Journal of Trends in Finance and Economics
Print ISSN: 3007-6951
Online ISSN: 3007-696X
DOI: https://doi.org/10.61784/jtfe4076

REAL-TIME FINANCIAL FRAUD DETECTION USING A FEATURE-LIGHT TEMPORAL DYNAMIC GRAPH NEURAL NETWORK WITH CONTRASTIVE LEARNING

YiWen Liang[1*], YaNan Jiao[2]
[1]*Cornell University, Ithaca 14853, NY, USA.*
[2]*Long Island University, Brooklyn 11201, NY, USA.*
[*]*Corresponding Author: YiWen Liang*

Abstract: Financial fraud detection is a rare-event, non-stationary graph classification problem in which relational patterns can change faster than labels become available. This paper presents FT-TDGCL, a feature-light temporal directed graph neural network with contrastive learning for near-real-time anti-money-laundering screening. Unlike approaches that depend on the 166 anonymized Elliptic attributes, the proposed pipeline uses only the public directed transaction edge list and labels. It derives 18 auditable topology features, applies incoming and outgoing graph filters at one and two hops, and fuses a node embedding with a causal gated recurrent snapshot context. During training, two independently channel-masked and noise-perturbed graph-filtered views are aligned with an NT-Xent objective in addition to class-balanced binary cross-entropy. We evaluate on the real Elliptic Bitcoin graph with 203,769 transactions, 234,355 payment flows, and 49 temporal steps. A strict chronological split uses steps 1–29 for training, 30–34 for validation and threshold selection, and 35–49 for testing; unknown labels participate only in topology construction. Across three seeds, FT-TDGCL obtains a test PR-AUC of 0.2152±0.0935 versus 0.1655±0.0333 without contrastive learning, a 30.0% relative gain. It reaches recall 0.5429±0.0444, while its F1 of 0.2344±0.0393 does not surpass every fixed-seed baseline, exposing a calibration trade-off rather than a universal improvement. Cached-feature inference over 16,670 labeled test nodes requires 6.95±0.64 ms on CPU. All metrics, predictions, hashes, and code are generated by the accompanying reproducibility package; no synthetic observations or hand-entered experimental results are used.
Keywords: Financial fraud detection; Anti-money laundering; Temporal graph neural network; Contrastive learning; Class imbalance; Elliptic Bitcoin dataset

1 INTRODUCTION

Fraud screening is usually deployed as a ranking system rather than as a balanced classification exercise. Only a small fraction of transactions are illicit, labels arrive after investigation, and the cost of sending every weak signal to a human analyst is prohibitive. These properties make accuracy and even ROC-AUC potentially misleading; precision-recall analysis is more informative under severe imbalance [1-2]. In addition, a transaction cannot be assessed in isolation. Its risk depends on incoming sources, outgoing dispersal, intermediary position, and how these motifs change across time. Graph neural networks (GNNs) offer a natural representation for this relational evidence, yet a practical anti-money-laundering (AML) system must also respect direction, chronology, latency, and confidentiality [3-6].

The Elliptic data set is a rare public benchmark that maps Bitcoin transactions to a directed payment-flow graph and supplies licit, illicit, and unknown labels [7]. The original release also contains 166 anonymized attributes, including features derived from non-public information. Those attributes are useful for benchmarking but complicate independent auditing, feature governance, and deployment in institutions that do not possess the same proprietary signals. This paper therefore asks a deliberately harder and operationally relevant question: how much fraud-ranking capability can be recovered from topology and temporal evolution alone?

We answer this question with FT-TDGCL, a feature-light temporal directed graph contrastive learning model. The model first computes interpretable structural descriptors and fixed directed graph filters. It then combines a node representation with a unidirectional gated recurrent unit (GRU) context summarizing the current and preceding snapshots [8]. Finally, it regularizes the encoder through view consistency: two stochastic masks and small perturbations are applied to graph-filtered feature channels, and corresponding node embeddings are attracted with an NT-Xent objective inspired by contrastive representation learning [9-15]. The design keeps sparse propagation outside the trainable network, reducing inference cost and allowing each component to be audited.

The term real-time is used carefully in this work. The Elliptic graph is released as 49 discrete temporal steps, not as a continuous transaction stream. Our claim is therefore near-real-time snapshot screening: once a snapshot and its directed edges are available, topology features can be updated and the trained classifier can score nodes with millisecond-scale cached-feature latency. We do not claim end-to-end exchange ingestion latency or immediate ground-truth feedback.

