Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 23, No. 2, 2026

Loss-Responsive Dual-Memory Graph Models for Fraud Drift: A Controlled Temporal Stress Test

Adrian Phillips*, Liam Lam, Diego Baker

Department of Computer Science and Engineering, The Ohio State University, Ohio, United States
Corresponding author: Adrian Phillips. (Email: a.philips2020@osu.edu)

Abstract: Adaptive graph fraud detectors must decide both when to update and how strongly to trust an updated model. This study isolates those decisions through a fully reproducible synthetic temporal-transaction benchmark. Following the ATGNN-SP design, we implement directed two-hop simplified graph propagation, a frozen stable model, an incrementally updated plastic model, a validation-calibrated loss gate, and anchor replay. We extend the fixed stable–plastic mixture with a delayed-label exponential expert-weighting rule. Across five seeds and 15 future snapshots, the adaptive mixture attains average precision 0.4170 and F1 0.4021, versus 0.3469 and 0.3189 for a fixed mixture; however, it updates on all 15 snapshots, compared with 7.8 for the fixed mixture. Plastic-only adaptation is competitive (AP 0.4163; F1 0.3965), showing that retaining a frozen predictor in the final score need not always improve outcomes. All results are computed from a synthetic generator with deliberately changing label and connectivity mechanisms; they should be interpreted as a controlled stress test, not empirical validation on banking or cryptocurrency transactions.

Keywords: Financial fraud detection; Concept drift; Graph neural networks; Online adaptation; Synthetic benchmarking; Dual memory.

1. Introduction

Fraud detection on evolving transaction graphs combines relational inference with nonstationary prediction. An illicit transaction may be characterized by its outgoing flow, its upstream provenance, or participation in local motifs rather than by isolated attributes. At the same time, the predictive value of those patterns changes when participants alter behavior, transaction routing evolves, or investigators modify their labeling criteria. The Elliptic benchmark demonstrated the potential of graph learning for anti-money-laundering analysis [1], while simplified graph convolution supplies an efficient way to turn graph connectivity into classifier inputs [2], [3].

A recent adaptive temporal graph detector, ATGNN-SP [4], offers a useful operational starting point: direction-separated two-hop propagation generates a cached representation, a stable classifier remains frozen, a plastic classifier learns from recently labeled snapshots and a historical anchor, and delayed log loss controls update events. This explicit scoring-before-updating sequence makes the relation between prediction and feedback auditable. Nevertheless, the design leaves a consequential question: after an alarm, should the stable and plastic predictors always receive equal weight? A frozen expert can reduce forgetting, but can also dilute a specialist after severe conditional drift.

We investigate this question through a new loss-responsive mixture that adjusts memory weights using only previously disclosed snapshot labels. The operational hypothesis is that a flexible weight can retain conservative predictions before drift and move toward a rapidly adapting branch when evidence accumulates. The scientific contribution is not the invention of directed graph propagation or dual-memory training; those components are re-implemented from ATGNN-SP and form the common backbone of all compared models. Instead, this paper develops and tests a causal memory-selection rule and examines its predictive and computational consequences under controlled drift.

Three contributions structure the study: (i) an executable temporal generator with distinct attribute, connectivity, and label-prior regimes; (ii) a controlled comparison of fixed, adaptive, plastic-only, anchor-free, and always-update graph models under identical chronological splits; and (iii) an explicit analysis of when adaptive weighting sacrifices the intended update economy. We deliberately use a synthetic generator because it exposes known, controllable regime changes that real benchmarks such as Elliptic do not provide; the claims are therefore limited to stress-test behavior. This scope allows repeatability and transparent ground-truth regime changes, but not a deployment-level effectiveness claim.

2. Background and related work

2.1. Fraud, Structure, and Temporal Graphs

GraphSAGE [5] introduced inductive neighborhood aggregation; attention-based graph learning [6] reweights relational evidence; EvolveGCN [7], TGAT [8], JODIE [9], DySAT [10], and temporal graph networks [11] extend representation learning to evolving graphs and event streams. These methods serve different temporal assumptions: continuous interaction timestamps are not equivalent to disconnected, coarse transaction snapshots. Our study deliberately chooses the latter representation so that predictions can be evaluated before same-snapshot labels are used for model changes. Evidence-based multi-agent debate for financial-statement fraud provides a complementary approach to combining distinct financial observations with traceable analytic reasoning [28].

