International Journal of Advanced Engineering and Technology Research
ISSN: 3106-8677 (Print), ISSN: 3106-8685 (Online) | Vol. 4, No. 2, 2026

Temporal Graph Learning for Robust Detection of Coordinated Fraud in Dynamic Transaction Networks

Sofia Lindqvist [1, *], Johan Berg [2], Maya Svensson [1]

[1] Department of Information Technology, Uppsala University, Uppsala, Sweden

[2] Department of Computer and Systems Sciences, Stockholm University, Stockholm, Sweden

Abstract: Coordinated fraud can produce repeated, synchronized transfers whose signatures are obscured by heterogeneous activity, missing edges, and evolving behavior. This paper studies a lightweight temporal graph learning strategy based on account-level transaction attributes, static neighborhood descriptors, and eight-bin temporal coordination statistics. Static and temporal feature variants are each trained with the same simulated edge-deletion augmentation to disentangle the effects of timing descriptors and robustness training. Experiments are conducted on a controlled, fully specified synthetic benchmark containing 4,800 accounts, approximately 15,000 transfers per run, and simulator-assigned coordinated actors; the benchmark is not a real-world labeled fraud dataset. Three independent runs use chronological, nonoverlapping train/validation/test snapshots. The edge-augmented static graph achieves test average precision 0.631, while the identically augmented temporal feature model reaches 0.602; their nonaugmented counterparts score 0.571 and 0.569, respectively. At 20% and 40% test-edge removal, augmented static-graph average precision is 0.518 and 0.389, respectively. A matched augmentation control, fixed-budget top-100 precision, and timestamp shuffling separate the roles of training robustness and within-snapshot timing. The study provides an efficient, transparent protocol for evaluating coordinated-pattern detection in evolving transaction networks.

Keywords: Temporal graphs, coordinated fraud, synthetic transaction networks, anomaly detection, perturbation augmentation, random forest.

1. Introduction

Fraud in transaction platforms is increasingly relational: harmful actors may distribute activity across multiple accounts, repeatedly transfer value among accomplices, or concentrate outgoing payments in short windows. These behaviors can be weakly expressed in a single transfer but pronounced in a graph of account-to-account interactions. General anomaly detection surveys distinguish unusual individual observations from connected structural anomalies [1, 2]. Anti-money-laundering studies based on the Elliptic Bitcoin transaction graph [3] and the Elliptic++ benchmark [4] further motivate the use of network context, while DGraph shows that temporal and background-node information is relevant in financial graph anomalies [5].

Graph fraud research has explicitly studied camouflage and synchronized behavior. CARE-GNN selects relations under adversarial disguise [6]; FRAUDAR evaluates dense patterns in the presence of camouflage [7]; and CopyCatch targets coordinated, lockstep actions with timestamp constraints [8]. Stream anomaly detection approaches such as MIDAS [9], NetWalk [10], and SpotLight [11] demonstrate the value of examining graph changes rather than treating the aggregated network as stationary. These methods motivate temporal evidence but differ from the supervised account-level classification setting considered here.

A useful baseline need not rely on a deep temporal graph neural network. Our focus is a practical question: which aspects of a compact topology-and-timing representation remain useful under distribution shift and missing transactions when edge augmentation is controlled for? The study combines (i) a compact account-level topology-and-timing signature; (ii) matched edge-deletion augmentation for both static and temporal feature spaces; and (iii) chronological evaluation, fixed-budget ranking, and matched perturbation stress tests over three independent simulations. All labels in this study are assigned by an explicit transaction simulator, and conclusions are limited to the simulated coordinated behaviors.

2. Background and Related Work

2.1. Graph Representation and Dynamic Learning

Random-walk embeddings such as DeepWalk [12] and node2vec [13] provide structural summaries; graph convolution [14], inductive neighborhood sampling [15], graph attention [16], and spectral filtering [17] learn representations from connectivity and features. For evolving graphs, EvolveGCN updates convolutional parameters [18], TGAT encodes time-dependent attention [19], and temporal graph networks employ event-driven memories [20]. Surveys organize these approaches by dynamics, temporal granularity, and learning objective [21, 22]. The present model instead converts fixed-time snapshots into compact numeric features, avoiding end-to-end message-passing training for this moderate-size benchmark.

2.2. Financial Fraud and Collective Behavior

Recent financial and cryptocurrency methods include self-supervised graph embeddings for Bitcoin AML [23], Ethereum-specific pretraining [24], suspicious subgraph modeling [25], convolution-based illicit-transaction classification [26], and local/global attention with pseudo-labels [27]. Feature-gated temporal learning [28] is especially relevant to balancing local descriptors with time-varying graph context. Work on asset-path tracing [29], community-enhanced relations [30], attribute-driven risk representations [31], and homophily/heterophily separation [32] illustrates that suspicious evidence may reside in paths, communities,

node attributes, or heterogeneous links. We examine an intentionally narrower account-level setting and quantify what can be recovered with transparent topology and timing descriptors.