The contributions are fourfold. First, we formulate a topology-only, chronologically evaluated AML task that does not require the large anonymized feature matrix. Second, we introduce a directed multi-scale encoder with causal snapshot

context and gated fusion. Third, we add channel-level contrastive consistency to improve rare-class ranking under temporal drift. Fourth, we release a reproducibility package that stores every seed-level metric, every test probability, temporal diagnostics, data-file hashes, and CPU timings. The empirical evidence is intentionally qualified: contrastive learning improves mean PR-AUC and recall but not every thresholded metric, and variation across seeds remains material.

2 RELATED WORK

2.1 Static and Directed Graph Learning

GCN, GraphSAGE, graph attention networks, and simplified graph convolution (SGC) established complementary mechanisms for aggregating neighborhood evidence [3-6]. Standard symmetric normalization is convenient for citation or social graphs but can blur source and destination roles in transaction networks. A payment entering a node and a payment leaving it have different forensic meanings; consequently, FT-TDGCL preserves incoming and outgoing transition operators and their two-hop powers. The approach resembles SGC in moving propagation before the trainable classifier, but it retains five directed channels rather than one undirected low-pass filter.

2.2 Dynamic and Temporal Graph Models

Dynamic graph learning has evolved along snapshot-based and event-based lines. EvolveGCN evolves GCN parameters through recurrent units, while DySAT applies structural and temporal self-attention [16-17]. TGAT represents embeddings as functions of continuous time [18]; Temporal Graph Networks combine memory, message functions, and time-aware aggregation for event streams [19]. DyRep models association and communication events, and JODIE predicts dynamic trajectories for interacting entities [20-21]. These methods are expressive, but reproducing them fairly can require event timestamps, dense attributes, or expensive temporal neighborhoods unavailable in the feature-light setting. FT-TDGCL instead adopts a compact snapshot summary GRU, which is causal and inexpensive while still adapting the decision representation to temporal context.

2.3 Contrastive Learning and Fraud Detection

Graph contrastive methods construct related views and maximize their representation agreement. GraphCL studies graph augmentations [9]; GRACE contrasts node representations under feature and edge corruption [10]; MVGRL contrasts diffusion and neighborhood views [11]; GCC learns transferable structural representations through subgraph instance discrimination [12]. SimCLR, supervised contrastive learning, and contrastive predictive coding provide the broader foundations for temperature-scaled discrimination [13-15]. Our use of contrastive learning is narrower: it is a supervised-training regularizer over topology-derived channels, not a claim of large-scale unsupervised pretraining. The two views preserve node identity and temporal context while perturbing individual filtered channels.

Fraud-specific GNNs address camouflage and imbalance. CARE-GNN selects informative neighbors against feature and relation camouflage, while PC-GNN combines label-balanced sampling with neighbor selection [22-23]. DOMINANT reconstructs attributes and topology for graph anomaly detection, and the broader graph anomaly literature stresses that anomalous behavior may be contextual rather than globally rare [24-25]. FT-TDGCL differs by targeting a directed temporal transaction graph with no original node attributes. Class imbalance is handled with class-balanced binary cross-entropy; focal loss is relevant but was not used, because the primary goal was to isolate the contribution of temporal context and contrastive regularization [26].

3 PROBLEM FORMULATION AND DATA

3.1 Temporal Directed Fraud Classification

Let G_t=(V_t,E_t) denote the directed transaction graph at temporal step t. A node v_i represents a Bitcoin transaction and an edge (v_i,v_j) indicates that output from transaction i is an input to transaction j [7]. For labeled nodes, y_i∈{0,1} denotes licit or illicit status. Unknown nodes have no supervised target but remain in the graph because they mediate flows and therefore affect topological neighborhoods. Given snapshots up to t, the objective is a causal score p_i^t=P(y_i=1|G_1,…,G_t). Model selection and threshold tuning must use only earlier validation steps; random node mixing would leak future structural regimes into training.

$$p_i^t = f_\theta(v_i, G_1, \ldots, G_t), \text{ with } \frac{\partial p_i^t}{\partial G_{t+k}} = 0 \text{ for } k > 0 \quad (1)$$

3.2 Exact Temporal Reconstruction

The downloaded class file contains transaction identifiers and labels, while the edge file contains directed transaction pairs. We did not use or impute the 166-dimensional feature matrix. An integrity procedure maps identifiers to class-file row indices, constructs an undirected union-find structure over the directed edges, and sorts connected components by their minimum row index. The real files yield exactly 49 components; each component occupies a contiguous,

non-overlapping row block, and the blocks partition all 203,769 nodes without gaps. These checks recover the 49 temporal steps from the released topology and row order. No synthetic timestamp is generated. The script raises an exception if the component count or contiguity conditions fail.