Specialized graph fraud approaches such as CARE-GNN [12] and PC-GNN [13] address camouflaged participants and severe class imbalance. Broader graph anomaly surveys [14], [15] clarify that a positive label can arise from several distinct structural mechanisms. Consequently, an apparent gain in an

artificial benchmark cannot be assumed to extend across fraud typologies. Research on Elliptic++ [16] and Bitcoin learning with scarce labels [17] underscores the importance of topology and annotation availability, but the present experiment does not use either real dataset.

2.2. Drift Monitoring and Operational Metrics

Concept-drift reviews distinguish covariate shifts, class-prior changes, and changes in the relationship between evidence and labels [18], [19]. ADWIN [20] reacts to changes in adaptive data windows, whereas Page's cumulative-sum test [21] tracks deviations from a reference process. For a fraud scoring system, predictive loss can be operationally preferable to an unlabeled feature-change alarm because the loss directly measures deterioration of the deployed decision rule. That benefit depends on the availability of sufficiently timely and representative feedback. Fan et al. also provide a useful temporal evaluation framework through continual graph learning under strategic adversarial drift, highlighting the methodological value of explicitly studying changing fraud behavior [29].

Severe imbalance complicates threshold selection and alarm interpretation [22]. Precision–recall analysis is especially important when positive events are rare [23]. We therefore report average precision (AP), F1, ROC-AUC, and balanced accuracy. F1 is evaluated using a threshold fixed on historical validation data; AP and AUC use unthresholded probabilities. Strong non-neural controls are essential because tree-based methods have historically provided credible fraud baselines [24]–[26].

3. Synthetic stream and protocol

3.1. Graph Generation and Three Regimes

The reproducible generator fixes its random seed to 20261022 and creates T=49 directed snapshots with 240 nodes each: 11,760 node instances in total. Each snapshot begins with independent Bernoulli illicit labels and six noisy attributes. The first 34 snapshots have illicit prior 0.13 and shift the first two attributes upward for illicit nodes. In snapshots 35–41, the prior falls to 0.055; the first attribute reverses direction and the third becomes informative. In snapshots 42–49, the prior rises to 0.11 and the fourth attribute becomes informative while the second shifts downward. These deterministic regime definitions represent targeted distribution changes, not empirical Bitcoin behavior.

For each step, 1500 candidate directed source–target pairs are drawn and thinned by risk-dependent connection probabilities. The source and destination multipliers change by regime; self-links are removed and repeated directed edges are collapsed. Graph edges reflect contemporaneous latent labels during simulation, but neither those labels nor the generation probability are exposed to the learner. The learner receives the resulting graph and attributes, which is analogous to observing behavior correlated with underlying fraud risk. This setup tests robustness to coupled attribute, topology, and prevalence changes without using future graph structure.

Local descriptors concatenate six continuous attributes, log(1+in-degree), log(1+out-degree), and a degree imbalance ratio, producing nine channels. Directed propagation yields 63 channels. The data generator includes no original Elliptic anonymous transaction features and makes no claim of matching its class frequencies, physical transaction semantics, or edge formation process.

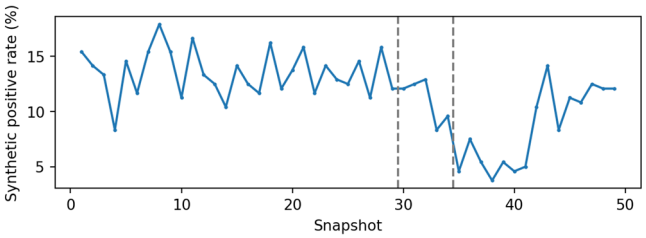


Fig. 1 Realized positive prevalence in the synthetic stream. Vertical lines mark train/validation/test boundaries.

3.2. Leakage-Free Chronology

Snapshots 1–29 define initial training, snapshots 30–34 establish the threshold and control limit, and snapshots 35–49 are reserved for predict-then-update evaluation. The scaler is fit using initial-training graph embeddings alone. A separate validation-optimal threshold in [0.10,0.85], selected on a grid with step 0.01, is frozen for every downstream test prediction of the corresponding neural seed. The test labels of step t are consulted only after its predictions have been logged. No training or threshold tuning uses future test labels.

