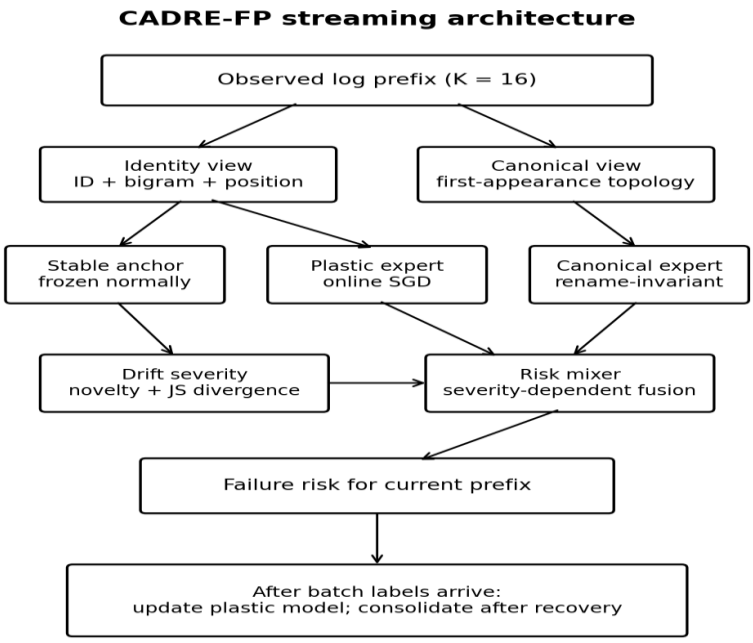


Figure 1CADRE-FP combines identity and canonical views with an unlabeled drift controller and delayed-label adaptation.

E. Stable–plastic experts and drift controller

Two logistic SGD identity models are initialized from the same training fit. The stable anchor is not routinely updated. The plastic model receives one online pass after each labeled batch. A separately trained canonical SGD model uses the canonical representation. All training updates use sample weight log(1+m_i) multiplied by a class-balancing factor derived from the training partition. Log multiplicity retains the operational significance of common paths without allowing a single duplicated trace to dominate optimization. Before labels arrive, the controller summarizes the current batch prefix vocabulary. Let nu_t be the fraction of visible event occurrences whose IDs have not previously appeared, and let J_t be the Jensen-Shannon divergence between the current event-frequency distribution and an exponentially updated reference distribution. The severity score is d_t=max(nu_t,min(1,J_t/0.5)). Adaptation becomes active when d_t>0.05 or when the previous labeled batch provides evidence that the plastic expert exceeded the anchor by more than 0.04 pattern PR-AUC. The latter signal is necessarily one-batch delayed. When active, the canonical mixture weight is lambda_t=clip((d_t-0.20)/0.80,0,1). The current risk is p_t=(1-lambda_t)p_plastic+lambda_t p_canonical; when inactive, the stable anchor is used. Severe drift (d_t>0.20) receives four plastic update passes after labels arrive; lower-severity active batches receive one pass. A three-batch cooldown avoids immediate oscillation. If severity falls below 0.05 and delayed feedback no longer favors the plastic expert, the plastic model is copied into the anchor and adaptation returns to the stable state. The event-frequency reference is updated after scoring with an exponential moving average coefficient of 0.5.

IV. EXPERIMENTAL METHODOLOGY

A. Dataset and measured audit

Experiments use the public HDFS_v1 block sequences associated with Loghub. The dataset documentation states that the logs were generated in a private-cloud HDFS environment, sliced by block identifier, and labeled normal or anomalous [2], [3]. We use the AIT-AECID preprocessed sequence files corresponding to 5,583 training-normal, 552,640 test-normal, and 16,838 anomalous sequences; their upstream split names are ignored after ingestion. SHA-256 checksums are verified before every run, and the exact digests are recorded in the accompanying audit artifact.