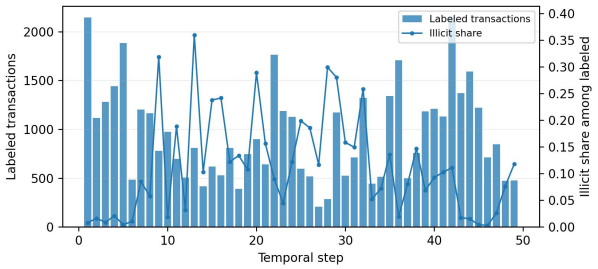

Figure 1 Actual Labeled Volume and Illicit Share Across the 49 Reconstructed Temporal Steps
Note: The large variation motivates chronological evaluation and PR-AUC reporting.

Figure 1 shows substantial label and prevalence drift. For example, the illicit share among labeled nodes exceeds 0.35 in step 13 but falls below 0.01 in several late steps. This variability means that a validation-selected threshold can become miscalibrated even when the underlying ranking remains useful. It also explains why a single aggregate F1 score should not be interpreted as a complete deployment verdict, see Table 1.

Table 1 Real Elliptic Data and Chronological Split

Item	Value
Transactions / directed edges	203,769 / 234,355
Temporal steps	49
Illicit / licit / unknown	4,545 / 42,019 / 157,205
Train: steps 1–29	26,381 labeled nodes
Validation: steps 30–34	3,513 labeled nodes
Test: steps 35–49	16,670 labeled nodes
Supervised use of unknown labels	None; topology only
Input files	class labels + directed edge list

4 FT-TDGCL METHOD

4.1 Auditable Topology Features

For each node, the base vector x_i contains 18 values: log in-degree, log out-degree, log total degree, source and sink indicators, log out/in ratio, mean predecessor and successor log-degree, two-hop fan-in and fan-out counts, PageRank, normalized forward and reverse DAG depth, fan-in and fan-out indicators, log snapshot size, log snapshot edge density, and normalized temporal position [27]. PageRank and DAG depths are computed separately within each snapshot. The features are inexpensive, interpretable, and available from the edge list alone. They do not encode labels or future snapshots.

4.2 Multi-Scale Directed Graph Filters

Let A_t be the adjacency matrix whose row i contains outgoing edges from node i. We define row-normalized outgoing and incoming transitions P_out=D_out^{-1}A_t and P_in=D_in^{-1}A_t^T, with zero rows for isolated directions. The graph-filtered input concatenates the original features with one- and two-hop incoming and outgoing averages:

$$h_i^{(0)} = \left[X, P_{\text{in}}X, P_{\text{out}}X, P_{\text{in}}^2X, P_{\text{out}}^2X\right]_i \in R^{90} \quad (2)$$

This channelization retains direction and exposes different receptive-field scales to a small multilayer perceptron. Unlike a trainable message-passing stack, the sparse products are cached, so model training and scoring do not repeatedly traverse the full graph. Standardization parameters are fitted on labeled training nodes only.

4.3 Causal Temporal Context and Gated Fusion

For snapshot t, a summary s_t concatenates the mean and standard deviation of the 18 base topology features across all nodes in V_t. A unidirectional GRU maps s_1,…,s_t to context c_t. Because the GRU is not bidirectional, c_t cannot use future summaries. The 90-dimensional filtered node input is encoded by a two-layer MLP with LayerNorm and GELU into z_i∈R^64. The 24-dimensional GRU state is projected to 64 dimensions and fused through a learned gate:

$$g_i = \sigma\left(W_g[z_i \,\|\, W_c c_t] + b_g\right), \quad \tilde{z}_i = g_i \odot z_i + (1 - g_i) \odot W_c c_t \quad (3)$$

A dropout-linear head produces the fraud logit. The gate permits the model to rely on stable local topology when the snapshot context is uninformative and to shift toward contextual evidence when the graph regime changes, see Figure 2.