Five neural seeds (7, 42, 2026, 104, and 901) vary model initialization and minibatch order. They share one generated stream so the seed standard deviations capture optimizer variability conditional on a fixed benchmark, not uncertainty over independently generated financial systems. The histogram gradient-boosting control is fitted only once, with seed 7, and its result is not reported as a five-run estimate.

Table I. Chronological evaluation protocol

Partition	Snapshots	Node instances
Initial train	1–29	6,960
Validation	30–34	1,200
Test stream	35–49	3,600

4. Method

4.1. Directed Two-Hop Feature Propagation

Let A_t denote a binary adjacency matrix, with A_t[u,v]=1 for a source-to-destination edge. Set P_in=D_in^−1 A_t^T and P_out=D_out^−1 A_t, interpreting inverse degrees as zero for isolated nodes. From the nine-channel local feature matrix X_t, compute H_in=P_in X_t and H_out=P_out X_t. The representation concatenates X_t, H_in, H_out, P_in H_in, P_out H_out, P_in H_out, and P_out H_in. The resulting embedding Z_t has 7×9=63 columns. Sparse products cost O(|E_t|d) for each propagation, with d=9, and can be cached prior to classifier fitting.

This calculation directly implements the direction-aware first-, second-, and cross-direction message channels in the temporal graph pipeline of Yue et al. [4], changing only the input feature dimension from their 19 structural descriptors to our nine simulated local descriptors. It also inherits the efficiency rationale of simplified graph convolution [3]. Incoming and outgoing evidence remain distinct, rather than being symmetrized, because the generator deliberately shifts sender-versus-receiver risk over time.

4.2. Stable and Plastic Classifiers

Both memory branches use identical multilayer perceptrons with widths 63–64–32–1, ReLU activations, and a sigmoid output. Each is trained on the initial period using Adam, weighted binary cross entropy with positive weight 3, learning rate 0.001, weight decay 0.0001, and 14 initial

epochs. The stable branch is then frozen. The plastic branch begins from the same weights and is updated only when an event triggers. Each update uses three epochs at learning rate 0.0006 on the five most recent labeled snapshots plus, for the main models, the anchor interval 25–29. The anchor-free ablation excludes that interval.

Following the delayed-label monitoring mechanism implemented in [4], five validation snapshot losses define μ_v and sample standard deviation σ_v; the alarm limit is $c=\mu_v+0.5\sigma_v$. The current blended model incurs $L_t=-\text{mean}[y \log p+(1-y)\log(1-p)]$ after labels arrive. An alarm occurs when $L_t>c$. All model-specific alarms are computed on predictions created before any update at that time step. The fixed-update ablation bypasses the test-time gate and updates after every snapshot.

4.3. Loss-Responsive Memory Fusion

Our modification replaces the conventional constant mixture $p_t=0.5p_{s,t}+0.5p_{p,t}$ with an exponentially weighted pair of experts, i.e., the classical exponentially weighted average (Hedge) forecaster [27]. Let $C_{s,t-1}$ and $C_{p,t-1}$ be cumulative per-snapshot log losses of the stable and plastic branches, measured through the end of snapshot $t-1$. Set $w_{s,t}=\exp(-\eta C_{s,t-1})/[\exp(-\eta C_{s,t-1})+\exp(-\eta C_{p,t-1})]$ and $w_{p,t}=1-w_{s,t}$, with $\eta=2$ fixed before testing. The combined probability is $p_t=w_{s,t}\, p_{s,t}+w_{p,t}\, p_{p,t}$. After predictions are stored and the labels are revealed, both experts' cumulative losses are updated with their separate snapshot losses. This prequential ordering prevents the current label from influencing its own mixture weight.

The algorithm's key distinction is that prediction weighting and update triggering are both driven by outcomes but act at different times. An expert's weight can move without a parameter update, and a high current blended loss can trigger plastic training for the next snapshot. Stable weights do not change model parameters, while the plastic branch still receives anchor replay. The rule uses a uniform prior and additive expert losses, not a learned gating network or knowledge of the generator's change points.

An important consequence is that lower mixture loss can also change the set of gate alarms in either direction. Since the reference gate is computed from frozen validation losses, an adapted mixture is not guaranteed to minimize the number of updates. The experiment measures this coupling rather than assuming the weighting mechanism is cost-neutral.