Our independent audit reads 575,061 sequences: 558,223 normal and 16,838 anomalous. They collapse to 18,373 exact full-sequence patterns. Ten patterns contain contradictory labels across 46 occurrences and are excluded, leaving 18,363 pure patterns and 575,015 occurrences. With K=16 and the requirement L_i>K, 17,733 unique patterns (466,689 occurrences) are eligible, including 4,027 anomalous patterns and 10,380 anomalous occurrences.

Table 1HDFS_v1 DATA AUDIT USED BY CADRE-FP

Item	Measured value
Raw labeled 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
Eligible at K=16	17,733 patterns / 466,689 occurrences
Eligible anomalous	4,027 patterns / 10,380 occurrences

B. Controlled drift protocol

HDFS_v1 does not provide a trustworthy natural timestamped software-evolution sequence for our block-level prefix protocol, so we do not present the following changes as observed production drift. Instead, every scenario starts from held-out real prefixes and real labels, and only event IDs are transformed deterministically after batch 3. No label is generated or flipped. The goal is a controlled stress test with a known onset at batch 4.

- No drift: held-out prefixes are unchanged. This measures unnecessary adaptation and false drift activation.
- Complete template rename: every event ID is shifted by +100 from batch 4 onward. The transformation is bijective and preserves equality patterns and order exactly. It models a parser/template namespace migration.
- Gradual template split: six frequent training-selected event IDs may be mapped to new IDs (+300). The affected fraction rises deterministically from 10% at batch 4 by seven percentage points per batch, capped at 65%. The mapping decision uses a deterministic hash of the sequence, position, and batch, making the scenario reproducible without random post-hoc choices.
- Fault-signature drift: eight event IDs with the largest anomaly-vs-normal prefix log-odds are selected using training data only. From batch 4, those IDs are shifted by +200 only in anomalous held-out sequences. Ground-truth labels are used solely by the offline stress generator to create a class-conditional feature change; the predictor still receives each label only after scoring its batch. This deliberately challenges a detector based on marginal vocabulary drift.

C. Baselines, thresholding, and metrics

Static-ID is the identity model frozen after training. Online-ID starts from the same model and receives one delayed-label update after every streaming batch regardless of detected drift. CADRE-FP uses the same identity initialization plus the canonical expert and controller described above. The calibration partition is used only to select an F1 threshold for secondary thresholded metrics; it is not used for streaming updates. The primary metric is area under the precision-recall curve because anomalies are rare and PR analysis is more informative than

ROC alone in imbalanced settings [26-31]. We report pattern-level PR-AUC as the primary generalization view and multiplicity-weighted occurrence PR-AUC as an operational-frequency view. Mean and standard deviation are computed over five independent pattern-disjoint splits [32-37].

D. Reproducibility and implementation

The complete experiment is a single Python script using NumPy, pandas, SciPy, and scikit-learn. Feature extraction and drift generation are deterministic given the recorded base seed 20260819. The artifact stores the data audit, contradictory-label patterns, split statistics, batch metrics, aggregate metrics, paired gains, figures, package requirements, and runtime metadata. The five-repeat run that generated this paper completed in 12.9 s in the execution environment used for artifact generation; this number is reported only as a reproducibility observation, not as a hardware-independent latency benchmark. Production readiness requires monitoring, versioning, and rollback discipline beyond an offline predictor [23], [24].

Table 2FIXED EXPERIMENT AND CONTROLLER SETTINGS

Setting	Value
Observed prefix K	16 events
Identity hash space	4,096 dimensions
Canonical hash space	2,048 dimensions
Stream construction	12 class-stratified batches
Split per repeat	50% train / 10% calibration / 40% stream
Drift activation	severity > 0.05
Severe canonical mixing	severity > 0.20
Update policy	4 passes if severe; otherwise 1
Reference EMA coefficient	0.5
Repeated splits	5

For K=16, the identity view requires at most 16 unigram updates, 15 ordered-bigram updates, and 16 position tokens before hashing; the canonical view has the same prefix-order cost plus four scalar topology statistics. Thus online feature construction is O(K), and no full-session history is required in the decision loop.