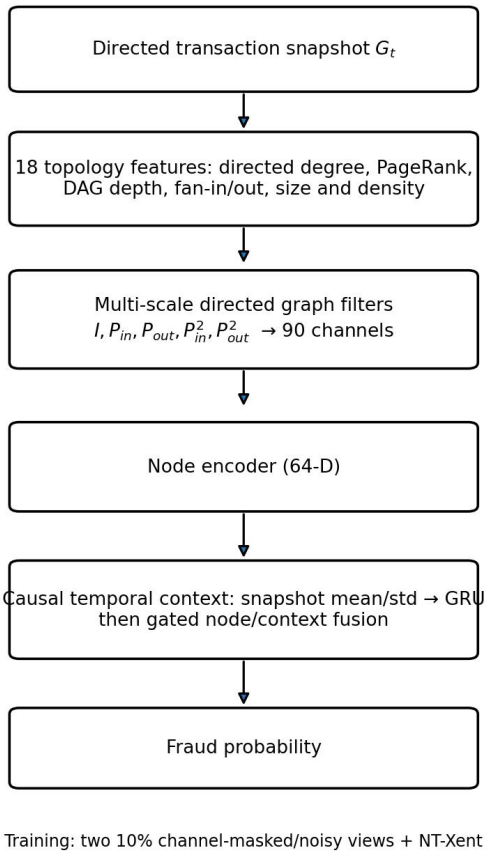

Figure 2 FT-TDGCL Architecture

Note: Graph filtering is precomputed; the temporal branch is causal. Contrastive perturbations are used only during training.

4.4 Contrastive View Consistency

Two views of each labeled training node are created from the standardized 90-channel vector. Each view independently masks 10% of channels and adds zero-mean Gaussian noise with standard deviation 0.01. This augmentation deliberately avoids claiming an unmeasured edge-drop process: it perturbs the outputs of directed graph filters rather than rebuilding a second graph. Both views share the node encoder and the same causal temporal context. For a mini-batch B, normalized embeddings from corresponding views form positive pairs and all other nodes in the batch act as negatives:

$$L_{\mathrm{CL}} = -\frac{1}{2|B|}\sum_i \left[\log \frac{\exp\left(\frac{\mathrm{sim}(u_i, v_i)}{\tau}\right)}{\sum_j \exp\left(\frac{\mathrm{sim}(u_i, v_j)}{\tau}\right)} + \text{symmetric term} \right] \tag{4}$$

The supervised loss is class-balanced binary cross-entropy with positive weight N_neg/N_pos. The total objective is

$$L = L_{\mathrm{BCE}} + \lambda L_{\mathrm{CL}}, \text{where } \lambda = 0.12 \text{ and } \tau = 0.25 \tag{5}$$

The contrastive batch size is 384. The augmentation and NT-Xent term are disabled at validation and test time, so they add no inference latency.

4.5 Training and Online Scoring

Algorithm 1 summarizes the pipeline. The graph and time reconstruction are deterministic; only neural initialization, dropout, mini-batch contrastive sampling, and augmentation masks depend on the seed. In a deployment implementation, snapshot topology descriptors and sparse directed filters can be incrementally updated as a block closes. The measured implementation recomputes and caches them from the released CSV files for auditability.

Algorithm 1. Feature-Light Temporal Fraud Screening

1) Verify the two input schemas and SHA-256 hashes, then map transaction identifiers.
2) Recover the 49 contiguous temporal components and stop if any integrity check fails.
3) Compute 18 topology descriptors and five directed one-/two-hop filter channels.
4) Fit standardization parameters using labeled training nodes from steps 1–29 only.
5) Train the encoder, causal GRU, gate, and classifier with weighted BCE and NT-Xent.
6) Select the checkpoint and decision threshold using validation steps 30–34 only.
7) Score steps 35–49 and export probabilities, metrics, timings, and diagnostics.

4.6 Complexity and Incremental Deployment

Let n_t and m_t denote the numbers of nodes and edges in snapshot t, d=18 the base feature dimension, and K=2 the maximum propagation depth. Degree, fan, and depth descriptors are linear in n_t+m_t for the acyclic transaction components, while PageRank is O(I(m_t+n_t)) for I power iterations. The five sparse graph-filter blocks require O(Km_td) arithmetic and are computed once per snapshot. After caching, the trainable scorer is independent of m_t: it evaluates an MLP, one snapshot-level GRU update, a gate, and a linear head. This separation is the main reason cached inference is fast on CPU.

A practical service can maintain degree counters and predecessor/successor aggregates as edges arrive, then finalize PageRank and depth features in a short micro-batch when a ledger block or institution-defined window closes. The GRU state is carried forward rather than recomputed from future windows, preserving causality. Because training views perturb cached channels instead of deleting live edges, deployment uses one deterministic feature vector per transaction. Model version, standardization statistics, temporal state, and threshold should be logged together so that an alert can be reproduced.