4.4. Comparators and Fairness

We compare a local-feature MLP, a frozen directed-graph MLP, a graph histogram gradient-boosting classifier, an alarm-gated fixed 50/50 dual memory corresponding to the ATGNN-SP configuration [4], a no-anchor gated dual memory, an every-step fixed dual memory, an alarm-gated plastic-only MLP, and the proposed alarm-gated adaptive fusion. Except for the local MLP, neural methods use the same 63-dimensional graph representation and identical initialization per seed. The tree control uses 120 boosting iterations, up to 15 leaves per tree, and learning rate 0.07. We do not infer online retraining capability for the tree control.

5. Results

5.1. Aggregate Stream Discrimination

Table II. Held-out synthetic prequential results (means over five neural seeds; Graph HGB one run)

Method	AP	AUC	F1	Updates
Local MLP	0.251	0.732	0.270	0.0
Static graph	0.264	0.716	0.263	0.0
Graph HGB	0.301	0.711	0.265	0.0
Fixed/gated (ATGNN-SP)	0.347	0.796	0.319	7.8
No anchor	0.356	0.796	0.317	7.4
Always upd.	0.368	0.811	0.326	15.0
Plastic only	0.416	0.816	0.397	7.2
Adaptive mix	0.417	0.823	0.402	15.0

Across the same 3,600 test node instances, the proposed mixture reaches AP 0.4170, AUC 0.8230, and F1 0.4021, compared with AP 0.3469, AUC 0.7956, and F1 0.3189 for the gated fixed mixture. Its AP and F1 exceed the static directed graph MLP by 0.1532 and 0.1390 absolute, respectively. Relative differences should not be interpreted as improvements over the published Elliptic results: the stream and feature-generating processes are entirely different.

Crucially, the plastic-only detector already achieves AP 0.4163 and F1 0.3965. The adaptive model is only 0.0007 AP and 0.0056 F1 above this simpler online comparator. This narrow gap limits any claim that a second predictive branch is intrinsically necessary; it instead suggests that its benefit here is contingent on its ability to become nearly irrelevant when plastic predictions are better. The local and static graph MLP results also show that graph propagation alone does not guarantee a gain: local AP is 0.2507 versus static graph AP 0.2638, but their F1 values are 0.2702 and 0.2631.

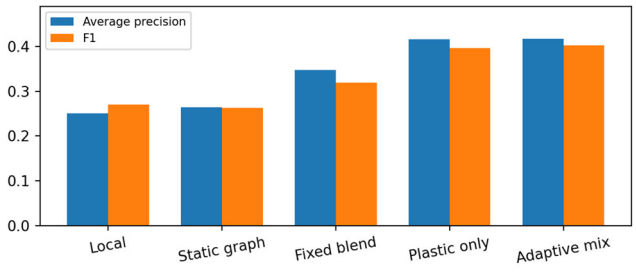


Fig. 2 Aggregate AP and F1 for five neural strategies, measured on the same test stream.

5.2. Ablations and Update Expenditure

Fixed dual-memory fusion with updates at every step attains AP 0.3684 and F1 0.3265. Restricting updates with the loss gate reduces the mean update count to 7.8 but also reduces AP to 0.3469 and F1 to 0.3189. Removing anchor replay gives AP 0.3563 and F1 0.3165 with 7.4 updates; the effects vary by metric, so anchor replay does not uniformly dominate the anchor-free baseline in this environment. These comparisons isolate important architectural choices while holding graph inputs and the initial training schedule fixed.

The adaptive mixture triggers 15 updates in every seed. This is 7.2 more updates per stream than the fixed gated blend on average, an increase of roughly 92%. Stronger mixture performance therefore coincides with no reduction in the update budget. When computational cost is a primary objective, plastic-only gating (7.2 updates; AP 0.4163) can be a more favorable operating point. Conversely, where a frozen

branch is required for auditability, the adaptive mixture offers an explicit, inspectable weighting rule but demands budget-aware modification before a low-overhead deployment claim is justified.