V. RESULTS

A. RQ1: Does the controller avoid unnecessary adaptation?

Yes in the no-drift control. Across five repeats and twelve batches per repeat, CADRE-FP never enters the active state (0 of 60 batch evaluations). Its post-control pattern PR-AUC is 0.671±0.010, exactly matching Static-ID (0.671±0.010) because the anchor remains untouched. Online-ID reaches 0.661±0.013 pattern PR-AUC; its continuous updates change the ranking even though no synthetic drift is present. On the occurrence-weighted view, Online-ID is higher (0.386±0.081) than CADRE-FP/Static-ID (0.336±0.088), illustrating why we report both views: frequent duplicated paths can favor a continuously adapting model even while pattern-level generalization is slightly lower.

B. RQ2: Can CADRE-FP recover from severe template renaming?

Complete renaming is the clearest success case. Post-drift pattern PR-AUC is 0.599±0.011 for CADRE-FP, 0.541±0.008 for Online-ID, and 0.214±0.025 for Static-ID. The absolute CADRE-FP gain is 0.057 over continuous online learning and 0.385 over the static identity model. Occurrence-weighted PR-AUC is 0.299±0.072, compared with 0.249±0.067 and 0.054±0.019, respectively. All five repeats detect drift at batch 4, the known onset.

The per-batch trajectory clarifies why the canonical path matters. At batch 4, immediately after the unseen namespace appears and before any new label can update the identity learner, CADRE-FP retains mean pattern PR-AUC 0.587 through its canonical expert. Online-ID and Static-ID are both near 0.22 because neither has yet learned the new token identities. Once labels arrive, Online-ID recovers rapidly; CADRE-FP combines that plastic recovery with decreasing canonical weight and reaches 0.647 mean pattern PR-AUC in the final batch. Static-ID remains near 0.21 throughout the renamed period.

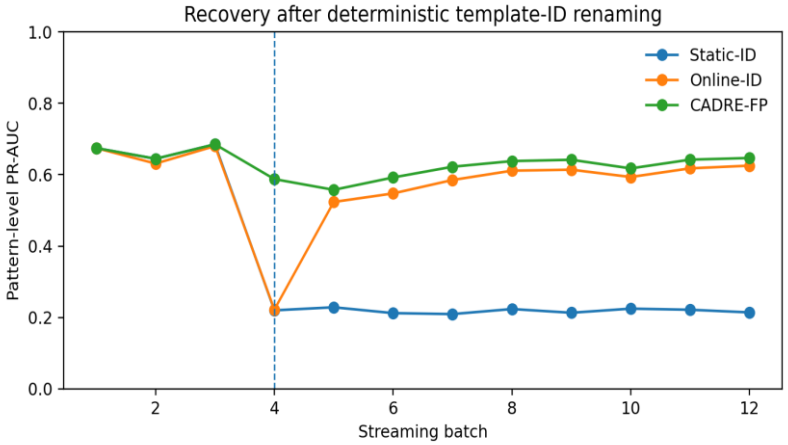


Figure 2. Pattern PR-AUC by streaming batch under deterministic template-ID renaming. The vertical line marks batch 4, when drift begins.

C. RQ3: What happens under gradual and class-conditional drift?

Under gradual template splitting, CADRE-FP obtains 0.631±0.012 post-drift pattern PR-AUC and 0.387±0.034 occurrence PR-AUC. Online-ID is numerically identical because drift severity remains below the 0.20 threshold for canonical mixing, so CADRE-FP deliberately behaves as the one-pass plastic learner. Both exceed Static-ID (0.580±0.013 pattern PR-AUC and 0.283±0.094 occurrence PR-AUC). The controller activates at batch 4 in all five repeats but does not overreact with the severe-drift path.