2.3. Imbalance and Evaluation

Graph anomaly and imbalanced-learning surveys note the difficulty of minority labels and shifting structural context [33, 34]. Residual and attribute/structure modeling further motivates retaining raw transaction statistics alongside relational descriptors [35, 36]. GraphSMOTE [37] and focal loss [38] address rare classes in neural pipelines; for the present low-cost design, class-balanced bagging within random forests [39] is sufficient as a comparison choice. With a 9% positive-account rate, we prioritize average precision (AP) rather than relying only on ROC-AUC or accuracy; the relationship of PR and ROC metrics and their interpretation on imbalanced samples are established in [40–42].

Compared with a supervised neural graph architecture, the approach makes an explicit trade-off. GNNs can learn interaction functions from raw adjacency, whereas hand-constructed relational signatures make the timing and degree evidence directly inspectable. As a result, our numerical results compare feature families and a fixed shallow learner, rather than seeking to reproduce the published performance of GNNs on their original datasets. The key question is whether perturbation training changes generalization in the specific simulated data regime, not whether one algorithm is universally superior.

3. Problem Formulation and Method

3.1. Dynamic Transaction Graph

Let $G_t=(V_t, E_t)$ denote the directed, weighted transaction graph during snapshot t, where an event $e=(u, v, \tau, a)$ records sender u, receiver v, within-snapshot time bin $\tau\in\{0, \ldots, 7\}$, and transferred amount $a>0$. Account labels $y_u\in\{0, 1\}$ identify whether the account belongs to a simulator-injected coordination group. Features are computed using only events present in the account's own snapshot; no labels of neighboring accounts enter the representation. The objective is to learn $f(x_u)\approx P(y_u=1)$, with training, validation, and test snapshots disjoint in time. This setting evaluates forward generalization to unseen snapshot-local account cohorts, not prediction of a hidden real-world criminal status.

3.2. Lightweight Temporal Neighborhood Signature

We construct a 21-dimensional vector $x_u=[b_u\|s_u\|q_u]$. Eight account attributes b_u describe logarithmic in/out event counts, logarithmic total and mean amounts, amount variation, and outgoing share. Seven static graph descriptors s_u summarize unique neighbors, reciprocity, repeat interactions, mean neighbor degree, clustering, the maximum counterparty share, and mean shared neighbors. They are calculated on a simple undirected projection when a metric requires undirected neighborhoods; direction and multiplicity are retained for degree and reciprocity measures.

Six temporal descriptors q_u operate on the eight within-snapshot event bins: peak-bin fraction, normalized bin entropy, later-minus-earlier activity, recency-weighted activity, mean cosine agreement between an account's bin histogram and its neighbors' histograms, and the fraction of outgoing events matched by a reverse transfer within one bin. For example, with $h_u(\tau)$ the incident-event count, the burst concentration is $B_u=\max_\tau h_u(\tau)/\max(1, \sum_\tau h_u(\tau))$; the synchronous-neighbor score is $S_u=|N(u)|^{-1}\sum_{v\in N(u)}(h_u\cdot h_v)/(\|h_u\|\|h_v\|+\varepsilon)$, with $S_u=0$ for isolated nodes. These statistics encode group coordination without requiring entity identifiers beyond the observed graph.

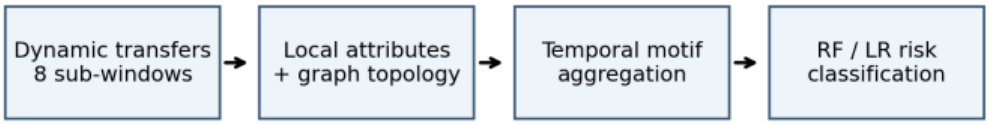


Figure 1. Lightweight temporal graph feature and learning pipeline.

3.3. Temporal Information and Feature Computation

The event histogram is assembled separately for each account and each snapshot. Every directed transfer contributes one event to the sender histogram and one event to the receiver histogram. A zero-degree account receives zero values for denominator-dependent scores by convention. The synchronization measure first computes time-histogram vectors and then averages normalized similarities over observed neighbors. When repeated transfers connect a pair, their timestamps are preserved to calculate the quick-reciprocity score; the simple undirected projection is used only for topology terms such as local clustering. Thus, two graphs with identical aggregated adjacency can still obtain different temporal vectors when their transfer timestamps differ.