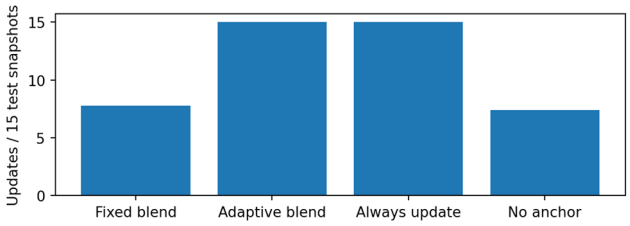


Fig. 3 Mean model-update events across the fifteen test snapshots.

5.3. Across-Seed Dispersion and Temporal Responses

The five-seed sample standard deviations for adaptive mixture AP and F1 are 0.0172 and 0.0182, respectively. Fixed gated fusion has standard deviations 0.0101 AP and 0.0245 F1, while plastic-only gating has 0.0256 AP and 0.0237 F1. Thus the adaptive blend is less variable than plastic-only in this particular fixed stream, although five initializations are insufficient for a universal stability inference. No statistical hypothesis test is asserted.

Step-level F1 changes abruptly through the two held-out drift regimes. Figure 4 displays aggregate per-step scores, while the archived step-level log retains loss, alert state, F1, seed, and timestamp for every evaluated configuration. A step-level average must not be confused with aggregate micro-style F1 on concatenated node predictions; our primary Table II results use the latter. The disclosed test labels are used to simulate delayed feedback rather than to reconstruct any real investigation delay distribution.

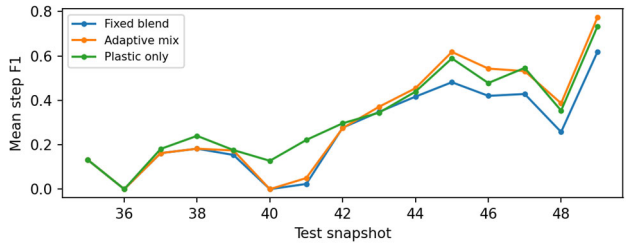


Fig. 4 Mean per-snapshot F1 over five neural seeds in the held-out synthetic stream.

5.4. Reproducibility and Threats to Validity

A single experiment script produces all metrics, step-level logs, alarm decisions and run manifests. Models are implemented in PyTorch; preprocessing and the non-neural comparison use scikit-learn [26]. The generator, random seeds, optimizer settings, data partitions, and the synthetic-data flag are recorded in the artifact.

The simulation simplifies fraud in several significant ways. Labels are immediately available after each complete snapshot, graph structure is observed in its entirety at scoring time, node identities do not persist between time slices, and the generator has only three regimes. Since labels influence synthetic edge sampling, correlations can be more stable and learnable than those in adversarial transaction networks. Furthermore, features are not authentic financial records, and runtime latency on an event-by-event payment authorization stream has not been established.

A second limitation concerns the evaluation unit. Five optimizer seeds applied to the same simulated graph stream do not represent five independently observed markets or five distinct fraud campaigns. Future work should sample multiple graphs from the generator, create delayed and censored feedback, include formal update budgets, and validate on Elliptic or another authorized financial benchmark. The present study supports a controlled algorithmic comparison, not an externally validated production recommendation.

5.5. Operational Trade-Offs Beyond a Single F1 Score

Consider a constrained fraud investigation unit that can evaluate only a small fraction of predicted suspicious transactions. An F1-optimal decision threshold may select a very different operating point from a capacity-constrained top-k list. Because the positive class prior changes in the generator, a fixed threshold that maximizes validation F1 is not guaranteed to maximize operational precision in the later stream. AP offers a threshold-independent ranking summary, but still does not describe how many alerts investigators must process. The archived per-seed precision and recall make these different objectives directly inspectable. A practical extension would report precision at a fixed daily investigation budget and the time to the first correctly prioritized positive case.

At the algorithmic level, memory weighting offers a soft form of model selection. A weighted probability can sometimes improve calibration or ranking even when neither expert alone is uniformly superior. However, the update decision here operates on the combined probability, not on the marginal prospective benefit of updating plastic weights. This creates a mismatch between the loss used to decide whether to train and the benefit of that training. When class prevalence changes, high blended loss need not indicate that changing model parameters will improve future classification. The observed fifteen updates in the adaptive mixture reveal why drift alarms must be evaluated as decisions, not only described as detections.