The fault-signature scenario is a useful negative result. Its event changes are class-conditional and affect only eight training-selected failure-associated IDs, so marginal novelty and Jensen-Shannon shift are small. CADRE-FP achieves 0.641±0.011 pattern PR-AUC, close to Static-ID at 0.643±0.012 and Online-ID at 0.637±0.014. On occurrence-weighted PR-AUC, however, CADRE-FP reaches 0.277±0.080, below Online-ID at 0.344±0.073. Four of five repeats never trigger the controller; one triggers only at batch 9 through delayed feedback. The experiment therefore does not support a claim that CADRE-FP solves arbitrary concept drift. It supports a narrower claim: representation invariance is highly effective for syntactic ID churn, while subtle conditional drift needs a label-aware competence detector or richer semantic features.

Table 3POST-DRIFT PR-AUC OVER FIVE PATTERN-DISJOINT REPEATS

Scenario	Model	Pattern PR-AUC	Occurrence PR-AUC
Complete rename	Static-ID	0.214±0.025	0.054±0.019

Complete rename	Online-ID	0.541±0.008	0.249±0.067
Complete rename	CADRE-FP	0.599±0.011	0.299±0.072
Gradual split	Static-ID	0.580±0.013	0.283±0.094
Gradual split	Online-ID	0.631±0.012	0.387±0.034
Gradual split	CADRE-FP	0.631±0.012	0.387±0.034
Fault-signature	Static-ID	0.643±0.012	0.274±0.082
Fault-signature	Online-ID	0.637±0.014	0.344±0.073
Fault-signature	CADRE-FP	0.641±0.011	0.277±0.080
No-drift control	Static-ID	0.671±0.010	0.336±0.088
No-drift control	Online-ID	0.661±0.013	0.386±0.081
No-drift control	CADRE-FP	0.671±0.010	0.336±0.088

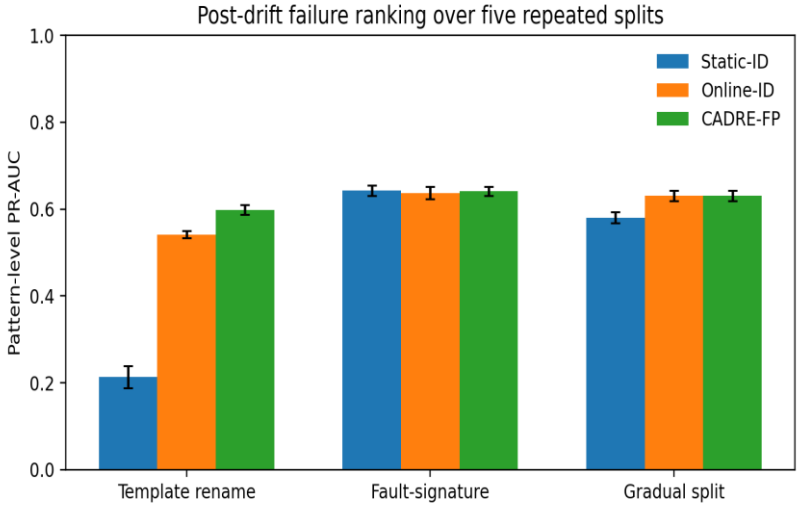


Figure 3Post-drift pattern PR-AUC for the three non-stationary stress scenarios. Error bars are one standard deviation over five splits.

D. RQ4: Does the unlabeled severity signal match the designed drift types?

The controller separates the scenarios in the intended way. Complete renaming produces severity 1.0 at batches 4 and 5 because every visible ID is novel. Severity then decays as the reference vocabulary incorporates the new namespace, which causes the canonical mixture weight to fall from 1.0 to zero by batch 8. Gradual splitting produces mean severity 0.089 at onset and then generally declines; it activates adaptation but never the severe canonical mixture. Fault-signature drift remains close to the stationary baseline because the changed IDs are sparse and class-conditional. This distinction is desirable for transparency: the controller's observable is input distribution change, not an oracle for every future loss increase.