In this small-snapshot implementation, topology statistics are computed with NetworkX, and the remaining features with vectorized arrays and incident-edge dictionaries. Histogram and weighted count statistics scale linearly with the event count. The local clustering computation adds cost dependent on neighborhood size; it is not necessary to construct a dense N×N adjacency matrix or update learnable GNN weights. Since all feature extraction precedes classifier fitting, a model can be retrained on a stored numeric matrix without repeatedly traversing the raw graph. This separation also makes the three feature families easy to ablate under identical labels and splits.

3.4. Robust Training Through Partial-Edge Augmentation

Each comparison classifier is a random forest with 110 trees, a minimum leaf size of 3, square-root feature sampling, and balanced-subsample class weights. For each static or temporal augmented forest, we independently discard 15% and 35% of training transaction events to produce two additional feature sets using exactly the same edge deletions. The account labels remain unchanged, and the three representations are concatenated for fitting. Validation and test accounts are never included in this augmented training set. This perturbation mechanism learns from multiple levels of observed evidence rather than relying on a single complete graph.

The decision threshold is chosen separately for each model by maximizing F1 on the unperturbed validation snapshots over 161 thresholds from 0.10 to 0.90. It is then frozen for the final test and every test-only robustness condition. AP and AUROC use the continuous model scores, whereas reported precision, recall, and F1 use the validation-selected threshold. All baselines, including logistic regression, use the same chronological split. Precision at the 100 highest-scored accounts (P@100) measures yield at a fixed 10% review budget among 1,000 held-out accounts. Concept-drift evaluation follows the general premise that the train/test relationship may evolve over time [43].

Formally, let $\varphi(G,u)$ be the 21-dimensional graph signature. For dropout rates $\delta \in \{0, 0.15, 0.35\}$, let $G_t^{(\delta)}$ denote the training snapshot after uniformly removing a fraction δ of its observed edges, with $G_t^{(0)}=G_t$. The fitted forest minimizes the usual tree-ensemble impurity objectives on the pooled examples $\{(\varphi(G_t^{(\delta)}, u), y_u): t \in T_{train}, \delta \in \{0,0.15,0.35\}\}$. Each account contributes one example at each retained evidence level. This is data augmentation, not resampling of labels or modification of the fraud mechanism. It targets robustness to randomly missing observations, not resistance to a fully adaptive adversary that knows the detector.

4. Controlled Experimental Design

4.1. Synthetic Network and Temporal Split

A seeded simulator generates 24 independent account-cohort snapshots, 200 accounts per snapshot, and eight event bins per snapshot. In every snapshot, three groups of six accounts are assigned synthetic-positive labels, yielding 18 positive accounts per snapshot and 432 positives among 4,800 accounts (9.0%). Background transactions are sampled from heterogeneous Gamma-distributed account activity propensities; their amounts follow a lognormal distribution and event times follow a shared nonuniform daily profile. Coordinated groups introduce short cycles, common-recipient transfers, and temporally concentrated cross-transfers. Positive accounts also participate in ordinary background transactions.

In steps 19–23, injected actors switch to fewer repeated ring transfers, distribute some transfers across more time bins, and add ordinary-looking transfers to unrelated accounts. This is a defined pattern shift, not an empirical claim about how actual criminals behave. All tested methods receive the same generated features and labels for each seed. Steps 0–13 constitute training (2,800 accounts), 14–18 validation (1,000 accounts), and 19–23 test (1,000 accounts, 90 positive). The network contains 15,038–15,055 directed events across seeds. The snapshot-local cohort design avoids overlap between account identities in training and testing.

Table I. Generated Benchmark and Evaluation Protocol

Setting	Value
Snapshots / bins	24 / 8
Accounts / positives	4,800 / 432 (9.0%)
Transfers (three seeds)	15,038–15,055
Train / val / test steps	0–13 / 14–18 / 19–23
Test accounts / positives	1,000 / 90
Seeds	11, 23, 37

4.2. Generator Parameters and Reproducibility

The simulator creates 570 candidate background transfers per snapshot before rejecting self-transfers. Account senders and recipients are sampled using propensities drawn from Gamma (shape=1.4, scale=1); amounts are lognormal with log-mean 2.3 and log-standard-deviation 0.85. Transfer bins are sampled from a fixed eight-bin nonuniform distribution. The 18 tagged accounts per snapshot are partitioned into three groups of size six. Early groups use two directed transfer cycles, five hub-bound transfers, and four cross-group-member transfers; late test groups use one directed cycle, five hub-bound transfers, and two cross-member transfers, together with six camouflage transfers to non-group counterparts. The simulator reassigns all group membership and normal traffic independently in each snapshot.