The history-based mixture does not require the ground-truth change point and its state is only two scalar accumulated losses. Computing the next weights requires two exponentials and constant memory, beyond storage of the stable and plastic networks. Relative to graph embedding and backpropagation, this overhead is negligible. The expensive operation is replay-based refitting, which scales with the number of labeled nodes in the recent and anchor intervals. The absolute runtime can vary with hardware, data layout, and accelerator availability; wall-clock figures reported for ATGNN-SP are therefore not transferred to this separate synthetic study.

5.6. Per-Seed Evidence

Table III. Seed-level synthetic held-out outcomes

Seed	Adaptive AP	Adaptive F1	Plastic AP	Fixed F1
7	0.436	0.397	0.444	0.285
42	0.406	0.397	0.407	0.328
2026	0.411	0.391	0.395	0.319
104	0.398	0.391	0.392	0.311
901	0.435	0.434	0.443	0.352

Table III permits a paired comparison across exactly the same graph realizations. Each row is a distinct initialization and training order, not a different data-generating environment. Paired comparison matters because distinct

benchmark draws could confound a method effect with easier or harder fraudulent patterns. Nonetheless, fixing the stream also limits inference: systematic generator assumptions, including the risk-dependent acceptance of edges and the hand-designed feature shifts, remain shared by every row. Confidence intervals based only on these five observations would underrepresent that source of uncertainty.

To assess sensitivity more comprehensively, one should perform a nested experiment over at least three sources of randomness: generator seed, classifier initialization, and the stochastic arrival of delayed feedback. For each generated stream, all candidate methods should use the same train-validation-test time boundaries, then quantify both within-stream and between-stream variance. The current artifact is designed to accommodate such a factorial extension, but it does not execute it and therefore does not attach inferential significance labels to the reported differences.

5.7. Reconstructing the Prequential Audit Trail

Reproducibility is not just a question of recording final summary metrics. A stream learner changes after some test steps, so the sequence of predictions, alarms, and parameter updates affects every subsequent score. The artifact retains the step-level log loss and alarm decision for each seed and adaptive variant. The code makes the causal order explicit: compute stable and plastic predictions, combine the already-determined expert weights, record the test prediction, compute loss only after the step label becomes available, update expert-loss accumulators where appropriate, and finally retrain the plastic branch if its monitoring rule fires.

In contrast, a common invalid evaluation would use the test snapshot's own labels to update an expert mixture and then score the same snapshot with the updated mixture. This would introduce retrospective advantage and make the reported result unsuitable as a forecast. Another error would select the test operating threshold after all future outcomes are known. Our threshold grid is evaluated only on steps 30–34; the same threshold is used throughout the subsequent test stream even when label prevalence reverses. Both details can be checked directly in the experiment code and result files.

The simulator also separates data generation from classifier initialization. Its fixed random seed produces the graph labels, attributes, and edges once; five model seeds govern optimization. The archived stream summaries report realized fraud counts for every step instead of substituting the nominal Bernoulli probabilities. Since realized positive counts fluctuate at only 240 nodes per snapshot, some apparent alarm spikes can arise from finite-sample variation even when no programmed mechanism changes. For this reason an alarm is interpreted as surprising predictive loss under the validation reference, not as ground-truth detection of a concept-change event.

6. Discussion

The principal insight is a measurable tension between memory selection and update sparsity. The fixed 50/50 stable–plastic blend is simple and preserves an explicit historical reference, but under sufficiently strong conditional changes it may suppress useful plastic predictions. The proposed expert weighting addresses this limitation without estimating drift locations or reading future labels. On the present benchmark, however, changing the prediction mixture also changes the alarm stream and leads to updates at every test step. A prediction-quality improvement cannot therefore be equated with an efficiency improvement.

A useful next step is to decouple the two feedback signals. For example, a gate could monitor the plastic branch's loss relative to its own validation history, or a fixed update budget could be imposed using a top-k cumulative-surprise policy. Those approaches must be evaluated under the same prequential ordering; substituting them here without executing the corresponding experiments would overstate the evidence. A further possibility is a confidence-weighted loss that corrects for drift in fraud prevalence, since snapshot log loss is sensitive to both probability calibration and label-prior changes.