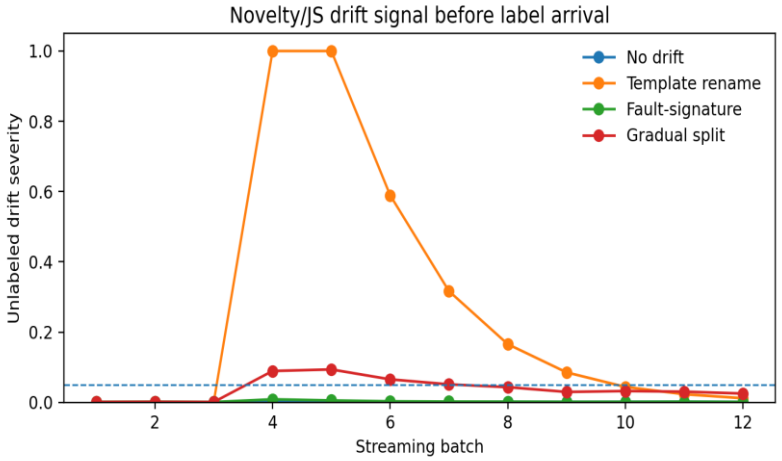


Figure 4Mean unlabeled drift severity by batch. The dashed horizontal line is the 0.05 activation threshold.

E. First-batch adaptation and controller diagnostics

A useful distinction is whether a method helps before the first post-drift label can be consumed. At batch 4, complete renaming drives all five CADRE-FP runs to canonical weight 1.0. Mean pattern PR-AUC is 0.587 for CADRE-FP, compared with 0.222 for Online-ID and 0.220 for Static-ID. This batch is the cleanest evidence for the representation contribution because no model can yet have learned the renamed identities from post-change labels. Gradual splitting activates adaptation in all five repeats but has zero canonical mixture at onset, so CADRE-FP and Online-ID are numerically identical there (0.646 pattern PR-AUC). The no-drift stream produces no activation at all, while fault-signature drift produces only three active batch evaluations out of 45 post-change evaluations, all attributable to one repeat after delayed evidence accumulates.

Table 4Mean unlabeled drift severity by batch. The dashed horizontal line is the 0.05 activation threshold.

Scenario	Batch-4 severity	Batch-4 canonical weight	Post-onset active
No drift	0.0016±0.0006	0.000	0 / 45
Complete rename	1.0000±0.0000	1.000	45 / 45
Gradual split	0.0894±0.0024	0.000	45 / 45
Fault-signature	0.0088±0.0009	0.000	3 / 45

The diagnostic table also clarifies what the detector does not claim. Severity summarizes only marginal prefix-vocabulary change; it is deliberately not reported as a direct estimator of future failure loss. The weak fault-signature signal is therefore an observed limitation rather than a detector error hidden by aggregate model scores.

VI. DISCUSSION

A. Why canonicalization helps without replacing identity features

The rename experiment provides a controlled mechanism test rather than only a headline score. Because the transformation is bijective, a purely structural representation should be unchanged. CADRE-FP behaves accordingly: its canonical expert gives useful rankings in the very first renamed batch, when a label-updated identity model cannot yet have adapted. This is not possible through ordinary online learning alone because the information needed to associate the new IDs with failures arrives only after the batch. Conversely, canonicalization cannot distinguish two executions whose equality topology is the same but whose concrete templates

carry different semantics. The identity expert is therefore retained, and its plastic version gradually takes back control as renamed labels accumulate.

This design extends a useful observation in the closed-loop CLARITY study: controlled event-template ID churn is a practical stressor for testing post-deployment feedback [1]. Our results add a complementary mechanism. Rather than relying solely on feedback after the churn, CADRE-FP creates an invariant structural path that can bridge the first unseen batch, then hands responsibility back to a learned identity model as evidence arrives. The relationship is additive rather than competitive: duplicate-aware evaluation and cautious feedback remain good practice, while representation-level invariance reduces how much the feedback path must relearn.