The design deliberately combines unrelated forms of randomness: ordinary account activity, amounts, event times, group membership, and adversarial transaction paths. Changing the random seed modifies each of these sources, while preserving the count of assigned coordinated accounts and the chronology. Background events may involve simulator-positive accounts, preventing positive labels from being defined simply by nonparticipation in ordinary transfers. The transaction records include the generation mechanism tag for auditing the simulator; this tag is never used as a model feature or label source beyond the explicit group assignment. Released deliverables contain the seed-11 synthetic transactions and accounts, with code to regenerate all other seeds.

Table II. Representation Components

Group	Count	Examples
Account attributes	8	flow counts, amount moments
Static graph	7	reciprocity, clustering, concentration
Temporal graph	6	burst entropy, synchronization, rapid reverse

4.3. Baselines, Ablation, And Stress Tests

Six main comparisons isolate feature and augmentation effects: (1) attributes only with a random forest; (2) attributes plus static graph statistics; (3) attributes plus temporal statistics without topology; (4) the full temporal graph representation without augmentation; (5) the static graph representation with matched edge augmentation; and (6) the full temporal graph representation with the same augmentation. A temporal-graph logistic regression supplies a low-capacity comparator. We also remove synchronous-neighbor agreement or the complete temporal block in separate ablations. As an unsupervised historical alternative, isolation forests are a well-established option [44], but are not used here because the target is supervised discrimination of known simulator-injected patterns.

The stress protocol modifies only held-out test edges after model fitting: independent uniform deletion of 20% or 40% of test transactions, or random shuffling of their time-bin assignments while leaving endpoints and amounts intact. The latter retains test topology and destroys event synchronization. No threshold or forest is retuned after perturbation. We report AP, AUROC, validation-threshold F1, and P@100, with means and sample standard deviations over the three seed-specific network generations and training runs; no cross-seed

confidence claims or statistical significance tests are implied.

5. Results and Analysis

5.1. Detection Quality

Table III reports measured scores. Matched training resolves the source of the observed gains. Unaugmented static and temporal forests reach AP 0.571 ± 0.033 and 0.569 ± 0.013, whereas augmentation yields 0.631 ± 0.042 and 0.602 ± 0.012, respectively. Therefore, the strongest tested ranking in this benchmark is the augmented static forest. At a fixed 100-account review budget, its P@100 is 0.537 versus 0.533 for the augmented temporal model; their F1 values are 0.526 and 0.385. The results identify augmentation as the more transferable factor under the imposed time shift, without attributing improvement to the additional timestamp descriptors.

Table III. Chronological Test Results (Mean ± Sd, Three Runs)

Model	AP ± SD	AUC	F1	P@100
Attributes	0.312 ± 0.045	0.837	0.339	0.310
Static graph	0.571 ± 0.033	0.903	0.404	0.503
Static + aug.	0.631 ± 0.042	0.902	0.526	0.537
Temporal only	0.352 ± 0.024	0.849	0.233	0.377
Temporal graph	0.569 ± 0.013	0.894	0.349	0.510
TG + aug.	0.602 ± 0.012	0.908	0.385	0.533
TG + logistic	0.540 ± 0.051	0.899	0.251	0.503

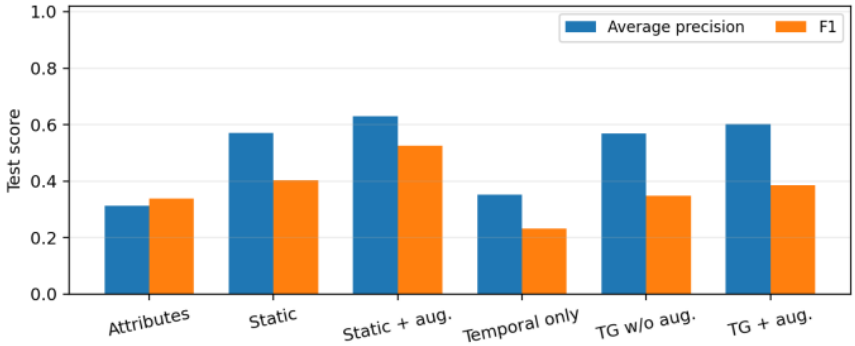


Figure 2. Measured average precision and validation-threshold F1 on future snapshots.

Each run includes the complete chronological split instead of randomly distributing accounts from every snapshot among folds. This matters because the late-period transfer strategy is changed intentionally. The first five validation periods provide a decision threshold that is then carried into future test periods. The AP metric is evaluated as the weighted mean of successive precision values over increases in recall (the non-interpolated average precision implementation), so it summarizes how suspicious accounts are ranked across operating points. The larger AUROC values in Table III must be interpreted jointly with minority-class precision and recall rather than as substitutes for them.