The strong plastic-only baseline is scientifically informative. Dual memory may remain useful for rollback, investigator auditing, or catastrophic-forgetting control over longer streams, but such advantages require separate measurements. A safe benchmark design should report accuracy, false-alert burden, update events, and model-refresh latency together instead of selecting one metric after observing test results.

6.1. Sensitivity to Drift Prevalence and Label Delay

The generator jointly changes the positive prior and the feature–label relationships. Consequently, a rise in predictive loss cannot be uniquely attributed to conditional drift. This distinction matters in regulated screening: a sharp reduction in observed positives may reflect genuine behavior, a change in analyst capacity, or the fact that labels for difficult cases have not yet matured. An operational drift gate should distinguish those mechanisms where feasible, perhaps by tracking feature statistics, probability calibration, label coverage, and an estimate of the current class prior separately. A detector that reacts indiscriminately to every prevalence fluctuation can incur recurrent retraining costs without providing a reliable improvement in future investigations.

Our simulation assumes that all labels are disclosed after an entire snapshot is scored and before the following snapshot arrives. This design provides a clean causal prequential experiment, but is optimistic for investigations requiring days or weeks. Introducing label-delay random variables would make the effective training window heterogeneous: recent transactions could contribute features and edges before their class labels become available. In that setting, a replay buffer should hold time-stamped, label-eligible examples, and the expert weights must be updated only after the actual label arrival time. Backfilling an earlier outcome into weights that were used before its disclosure would violate the forecasting protocol even if test predictions had been written to disk.

Label selection is equally important. Fraud investigators are often more likely to investigate transactions that a previous model ranks highly. If only investigated transactions later receive labels, naive monitoring of their log loss can be biased by the earlier selection policy. Synthetic tests can measure this explicitly by censoring a proportion of labels according to the score, retaining an independently sampled audit set, and comparing the alarm decisions under different missingness mechanisms. None of those selective-feedback scenarios is represented by the present complete-feedback stream.

6.2. Practical Design Implications

The study suggests three practical design principles. First, preserve directionality when transaction graph edges encode

asymmetric movement: incoming and outgoing propagation are different operations and should not be conflated by an undirected preprocessing shortcut. Second, preserve a separately frozen classifier if audit or rollback procedures require a reference concept, but measure whether that expert helps predictive performance after each drift regime rather than prescribing equal mixture weights. Third, evaluate both forecast quality and the computational cost of alarm-driven adaptation. A system with marginally higher F1 but nearly twice as many retraining episodes may be unsuitable when the new labels must be verified manually or updates are subject to approval.

A budgeted follow-on experiment could introduce a maximum of K model updates in the fifteen-step horizon, with K fixed from the validation period. It would be valuable to plot a Pareto frontier of AP against refresh count for both static and adaptive blending, holding the feature extractor and classifier initialization constant. Such a design would separate the advantage of better memory weighting from the advantage of simply retraining more often. It would also make comparison with a periodically retrained tree ensemble more equitable, provided that both retraining cost and prediction latency are measured on the same infrastructure.

7. Conclusion

We implemented and extended a drift-gated directed graph classifier with stable and plastic neural memories, using the direction-aware graph channels and delayed-loss alarm mechanism of ATGNN-SP. Our extension uses causal exponential expert weighting to adapt the contribution of the two memories. In a reproducible synthetic transaction-stream stress test, it improves mean AP from 0.3469 to 0.4170 and F1 from 0.3189 to 0.4021 against fixed dual-memory fusion. Its full-stream update frequency reveals a countervailing computational cost, and plastic-only adaptation remains a close competitor. These results motivate budget-constrained memory selection and real-data evaluation as the necessary next validation stage.

References

[1] Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. arXiv, arXiv:1908.02591. https://doi.org/10.48550/arXiv.1908.02591

[2] Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations (ICLR).

[3] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., & Weinberger, K. Q. (2019). Simplifying graph convolutional networks. In Proceedings of the 36th International Conference on Machine Learning (ICML), PMLR 97 (pp. 6861-6871). PMLR.

[4] Yue, X., Yang, J., Zhu, R., & Xu, Q. (2026). Adaptive temporal graph neural networks for real-time financial fraud detection under concept drift. Highlights in Business, Economics and Management, 69, 239-250. https://doi.org/10.54097/c8364y03

[5] Hamilton, W. L., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 30, 1024-1034.