B. Operational interpretation

For cloud operations, a high score under complete renaming should not be interpreted as proof of resilience to arbitrary software upgrades. A parser update may also merge templates, split one event into several semantically different events, alter message grouping, or change control flow. Those cases break the assumptions behind Proposition 1. The controller therefore should be logged together with parser/model versions and should not silently authorize infrastructure actions. The hidden-technical-debt and ML-readiness literature emphasizes exactly this separation between model behavior, data dependencies, testing, and deployment controls [23], [24]. The same caution applies to delayed labels. In HDFS, outcome labels are available offline for every block sequence. Production labels may be delayed, censored, or affected by remediation. CADRE-FP's prequential ordering ensures that current-batch labels never influence current predictions, but it does not solve label quality. A practical implementation should record time-to-label, abstain from consolidation when labels stop arriving, and keep a rollback path for any model promotion. The AIOps literature similarly treats data quality, diagnosis, and operational integration as first-class problems rather than post-processing details [22].

C. Evidence boundary and negative findings

Two boundaries are central to the interpretation. First, the base data and labels are real, whereas drift is semi-synthetic: deterministic event-ID transformations are applied to held-out real sequences. This allows controlled paired comparisons but does not estimate how often each drift type occurs in a live cloud. Second, the fault-signature result prevents an overly broad robustness claim. A marginal novelty/Jensen-Shannon controller may miss a class-conditional change that leaves aggregate vocabulary statistics almost stationary. That failure mode suggests a concrete next step: combine unlabeled syntactic drift with a delayed supervised detector over residual loss or expert disagreement, borrowing ideas from error-based drift detection [18] and adaptive windows [19].

VII. THREATS TO VALIDITY

External validity is limited by one public HDFS corpus. HDFS block traces do not cover host metrics, network partitions, container orchestration, service graphs, or modern observability schemas. Loghub is a valuable benchmark [3], but generalization to production cloud systems requires additional datasets and action-specific validation. The sequence files are already parsed, so raw-text parser errors are outside the experiment. Construct validity is limited by the controlled drift generator. Complete renaming is intentionally clean and favors the canonical invariance property; it should be read as a mechanism stress test, not a prevalence estimate. Gradual splitting is more heterogeneous but still operates on IDs rather than raw log semantics. Fault-signature drift uses held-out labels inside the offline generator to create a class-conditional change. This is acceptable for stress construction because the predictor never receives those labels before prediction, but it is not a naturally observed intervention. Internal validity is strengthened by exact-pattern separation, fixed checksums, deterministic transformations, delayed-label ordering, and five repeated splits. Nevertheless, hashed feature collisions and stochastic SGD order can affect absolute values. All seeds and outputs are

supplied to make those effects auditable. Because the threshold is selected on a disjoint calibration partition, test batches do not tune the F1 threshold; PR-AUC remains the primary threshold-free measure.

Conclusion validity is limited by five repeats and three designed drift families. We therefore report mean and standard deviation, paired absolute gains, and negative results rather than claiming universal superiority. Deep log architectures are represented in related work but are not retrained as baselines in this compact experiment; the empirical comparison isolates adaptation mechanisms under identical sparse prefix features, not state-of-the-art model capacity.

VIII. CONCLUSION

CADRE-FP reframes adaptive cloud failure prediction as a combination of invariance and selective learning. The method preserves template-specific evidence through an identity expert, protects against unnecessary updates with a stable anchor, adapts through a delayed-label plastic learner, and bridges severe template-ID churn with a first-appearance canonical expert that is exactly invariant to bijective renaming. On real HDFS_v1 sequences with controlled label-preserving drift, the approach substantially improves the first-batch and aggregate ranking under complete template renaming, matches continuous online learning under gradual splitting, and remains inactive in the no-drift control. The class-conditional fault-signature stressor exposes where the current unlabeled detector is insufficient. The main contribution is therefore not a claim that one benchmark model solves cloud concept drift, but a reproducible design pattern: neutralize known nuisance transformations structurally, adapt only when evidence warrants it, and keep measured data separate from stress assumptions.