5.2. Missing Transfers and Temporal Disruption

Figure 3 compares stress-test robustness using the same held-out snapshots. At 20% edge deletion, augmented static and temporal AP are 0.518 and 0.495; at 40% they are 0.389 and 0.367. The augmented static forest ranks highest under both deletions. Its fixed-threshold F1 at 40% deletion is 0.219; hence, ranking retention and threshold stability remain different operational properties.

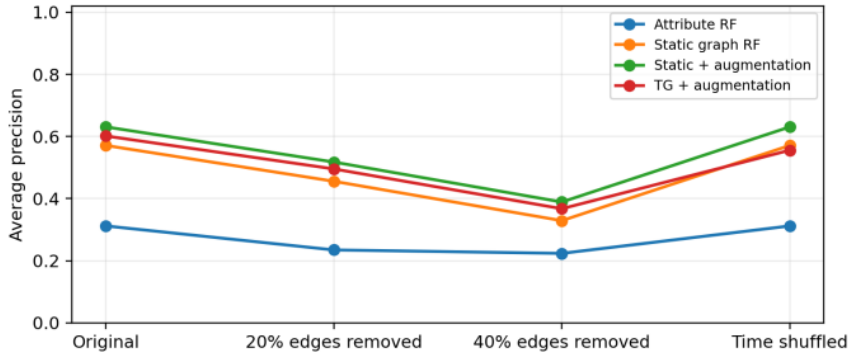


Figure 3. Matched held-out AP under transaction deletion and timestamp shuffling.

Table IV. Ap Under Held-Out Perturbation (Mean of Three Runs)

Model	Intact	−20%	−40%	Shuffle
Attributes	0.312	0.235	0.224	0.312
Static graph	0.571	0.456	0.329	0.571
Static + aug.	0.631	0.518	0.389	0.631
TG + aug.	0.602	0.495	0.367	0.556

For timestamp shuffling, the augmented temporal graph model changes from AP 0.602 to 0.556; the augmented static graph remains at 0.631 because its features ignore timestamps. This controlled intervention provides direct evidence that the added temporal descriptors are used by the fitted classifier. It does not identify which particular transfer mechanism would be most prevalent in a real payment service.

Table IV is a test-time intervention table: all rows use identical fitted models and validation-selected thresholds to avoid allowing re-optimization on deliberately corrupted test

data. Attribute features change when edges are removed because counts and amounts are recomputed from observed events; therefore even the attribute-only model is sensitive to deletion. Conversely, time shuffling does not alter its input quantities. In this sense, the perturbations isolate different failure modes rather than supplying independent alternative datasets. The test labels are kept fixed by the simulator, and unobserved transactions are never treated as ground truth negatives.

5.3. Ablation, Temporal Consistency, And Cost

Removing the synchronous-neighbor feature from the unaugmented full model produces AP 0.579, while deleting all temporal descriptors produces 0.571. Relative to the unaugmented full model (0.569), the effect of the single synchronization feature is not uniformly positive under shifted test patterns. Under matching augmentation, the static representation performs better than the full temporal representation on pooled held-out AP; the synchronization feature therefore does not supply a general gain in this simulator. This is compatible with a useful time-shuffling sensitivity test but cautions against conflating feature sensitivity with improved ranking.

Table V. Feature Ablation on Future Snapshots

Configuration	AP (mean ± SD)	F1 (mean ± SD)
TG without augmentation	0.569 ± 0.013	0.349 ± 0.100
Remove synchronization	0.579 ± 0.001	0.392 ± 0.080
Remove all temporal	0.571 ± 0.033	0.404 ± 0.107
Static with augmentation	0.631 ± 0.042	0.526 ± 0.080
TG with augmentation	0.602 ± 0.012	0.385 ± 0.033

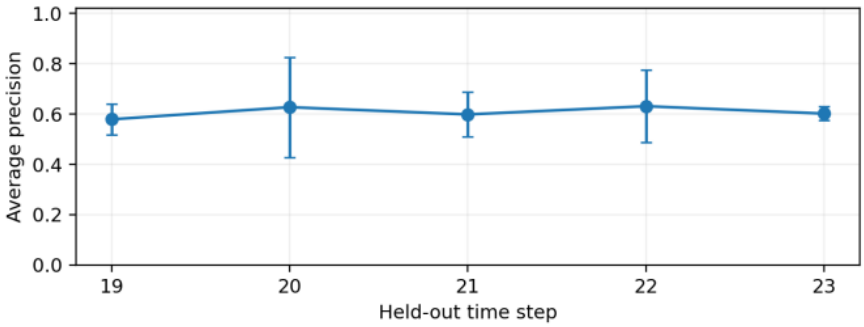


Figure 4. AP by held-out time step for the augmented model; error bars show across-seed SD.