5 EXPERIMENTAL PROTOCOL

5.1 Baselines and Ablations

Four feature-light baselines are evaluated: class-balanced logistic regression and random forest, a topology-only MLP, and Directed SGC using the 90 filtered channels [28]. The temporal ablation uses the same directed filters, node encoder, GRU, gate, and training procedure as FT-TDGCL but removes L_CL. Logistic regression, random forest, MLP, and Directed SGC use the fixed reference seed 7; the two principal temporal variants are repeated with seeds 7, 19, and 43. This compute-efficient asymmetry is reported explicitly and fixed-seed rows are marked in the result table. Multi-seed uncertainty is reserved for the central temporal and contrastive ablation, so no uncertainty claim is made for the fixed-seed baselines.

5.2 Chronology, Metrics, and Thresholds

Training uses steps 1–29, validation uses 30–34, and test uses 35–49. Unknown labels are excluded from supervised losses and all metrics. Early stopping maximizes validation average precision with patience 7 over at most 50 epochs. The classification threshold maximizes validation F1 and is then frozen for the entire test period. We report PR-AUC as the primary measure, together with ROC-AUC, F1, precision, recall, training time, classifier forward-pass latency, and throughput. PR-AUC is emphasized because it directly reflects the trade-off between alert precision and fraud coverage under imbalance [1-2].

5.3 Implementation and Reproducibility

The experiment runs on CPU with Python 3.13.5, PyTorch 2.10.0+cpu, NumPy 2.3.5, SciPy 1.17.0, pandas 2.2.3, and scikit-learn 1.8.0 [29]. AdamW uses learning rate 0.002, weight decay 10^{-4}, hidden dimension 64, temporal dimension 24, dropout 0.25, and gradient norm clipping at 5. The package records data hashes: 93e2e7b2405c735b… for the class file and a35053ba68a98e43… for the edge file [30]. The complete hashes are stored in dataset_stats.json. Metrics are calculated by code from saved probabilities; the manuscript does not contain invented or manually adjusted measurements.

6 RESULTS AND ANALYSIS

6.1 Aggregate Detection Performance

Table 2 Test Performance on Steps 35–49

Model	PR-AUC	ROC-AUC	F1	Recall
LR†	0.170	0.726	0.212	0.480
RF†	0.164	0.735	0.238	0.335
MLP†	0.153	0.749	0.253	0.298
Dir-SGC†	0.186	0.720	0.245	0.495
TDGNN-noCL	0.165±0.033	0.716±0.046	0.236±0.050	0.503±0.066
FT-TDGCL	0.215±0.094	0.726±0.059	0.234±0.039	0.543±0.044

Note: † Fixed seed 7. Temporal variants report mean±standard deviation across seeds 7, 19, and 43.

FT-TDGCL achieves mean PR-AUC 0.2152, compared with 0.1655 for the identical temporal network without contrastive learning, see Table 2. The absolute gain is 0.0497 and the relative gain is 30.0%. Against the strongest fixed-seed non-temporal PR-AUC baseline, Directed SGC at 0.1861, the mean gain is 15.6%. These comparisons support the main hypothesis that consistency across perturbed graph-filtered views can improve ranking when only topology is available.

The result is not uniformly dominant. FT-TDGCL has F1 0.2344±0.0393, below the fixed-seed MLP value 0.2533 and Directed SGC value 0.2453. Its advantage is instead concentrated in recall: 0.5429±0.0444, higher than the reported fixed-seed baselines. Thus, contrastive regularization produces a more aggressive ranking that captures more illicit transactions but creates additional false alerts at the threshold selected on earlier validation steps. For AML triage, that trade-off may be desirable only when analyst capacity permits; otherwise, post-hoc calibration or cost-sensitive thresholding is needed, see Figure 3.