REFERENCES

M. Rhee, J. Zou, and Y. Qin, “CLARITY: A Closed-Loop Framework for Duplicate-Aware Failure Prediction and Budgeted Automated Remediation in Cloud Infrastructure,” AI and Data Science Journal, vol. 1, no. 1, pp. 76–85, 2024, doi: 10.61784/adsj4039.

Zhang, S. and Qiu, L. (2024). Data-driven assessment of pavement performance and resilience under extreme weather using the Long-Term Pavement Performance Program. AI and Data Science Journal, vol. 1, no. 1, pp. 57–66

Zhang, H. (2024). Graph-Aware Multi-Task Learning For Early Prediction Of Timing Violations And Routing Congestion In Digital Ic Physical Design. World Journal of Information Technology, 2(1): 13-20.

Wang, J., Tan, Y., Jiang, B., Wu, B., & Liu, W. (2025). Dynamic marketing uplift modeling: A symmetry-preserving framework integrating causal forests with deep reinforcement learning for personalized intervention strategies. Symmetry, 17(4), 610.

Zhao, X., Liu, J., Wang, Y., & Wang, J. (2026). CryptoMamba-SSM: Linear complexity state space models for cryptocurrency volatility prediction. IEEE Open Journal of the Computer Society, 7, 226-243.

Wang, J., Liu, J., Zheng, W., & Ge, Y. (2025). Temporal heterogeneous graph contrastive learning for fraud detection in credit card transactions. IEEE Access.

Sun, T., Yang, J., Li, J., Chen, J., Liu, M., Fan, L., & Wang, X. (2024). Enhancing auto insurance risk evaluation with transformer and SHAP. IEEE Access, 12, 116546-116557.

Zhang, H. (2025). Reinforcement Learning Approaches for Layout Optimization in Electronic Design Automation with Electromagnetic Compatibility Constraints. Frontiers in Robotics and Automation, 2(2), 77-93.

Han, X., Yang, Y., Chen, J., Wang, M., & Zhou, M. (2025). Symmetry-aware credit risk modeling: A deep learning framework exploiting financial data balance and invariance. Symmetry, 17(3), 341.

Chen, J., Cui, Y., Zhang, X., Yang, J., & Zhou, M. (2024). Temporal convolutional network for carbon tax projection: A data-driven approach. Applied Sciences, 14(20), 9213.

Sun, T., Wang, M., & Chen, J. (2025). Leveraging machine learning for tax fraud detection and risk scoring in corporate filings. Asian Business Research Journal, 10(11), 1-13.

Chen, J., Wang, M., & Sun, T. (2025). Intelligent tax systems and the role of natural language processing in regulatory interpretation. American Journal of Machine Learning, 6(4), 74-94.

Chen, Z., Liu, J., & Chen, J. (2025). Machine Learning Methods for Financial Forecasting in Enterprise Planning: Transitioning from Rule-Based Models to Predictive Analytics. Frontiers in Artificial Intelligence Research, 2(3), 541-564.

Chen, J., Liu, J., Liang, Y., & Zhou, M. (2026). KE-MLLM: A knowledge-enhanced multi-sensor learning framework for explainable fake review detection. Applied Sciences, 16(6), 2909.

Yue, X., Yang, J., & Liu, W. (2026). FraudDebate-Agent: A Multi-Agent LLM Framework with an Evidence-Based Debate Mechanism for Financial Statement Fraud Detection. Mathematics, 14(15), 2695.

Zhu, R. & Yue, X. (2024). An Explainable Machine Learning Framework For Predicting Dividend Sustainability And Financial Risk Of U.s. Reits Under Interest-Rate Shocks. Journal of Trends in Finance and Economics, 1(2): 48-57.

Yue, X. & Zhu, R. (2024). An Explainable Hybrid Machine Learning Framework For Cash-Flow-Conditioned Multi-Horizon Liquidity Risk Early Warning: An Sme-Oriented Benchmark Study. World Journal of Management Science, 2(1): 63-72.

Cui, Y., Han, X., Chen, J., Zhang, X., Yang, J., & Zhang, X. (2025). FraudGNN-RL: a graph neural network with reinforcement learning for adaptive financial fraud detection. IEEE Open Journal of the Computer Society, 6, 426-437.