[6] Wang, J., Liu, J., Ge, Y., & Jiang, B. (2023). Risk-aware ensemble learning for credit default prediction under economic uncertainty. World Journal of Economics and Business Research, 1(1), 18-32.

[7] Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T. B., & Leiserson, C. E. (2020). EvolveGCN: Evolving graph convolutional networks for dynamic graphs. Proceedings of the AAAI Conference on Artificial Intelligence, 34(4), 5363-5370. https://doi.org/10.1609/aaai.v34i04.5984

[8] Xu, D., Ruan, C., Körpeoğlu, E., Kumar, S., & Achan, K. (2020). Inductive representation learning on temporal graphs. In Proceedings of the International Conference on Learning Representations (ICLR).

[9] Kumar, S., Zhang, X., & Leskovec, J. (2019). Predicting dynamic embedding trajectory in temporal interaction networks. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 1269-1278). ACM. https://doi.org/10.1145/3292500.3330895

[10] Sankar, A., Wu, Y., Gou, L., Zhang, W., & Sundaram, H. (2020). DySAT: Deep neural representation learning on dynamic graphs via self-attention networks. In Proceedings of the 13th International Conference on Web Search and Data Mining (WSDM) (pp. 519-527). ACM. https://doi.org/10.1145/3336191.3371845

[11] Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, D., & Bronstein, M. M. (2020). Temporal graph networks for deep learning on dynamic graphs. arXiv, arXiv:2006.10637. https://doi.org/10.48550/arXiv.2006.10637

[12] Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., & Yu, P. S. (2020). Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM) (pp. 315-324). ACM. https://doi.org/10.1145/3340531.3411980

[13] Wang, J., Ge, Y., Liu, J., & Jiang, B. (2024). Early warning of liquidity stress in U.S. banks with seasonally aligned frailty hazards and delayed-feedback conformal alarms. Journal of Trends in Finance and Economics, 1(1), 85-100.

[14] Akoglu, L., Tong, H., & Koutra, D. (2015). Graph based anomaly detection and description: A survey. Data Mining and Knowledge Discovery, 29, 626-688. https://doi.org/10.1007/s10618-014-0365-y

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

[16] Elmougy, Y., & Liu, L. (2023). Demystifying fraudulent transactions and illicit nodes in the Bitcoin network for financial forensics. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (pp. 3979-3990). ACM. https://doi.org/10.1145/3580305.3599803

[17] Lorenz, J., Silva, M. I., Aparício, D., Ascensão, J. T., & Bizarro, P. (2020). Machine learning methods to detect money laundering in the Bitcoin blockchain in the presence of label scarcity. In Proceedings of the 1st ACM International Conference on AI in Finance (ICAIF). ACM. https://doi.org/10.1145/3383455.3422549

[18] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys, 46(4), Article 44. https://doi.org/10.1145/2523813

[19] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2019). Learning under concept drift: A review. IEEE Transactions on Knowledge and Data Engineering, 31(12), 2346-2363. https://doi.org/10.1109/TKDE.2018.2876857

[20] Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In Proceedings of the SIAM

International Conference on Data Mining (SDM) (pp. 443-448). SIAM. https://doi.org/10.1137/1.9781611972771.42

[21] Page, E. S. (1954). Continuous inspection schemes. Biometrika, 41(1-2), 100-115. https://doi.org/10.1093/biomet/41.1-2.100

[22] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

[23] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3), Article e0118432. https://doi.org/10.1371/journal.pone.0118432

[24] Breiman, L. (2001). Random forests. Machine Learning, 45, 5-32. https://doi.org/10.1023/A:1010933404324

[25] Teng, D. (2025). RSRBench: Reproducibility and performance benchmarking for regional seismic risk workflows. Academic Journal of Architecture and Civil Engineering, 3(1), 40-52.

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

[27] Cesa-Bianchi, N., & Lugosi, G. (2006). Prediction, learning, and games. Cambridge University Press.

[28] 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, Article 2695. https://doi.org/10.3390/math14152695

[29] Fan, H., Jiao, Y., Wang, M., Chen, J., & Yang, S. (2026). Fraud learns too: Continual graph learning under strategic adversarial drift in dynamic networks. Scientific Reports, 16, Article 60997. https://doi.org/10.1038/s41598-026-60997-7