Figure 4 presents separate results for steps 19–23 and makes the temporal evaluation inspectable rather than reducing it to one pooled number. End-to-end generation, feature computation, training, and stress evaluation required a measured mean of 12.3 s per independent run in the execution environment, with three complete runs and multiple forest fits; this is a descriptive implementation timing rather than a hardware-normalized benchmark. The system uses NumPy, pandas, NetworkX, and scikit-learn, and does not require a GPU.

The dataset generator and classifier are deterministic given a seed. This permits a clear audit trail from individual transfer records through the extracted attributes to per-seed metrics. The results directory includes the per-period scores, robust-test records, model summary, feature importance, runtime, and dataset counts. For example, mean AP 0.602 is calculated by averaging AP on the same 1,000 held-out accounts for seeds 11, 23, and 37; it is not a value copied from an external study. The three seed results represent a small controlled replication set, so standard deviations describe observed simulator variability rather than uncertainty over all possible financial networks.

6. Discussion

Three observations emerge. First, static structure adds substantial ranking information over attributes alone (AP 0.571 versus 0.312). Second, augmentation improves both matched static and temporal variants (from 0.571 to 0.631 and from 0.569 to 0.602), but adding timing descriptors after augmentation does not further improve average AP. Third, both augmented variants retain more AP under randomly missing test edges than their nonaugmented counterparts, although validation-threshold F1 remains vulnerable to corruption [42-50].

The controlled benchmark gives known ground truth for group coordination and isolates topology, timing, and missing-edge mechanisms. It is intentionally distinct from Elliptic, DGraph, or real banking systems: it does not measure financial loss, verified fraud cases, cross-platform identity linkage, or deployment false-positive costs. Its account cohorts do not persist across snapshots, and temporal learning is limited to within-snapshot event timing plus chronological generalization. Thus, the reported performance describes detection of injected coordinated behaviors, not real-world anti-money-laundering effectiveness [51-59].

For a deployment-oriented extension, this approach can be transferred to real timestamped account-to-account payments with compliance-confirmed labels, rolling windows, and account persistence. The same chronology-aware protocol and edge-removal evaluation would remain appropriate; feature normalization, investigation capacity, distribution drift, and threshold recalibration would require system-specific review. The present work provides a small-cost benchmark and empirical evidence about the ranking behavior of compact temporal graph features under controlled changes.

A ranking-first interpretation is also important for operational use. At a chosen review budget, an institution usually inspects the highest-scoring accounts rather than acting on every thresholded positive. Average precision describes this score ordering across all recall levels. In contrast, F1 reflects one threshold selected on a specific validation period and may change substantially after missing transfers alter the score distribution. Here, both augmented feature variants improve AP over their own nonaugmented counterparts under missing-edge conditions, but neither dominates every metric; these observations recommend reporting both ranking and fixed-threshold performance when

examining robustness. A deployed system would additionally need calibrated probabilities, reviewer capacity constraints, and cost-weighted decision criteria, none of which are inferred from the simulator.

References

[1] Feng, E., Li, P., Mo, T., & Wang, G. (2022). Prism: Prediction-Guided Risk-Aware Deep Reinforcement Learning for Energy-Aware Dynamic Resource Scheduling in Distributed Cloud Data Centers. Journal of Computer Science and Electrical Engineering, 4(1), 53–61.

[2] Li, P., Feng, E., Wang, G., & Mo, T. (2024). Hybrid Deep Reinforcement Learning and Integer Optimization for Resource Scheduling In Heterogeneous Ai Computing Clusters. Innovation and Technology Studies, 1(1), 48–56.

[3] Teng, D., Rhee, M., Qin, Y., Zi, B., & Liu, W. (2026). SW-SpeedDLM: Sliding Window Speculative Decoding for Diffusion Language Models Under Long Context Constraints. Mathematics, 14(12), 2137.

[4] 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.

[5] Chen, J., Liang, Y., Liu, J., & Zhou, M. (2026). Temporal Transformer with Conditional Tabular GAN for Credit Card Fraud Detection: A Sequential Deep Learning Approach. Mathematics, 14(7), 1183.

[6] Mo, T., Wang, G., Feng, E., & Li, P. (2023). Execution-Feedback-Guided Context Selection for Large Language Model-Based Software Testing and Debugging. World Journal of Information and Knowledge Management, 1(1), 55–64.

[7] Zhu, R., & Yue, X. (2024). An explainable machine learning framework for predicting dividend sustainability and financial risk of US REITs under interest-rate shocks. Journal of Trends in Finance and Economics, 1(2), 48–57.

[8] Mo, T., Zhang, C., Zou, J., Guo, Z., & Rhee, M. (2026). Self-Evolving AI Agents with Dual Memory for Automated Software Testing and Bug Localization. IEEE Access.