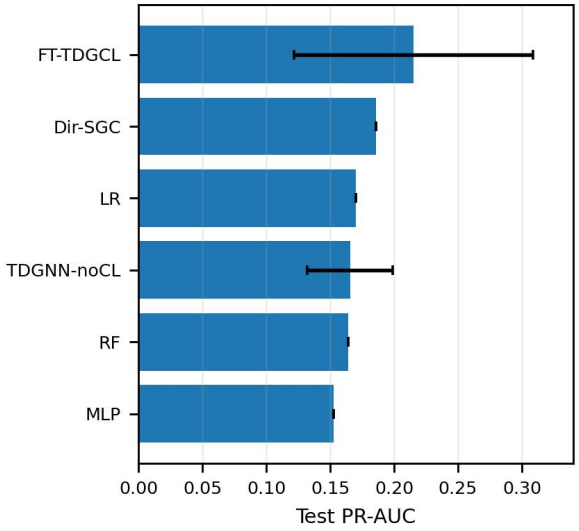

Figure 3 Actual Test PR-AUC

Note: Error bars are one standard deviation where three seeds were run; fixed-seed baselines have no error bar.

6.2 Seed-Level Contrastive Ablation

Table 3 Contrastive Ablation by Random Seed

Seed	No CL	FT-TDGCL	Relative Δ
7	0.1801	0.3038	+68.7%
19	0.1274	0.1174	-7.9%
43	0.1890	0.2244	+18.8%

The seed-level view prevents the mean from hiding instability, see Table 3. Seed 7 improves from 0.1801 to 0.3038, seed 43 improves from 0.1890 to 0.2244, whereas seed 19 decreases from 0.1274 to 0.1174. Consequently, the standard deviation of FT-TDGCL is 0.0935. The contrastive objective is therefore promising but not yet robust enough to claim deterministic superiority. The low-label, temporally shifted regime makes early-stopping trajectories sensitive to initialization; future work should investigate larger seed sets, temporal ensembling, and representation collapse diagnostics.

6.3 Temporal Drift and Failure Modes

Per-step PR-AUC varies sharply even for the best seed. FT-TDGCL reaches 0.5258 in step 42 and 0.4904 in step 38, but falls near the base rate in several late snapshots. Those late periods also contain very few illicit labels—for example, two illicit nodes in step 46 and five in step 45—so a small number of rank reversals changes PR-AUC and F1 substantially. More importantly, the frozen validation threshold transfers poorly when the illicit prevalence changes by an order of magnitude. The result argues for rolling calibration and alert-budget evaluation rather than a single permanent threshold.

Topology-only screening has an additional failure mode: structurally ordinary illicit transactions may be indistinguishable from licit transactions without amounts, address histories, entity information, or off-chain intelligence. Conversely, high fan-out or intermediary behavior can be legitimate. The feature-light setting improves auditability and portability but imposes an information ceiling. The present model should therefore be interpreted as a relational risk layer, not a complete AML decision system.

6.4 Efficiency

The timed local structural block includes snapshot PageRank, DAG-depth, size, and density calculations while the cache is built, see Table 4. It does not include CSV download, file parsing, the one-time full sparse propagation, or an external transaction-ingestion system. Likewise, classifier latency is measured after feature caching and standardization. These boundaries are important: the measurements demonstrate that the learned scoring stage is light, but they are not an end-to-end service-level benchmark.

Table 4 Measured CPU Efficiency

Operation	Measured value
Per-snapshot local structural block	mean 11.10 ms; p95 13.11 ms
FT-TDGCL training	3.241±2.226 s
Forward pass, 16,670 test nodes	6.95±0.64 ms
Cached-feature throughput	2.41±0.22 M nodes/s

6.5 Operational Interpretation and Calibration

The aggregate ranking gain and the weaker thresholded F1 answer different operational questions. PR-AUC evaluates whether illicit nodes are moved upward across possible thresholds, whereas F1 evaluates one validation-selected operating point. Under temporal prevalence drift, a model may improve the ordering but still use a stale threshold. The observed behavior is therefore consistent with a screening model that supplies better candidates to a capacity-constrained queue but requires threshold recalibration before automatic case creation.

For deployment, the natural control variable is often an alert budget rather than a universal probability cutoff. An institution can select the top k scores per hour or optimize a threshold for a target review volume, then monitor precision at k, recall at k, calibration error, and the age of confirmed labels. None of those institutional quantities is available in Elliptic, so this study does not report them. The saved test probabilities make such analyses possible when a valid cost model or review budget is later supplied.

The contrastive seed ablation also suggests a conservative release policy. The proposed objective should be approved only after repeated training runs, validation across consecutive windows, and a challenger comparison against the no-contrastive temporal model. An ensemble or checkpoint-averaging strategy could reduce initialization sensitivity, but it was not evaluated here and is left as future work. This distinction between measured findings and deployment recommendations prevents untested mechanisms from being presented as experimental results.