Zi, B. (2024). Cloud-native distributed systems for real-time payment intelligence. AI and Data Science Journal, 1(1), 51-56.

Zhang, S. & Qiu, L. (2023). An Interpretable, Uncertainty-Aware Framework For Bridge Deterioration Prediction And Risk-Based Maintenance Prioritization. Academic Journal of Architecture and Civil Engineering, 1(2): 21-28.

Jin, J., Xing, S., Ji, E., & Liu, W. (2025). Xgate: explainable reinforcement learning for transparent and trustworthy api traffic management in iot sensor networks. Sensors, 25(7), 2183.

Xing, S., Wang, Y., & Liu, W. (2025). Multi-dimensional anomaly detection and fault localization in microservice architectures: A dual-channel deep learning approach with causal inference for intelligent sensing. Sensors, 25(11), 3396.

Xing, S., & Wang, Y. (2025). Proactive data placement in heterogeneous storage systems via predictive multi-objective reinforcement learning. IEEE Access, 13, 117986-117998.

Ji, E., Wang, Y., Xing, S., & Jin, J. (2025). Hierarchical reinforcement learning for energy-efficient API traffic optimization in large-scale advertising systems. IEEE Access.

Xing, S., & Wang, Y. (2025). Cross-modal attention networks for multi-modal anomaly detection in system software. IEEE Open Journal of the Computer Society.

Liu, Y., Ren, S., Wang, X., & Zhou, M. (2024). Temporal logical attention network for log-based anomaly detection in distributed systems. Sensors, 24(24), 7949.

Ren, S., Jin, J., Niu, G., & Liu, Y. (2025). ARCS: Adaptive reinforcement learning framework for automated cybersecurity incident response strategy optimization. Applied Sciences, 15(2), 951.

Li, P., Ren, S., Zhang, Q., Wang, X., & Liu, Y. (2024). Think4SCND: Reinforcement learning with thinking model for dynamic supply chain network design. IEEE Access, 12, 195974-195985.

Hu, X., Zhao, X., Wang, J., & Yang, Y. (2025). Information-theoretic multi-scale geometric pre-training for enhanced molecular property prediction. Plos one, 20(10), e0332640.

Wang, M., Zhang, X., Yang, Y., & Wang, J. (2025). Explainable machine learning in risk management: Balancing accuracy and interpretability. Journal of Financial Risk Management, 14(3), 185-198.

Yang, J., Li, P., Cui, Y., Han, X., & Zhou, M. (2025). Multi-sensor temporal fusion transformer for stock performance prediction: An adaptive Sharpe ratio approach. Sensors, 25(3), 976.

Zhao, X., Sun, T., Ren, S., Yang, J., & Liu, Y. (2025). RAG-Based AI Agents for Enterprise Software Development: Implementation Patterns and Production Deployment. Frontiers in Artificial Intelligence Research, 2(3), 501-520.

Sun, T., Wang, M., & Chen, J. (2025). Leveraging machine learning for tax fraud detection and risk scoring in corporate filings. Asian Business Research Journal, 10(11), 1-13.

Bian, M., Li, P., Lin, Y., Wang, Y., & Teng, D. (2026). Assessing Green Public Supply Chains Resilience under Carbon Neutrality Goals: A Multi-Agent Reinforcement Learning Framework. IEEE Access.

Teng, D. (2025). TEAS: Token-and energy-aware autoscaling for cost-efficient LLM serving. AI and Data Science Journal, 6(3), 1-10.

Rhee, M., Zou, J., Mo, T., Teng, D., & Yang, J. S. (2026). Enhancing web search agents with self-play contrastive fine-tuning. IEEE Access.

Wang, B., Zhao, W., & Wang, Z. (2024). Das-Gnn: A Scalable Disagreement-Aware Graph Neural Framework For Anomaly Detection In Large-Scale Networks. AI and Data Science Journal, 1(1): 67-75.

.