[9] Liang, Y., Jiao, Y., Ping, W., Fan, H., & Han, X. (2026). Adaptive event-driven labeling: A neuro-symbolic multiagent framework for causal inference in non-stationary time series. IEEE Access.

[10] Shen, A., & Huang, S. (2022). A Systems-Aware Framework for Intelligent Large-Scale Recommendation Models. Journal of Computer Science and Electrical Engineering, 4(1), 36–44.

[11] Zou, J., Qin, Y., & Rhee, M. (2024). Beyond model scale: A task-aware reliability benchmark for large language models in clinical decision support. Innovation and Technology Studies, 1(1), 39–47.

[12] 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.

[13] Teng, D. (2025). Rsrbench: Reproducibility and Performance Benchmarking for Regional Seismic Risk Workflows. Academic Journal of Architecture and Civil Engineering, 3(1), 40–52.

[14] Rhee, M., Zou, J., & Qin, Y. (2024). CLARITY: A closed loop framework for duplicate aware failure prediction and budgeted automated remediation in cloud infrastructure. AI and Data Science Journal, 1(1), 76–85.

[15] Mai, N. T., Cao, W., & Fang, Q. (2025). A study on how LLMs (eg GPT-4, chatbots) are being integrated to support tutoring, essay feedback and content generation. Journal of Computing and Electronic Information Management, 18(3), 43–52.

[16] Zhou, F. (2026). Moment-Constrained Side-Force Allocation for Wind-Assisted Merchant Ships: Hydrodynamic Performance of Retractable Anti-Leeway Fins and Flap Rudders. Innovation and Technology Studies, 3(1), 27–41.

[17] Chen, J., Liu, J., Liang, Y., & Zhou, M. (2026). HeteroGCL: A Heterogeneous Graph Contrastive Learning Framework for Scalable and Sustainable Cryptocurrency AML. Applied Sciences, 16(6), 2860.

[18] Chen, J., Han, X., Zhang, X., & Yang, Y. (2023). Exact Concept-Level Explanations for Early Detection of Financial Misstatements. Eurasia Journal of Science and Technology, 5(2), 34–48.

[19] Liang, Y., & Jiao, Y. (2024). Real-time financial fraud detection using a feature-light temporal dynamic graph neural network with contrastive learning. Journal of Trends in Finance and Economics, 1(1), 76–84.

[20] Zhang, C., & Wang, Y. (2024). A Data-Driven Multi-Objective Production Scheduling Framework for Battery Pack Manufacturing Under Material Arrival Uncertainty. AI and Data Science Journal, 1(1), 86–93.

[21] Qin, Y., Zou, J., & Rhee, M. (2022). An AI-driven intelligent transaction execution framework for internet finance platforms: Cost-sensitive real-time integrity interdiction. Eurasia Journal of Science and Technology, 4(2), 36–44.

[22] Jiao, Y., Fan, H., Yue, X., Ping, W., Sun, T., & Wang, J. (2026). Dynamic heterogeneous graph contrastive learning for uncovering collusive financial fraud. Scientific Reports.

[23] Shen, A. (2023). Compute-Efficient Scaling of Large Sequential Recommendation Models Under Dynamic User Behavior. World Journal of Information Technology, 1(1), 62–71.

[24] Fang, Q. (2024). Bias-Aware Large Language Models for Equitable Educational Feedback and Assessment. Trends in Social Sciences and Humanities Research, 2(1), 41–50.

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

[26] Shang, W., & Chen, T. (2024). PrefixKD-Edge: Real-time locomotion-intent prediction with prefix self-distillation on lightweight temporal networks. Innovation and Technology Studies, 1(1), 30–38.

[27] Bian, M., Wang, Y., & Lin, Y. (2024). Explainable Supplier Risk Prediction for Critical Telecommunications Infrastructure Procurement: An Empirical Xgboost–Shap Framework. Social Science and Management, 1(1), 64–77.

[28] 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.

[29] Cao, J., Zheng, W., Ge, Y., & Wang, J. (2025). DriftShield: Autonomous fraud detection via actor-critic reinforcement learning with dynamic feature reweighting. IEEE Open Journal of the Computer Society, 6, 1166–1177.

[30] Guo, Z., Chen, T., & Shang, W. (2022). Class-conditional intermediate-domain adversarial adaptation for cross-device acoustic scene classification. Journal of Computer Science and Electrical Engineering, 4(1), 26–35.

[31] 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.

[32] Wang, Y., Bian, M., & Lin, Y. (2022). An Explainable Temporal Graph Neural Network for Disruption Prediction

And Risk Propagation in Multi-Tier Supply Chains. Journal of Computer Science and Electrical Engineering, 4(1), 17–25.