7 DISCUSSION, THREATS TO VALIDITY, AND ETHICS

7.1 Internal Validity

Internal validity is supported by chronological isolation, validation-only threshold selection, deterministic temporal reconstruction checks, and saved prediction probabilities. However, the hyperparameter search was intentionally limited for efficiency, and only the core temporal comparison uses three seeds. The observed seed variance means that a larger study could change the precise mean and uncertainty. The feature cache also uses each complete snapshot; an event-by-event deployment would require incremental descriptors or micro-batches.

7.2 External Validity

External validity is limited by a single cryptocurrency graph. Bitcoin transaction topology differs from card payments, wire transfers, account-device networks, and enterprise ledgers. Elliptic labels are incomplete and reflect investigative knowledge, so unknown does not mean licit. The model does not train on unknown labels, but unknown nodes affect propagation, which is appropriate structurally while still leaving label-selection bias. Results should not be generalized to an institution without local backtesting.

7.3 Metric and Construct Validity

Construct validity is addressed by prioritizing PR-AUC and showing precision, recall, and F1 rather than claiming that one metric captures operational value. A production system should evaluate precision at a fixed alert budget, expected investigation cost, time-to-detection, calibration, and stability by customer segment. The present benchmark lacks those organizational costs. Its purpose is comparative modeling under a transparent temporal split.

7.4 Ethics and Governance

AML models can affect account access and may amplify bias when graph connectivity correlates with geography, business type, or socioeconomic status. A score should trigger review, not automatic guilt attribution. Human oversight, reason codes, appeal processes, privacy minimization, and monitoring for disparate impact are necessary. The topology-only design reduces dependence on sensitive attributes but does not eliminate proxy risk. Releasing anonymized code and predictions improves scientific auditability, yet deployment data and investigator decisions require stricter governance.

7.5 Reproducibility and Audit Trail

The accompanying package separates raw inputs, deterministic topology caching, model training, evaluation, and manuscript generation. The experiment script writes one row per model and seed, a summary table computed from those rows, test probabilities for every labeled test node, per-temporal-step diagnostics, training histories, run configuration, software versions, and data hashes. The figures and manuscript tables read these files directly. This

organization reduces transcription risk and makes discrepancies traceable to a particular stage rather than to an undocumented spreadsheet operation.

Reproducibility does not imply that every future execution will produce bit-identical neural weights across all hardware and library versions. It means that the data identity, split, transformations, seeds, hyperparameters, stopping rule, threshold rule, and metric code are explicit. The cache validation rejects an unexpected temporal decomposition, and the paper reports uncertainty where repeated seeds were actually completed. Fixed-seed baselines are marked rather than being presented with artificial error bars.

7.6 Research Agenda

The most immediate research priority is temporal calibration. Rolling isotonic or logistic calibration, thresholding under a fixed alert budget, and conformal risk sets could be evaluated without changing the graph encoder, provided each procedure is fitted only on past labels. A second priority is stability: larger seed studies, checkpoint averaging, and temporal ensembles should determine whether the contrastive PR-AUC gain persists while reducing the observed variance. These methods are recommendations, not results of the current experiment.

A broader evaluation should test the topology-only representation on account-to-account bank transfers, card-merchant-device graphs, and heterogeneous entity networks. Such data would permit relation-specific filters, continuous-time updates, and investigator-cost metrics, but would also require stronger privacy and access controls. Incorporating transaction amounts or entity attributes may improve discrimination, yet the feature-light model remains useful as an auditable fallback and as a relational component in a multimodal screening system.

8 CONCLUSION

This paper introduced FT-TDGCL, a feature-light temporal directed graph model for near-real-time financial fraud ranking. The method replaces proprietary transaction attributes with 18 auditable topology features, five directed graph-filter channels, causal GRU context, gated fusion, and a channel-perturbation contrastive objective. On the real Elliptic graph and a strict 29/5/15-step chronological split, contrastive learning increases mean PR-AUC from 0.1655 to 0.2152 and yields mean recall 0.5429. The experiment also reveals important limits: F1 is not universally better, one seed degrades, late-step performance is volatile, and measured latency excludes full ingestion and propagation. These qualified findings support contrastive temporal graph learning as a useful relational screening layer while identifying calibration, robustness, and cross-domain evaluation as the next priorities. The accompanying code and result files make every reported number directly auditable.

COMPETING INTERESTS

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

REFERENCES

[1] Davis J, Goadrich M. The Relationship Between Precision-Recall and ROC Curves. In: Proceedings of the International Conference on Machine Learning (ICML), 2006: 233-240.