[33] Lin, Y., Wang, Y., & Bian, M. (2024). Xor-Ew: An Explainable and Cost-Aware Artificial Intelligence Framework for Operational Risk Early Warning In Financial Institutions. AI and Data Science Journal, 1(1), 94–102.

[34] 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.

[35] Fang, Q., & Liu, W. (2025). HARLA-ED: Resolving Information Asymmetry and Enhancing Algorithmic Symmetry in Intelligent Educational Assessment via Hybrid Reinforcement Learning. Symmetry, 18(1), 58.

[36] 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.

[37] Chen, T., Shang, W., & Guo, Z. (2024). Xsla: An SLA-aware heterogeneous XPU scheduling framework for edge AI inference. World Journal of Information Technology, 2(2), 48–59.

[38] Lin, Y., Yue, X., Bian, M., Zhu, R., & Teng, D. (2026). Subgraph-level temporal graph transformer with laundering-motif prototypes for anti-money laundering on blockchains. IEEE Access.

[39] 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.

[40] Ping, W., & Fan, H. (2023). Beyond Isolated Ratios: Accrual–Revenue–Cash Flow Indicators and Interaction-Aware Learning for Detecting Financial Reporting Irregularities in U.s. Public Companies. World Journal of Economics and Business Research, 1(1), 46–55.

[41] Wang, Y., & Zhang, C. (2023). Aeris: Adaptive Executor-Scoped Resource Management for Privacy-Preserving Data Processing In Apache Spark. Eurasia Journal of Science and Technology, 5(2), 25–33.

[42] Chen, J., Zhang, X., & Han, X. (2023). Explainable Identification of Tax-Related Financial Reporting Anomalies In U.s. Public Companies: Cross-Fitted Tax-Consistency Residuals and Owen-Value Shapley Propagation. World Journal of Economics and Business Research, 1(1), 33–45.

[43] Shen, A., & Liu, W. (2026). Budget-Aware Adaptive Test-Time Compute with Hierarchical Self-Verification for Semantic ID Generative Recommendation. Computers.

[44] Fan, H., Liang, Y., Jiao, Y., Ping, W., & Lin, Y. (2026). An Explainable Temporal Machine Learning Framework for Financial Fraud Risk Prediction in Listed Institutions. Highlights in Business, Economics and Management, 70, 173–185.

[45] Fang, Q. (2024). True-Ews: Trustworthy Early Warning with Equalized Conformal Guarantees for Equitable Student Support In U.s. Education. Educational Research and Human Development, 1(2), 44–54.

[46] 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.

[47] Zhang, F., & Yang, J. (2024). Robustness-Aware Model Compression for Multimodal Perception Under Challenging Conditions. AI and Data Science Journal, 1(1), 103–111.

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

[49] 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.

[50] Zhang, F., & Yang, J. (2022). Accuracy-Aware Model Compression for Real-Time Visual Perception Under Resource Constraints. Journal of Computer Science and Electrical Engineering, 4(1), 45–52.

[51] Jiao, Y., & Liang, Y. (2024). An Integrated Process-Mining and Explainable Machine-Learning Framework for Bottleneck And Internal-Control Risk Detection in the Multi-Entity Financial Close. Social Science and Management, 1(3), 96–107.

[52] Teng, D. (2025). Triad: Deadline- And Reliability-Aware Scheduling for Post-Earthquake Structural Assessment Workflows. World Journal of Engineering Research, 3(6), 55–66.

[53] Zhang, F., Shen, A., & Huang, S. (2026). TRIAD: Co-designing mixed-precision quantization, operator fusion, and on-chip memory reuse for real-time vision–language–action inference on automotive NPUs. Results in Engineering, 113673.

[54] Qin, Y., Rhee, M., Teng, D., Zhang, C., & Zi, B. (2026). Neuro-symbolic constraint verification for LLM-driven internet finance transaction execution. Scientific Reports.

[55] Wang, G., Li, P., Feng, E., Shen, A., & Rhee, M. (2026). A dual-process memory-augmented large language model agent for explainable sequential recommendation. IEEE Access.

[56] 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.

[57] Wang, G., Li, P., Feng, E., & Mo, T. (2024). Uncertainty-Aware Llm-As-A-Judge Evaluation for Reliable Search and Recommendation Systems: Calibrated Confidence and Impact-Aware Selective Human Review. AI and Data Science Journal, 1(1), 112–121.

[58] Guo, Z., Fang, Q., Yang, J. S., & Rhee, M. (2026). Closed-loop audio neuroregulation for ADHD: Graph-based attention decoding and rhythm entrainment. IEEE Access.

[59] Mai, N. T., Fang, Q., & Cao, W. (2025). Measuring student trust and over-reliance on AI tutors: Implications for STEM learning outcomes. International Journal of Social Sciences and English Literature, 9(12), 11–17.