[2] Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLoS ONE, 2015, 10(3): e0118432.

[3] Kipf T N, Welling M. Semi-Supervised Classification with Graph Convolutional Networks. In: Proceedings of the International Conference on Learning Representations (ICLR), 2017.

[4] Hamilton W L, Ying R, Leskovec J. Inductive Representation Learning on Large Graphs. In: Advances in Neural Information Processing Systems (NeurIPS), 2017: 1024-1034.

[5] Veličković P, Cucurull G, Casanova A, et al. Graph Attention Networks. In: Proceedings of the International Conference on Learning Representations (ICLR), 2018.

[6] Wu F, Souza A, Zhang T, et al. Simplifying Graph Convolutional Networks. In: Proceedings of the International Conference on Machine Learning (ICML), 2019, 97: 6861-6871.

[7] Weber M, Domeniconi G, Chen J, et al. Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. arXiv:1908.02591, 2019.

[8] Cho K, Merriënboer B, Gulcehre C, et al. Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014: 1724-1734.

[9] You Y, Chen T, Sui Y, et al. Graph Contrastive Learning with Augmentations. In: Advances in Neural Information Processing Systems (NeurIPS), 2020: 5812-5823.

[10] Zhu Y, Xu Y, Yu F, et al. Deep Graph Contrastive Representation Learning. arXiv:2006.04131, 2020.

[11] Hassani K, Khasahmadi A H. Contrastive Multi-View Representation Learning on Graphs. In: Proceedings of the International Conference on Machine Learning (ICML), 2020, 119: 4116-4126.

[12] Qiu J, Chen Q, Dong Y, et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2020: 1150-1160.

[13] Chen T, Kornblith S, Norouzi M, et al. A Simple Framework for Contrastive Learning of Visual Representations. In: Proceedings of the International Conference on Machine Learning (ICML), 2020, 119: 1597-1607.
[14] Khosla P, Teterwak P, Wang C, et al. Supervised Contrastive Learning. In: Advances in Neural Information Processing Systems (NeurIPS), 2020: 18661-18673.
[15] van den Oord A, Li Y, Vinyals O. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748, 2018.
[16] Pareja A, Domeniconi G, Chen J, et al. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020, 34(4): 5363-5370.
[17] Sankar A, Wu Y, Gou L, et al. DySAT: Deep Neural Representation Learning on Dynamic Graphs via Self-Attention Networks. In: Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM), 2020: 519-527.
[18] Xu D, Ruan C, Korpeoglu E, et al. Inductive Representation Learning on Temporal Graphs. In: Proceedings of the International Conference on Learning Representations (ICLR), 2020.
[19] Rossi E, Chamberlain B, Frasca F, et al. Temporal Graph Networks for Deep Learning on Dynamic Graphs. arXiv:2006.10637, 2020.
[20] Trivedi R, Farajtabar M, Biswal P, et al. DyRep: Learning Representations over Dynamic Graphs. In: Proceedings of the International Conference on Learning Representations (ICLR), 2019.
[21] Kumar S, Zhang X, Leskovec J. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2019: 1269-1278.
[22] Dou Y, Liu Z, Sun L, et al. Enhancing Graph Neural Network-Based Fraud Detectors against Camouflaged Fraudsters. In: Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), 2020: 325-334.
[23] Liu Y, Ao X, Qin Z, et al. Pick and Choose: A GNN-Based Imbalanced Learning Approach for Fraud Detection. In: Proceedings of The Web Conference, 2021: 3168-3177.
[24] Ding K, Li J, Bhanushali R, et al. Deep Anomaly Detection on Attributed Networks. In: Proceedings of the SIAM International Conference on Data Mining (SDM), 2019: 594-602.
[25] Akoglu L, Tong H, Koutra D. Graph Based Anomaly Detection and Description: A Survey. Data Mining and Knowledge Discovery, 2015, 29: 626-688.
[26] Lin T-Y, Goyal P, Girshick R, et al. Focal Loss for Dense Object Detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2017: 2980-2988.
[27] Page L, Brin S, Motwani R, et al. The PageRank Citation Ranking: Bringing Order to the Web. Stanford InfoLab, Technical Report 1999-66, 1999.
[28] Breiman L. Random Forests. Machine Learning, 2001, 45: 5-32.
[29] Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 2011, 12: 2825-2830.
[30] Loshchilov I, Hutter F. Decoupled Weight Decay Regularization. In: Proceedings of the International Conference on Learning Representations (ICLR), 2019.