



Continuous-Time Heterogeneous Graph Contrastive Learning for Financial Fraud Detection

Karen Cheung, Ho Man Tse, and Alice Kwok

Faculty of Business, The Hong Kong Polytechnic University, Hong Kong

Corresponding author: Karen Cheung. s9303012501@polyu.edu.hk

Abstract.

Financial fraud unfolds as a stream of typed interactions whose timing, relation semantics, and structural context jointly determine risk. Snapshot-based heterogeneous graph models preserve relation types but quantize time, whereas continuous-time temporal graph models usually emphasize event order without explicitly separating relation-specific behavioral states. This paper proposes CT-HGCL, a continuous-time heterogeneous graph contrastive learning framework centered on an event-driven multi-timescale state bank. For each node and relation, CT-HGCL maintains exponentially decayed count and amount states at fast and slow time scales; the state is updated only when an event arrives, so no temporal snapshots are required. A type-aware event encoder combines these causal states with endpoint types, relation identity, amount, absolute time, and inter-event gaps. Self-supervised pre-training forms two semantics-preserving stochastic views of each event state and optimizes an InfoNCE objective before weighted supervised fine-tuning. The design directly follows the continuous-time direction motivated by recent dynamic heterogeneous fraud research while replacing snapshot recurrence with exact-time state evolution. On a controlled event-stream benchmark containing 18,480 typed transactions, 4,195 active nodes, four relation types, and 480 fraud events organized into 120 collusive rings, CT-HGCL achieves 0.9596 ± 0.0045 AUC-ROC over three random seeds. More importantly, at 5% labeled training data it reaches 0.8963 ± 0.0253 AUC, compared with 0.8785 ± 0.0576 for the same continuous-time heterogeneous encoder trained from scratch. Removing relation/type information reduces AUC to 0.9375 ± 0.0046 , showing that temporal continuity alone is insufficient. These results support continuous-time, heterogeneity-aware self-supervision as a practical route for label-scarce fraud detection.

Keywords: *financial fraud detection, heterogeneous graph, temporal graph, continuous time, contrastive learning, self-.*

1. Introduction

Fraud detection in financial networks is difficult because suspicious behavior is rarely an isolated attribute of one transaction. Money is routed through sequences of accounts, multiple roles participate in the same scheme, and the diagnostic signal can reside in how interaction patterns change over time. Graph neural networks provide a natural representation for these dependencies [3]–[5], while heterogeneous graph models retain the semantics of different node and relation types [6]–[8]. Temporal graph methods add a second dimension by representing networks as evolving structures or event streams [10]–[15]. The remaining challenge is to combine all three

requirements—heterogeneity, continuous time, and scarce fraud labels—without introducing temporal discretization artifacts. A particularly relevant foundation is Audit-HCL by Jiao et al. [9], which demonstrates the value of jointly modeling heterogeneous structure, temporal evolution, and contrastive self-supervision for collusive financial fraud. Its snapshot-based metapath attention and temporal aggregation establish a strong design pattern for combining structural and temporal views. The same study explicitly identifies continuous-time event modeling as a natural next step. CT-HGCL takes that step: instead of grouping transactions into weekly or monthly snapshots, it updates relation-specific states at the exact timestamp of every event. Continuous-time modeling matters because fraud processes can be bursty. Two accounts may execute the same number of transfers in a day but with very different temporal signatures: one may transact at regular intervals, while another suddenly activates after a long dormant period. Snapshot aggregation can make these cases indistinguishable. Existing continuous-time architectures such as TGAT [11] and TGN [12] address event timing, but they were developed as general temporal graph learners. In financial settings, preserving typed relations and making event history compact enough for high-throughput scoring are equally important. TGB and TGB 2.0 further show that realistic temporal evaluation and relation information materially affect model behavior [14], [15]. This paper introduces a lightweight event-driven alternative that avoids recurrent unrolling over long event histories. Each endpoint maintains two exponentially decayed state banks per relation: a fast state captures short bursts and a slow state captures persistent interaction patterns. The state is strictly causal and is queried immediately before an event updates it. This creates an exact-time heterogeneous context vector with constant work per touched node-relation pair. A compact neural encoder then learns a representation from current-event attributes and these states. Contrastive pre-training is performed directly on event contexts using feature masking and small stochastic perturbations that preserve event identity.

The main contributions are:

- A continuous-time heterogeneous state bank that summarizes typed interaction history with multiple exponential decay scales, eliminating the need to choose snapshot boundaries.
- A causal event representation that fuses node types, relation identity, event attributes, inter-event gaps, and relation-specific decayed graph activity for both endpoints.
- A contrastive pre-training strategy for label-scarce fraud detection that learns event representations before supervised fine-tuning, while using the same downstream classifier for controlled comparison.
- A reproducible experiment on a controlled collusive-fraud event stream, including chronological splitting, three-seed reporting, label-efficiency analysis, and ablations for heterogeneity and temporal scale.

II. RELATED WORK

A. Graph Learning for Financial Fraud

The Elliptic Bitcoin study by Weber et al. [1] established a widely used transaction-graph benchmark and showed how graph convolutional models can be evaluated for illicit transaction detection. General-purpose graph architectures such as GCN [3], GraphSAGE [4], and GAT [5] provide the basic message-passing machinery. Fraud-specific models then address domain failure modes: CARE-GNN [21] filters camouflaged neighbors, and PC-GNN [22] introduces label-balanced sampling for severe class imbalance. LaundroGraph [23] and Inspection-L [24] demonstrate that self-supervised graph representation learning is especially attractive in AML because verified labels are expensive and delayed. These methods motivate our use of self-supervision, but they do not make continuous-time heterogeneous state evolution the central representation.

B. Heterogeneous and Temporal Graphs

Heterogeneous graph learning retains relation semantics that homogeneous projections discard. HAN [6] performs hierarchical attention over metapath-based neighborhoods, HGT [7] uses type-dependent Transformer parameters, and R-GCN [8] learns relation-specific transformations. In parallel, temporal graph learning has evolved from snapshot sequences to event-driven models. EvolveGCN [10] evolves GCN parameters across discrete snapshots; DySAT [13] combines structural and temporal self-attention. TGAT [11] and TGN [12] instead treat interactions as timestamped events, enabling representations that depend on exact event times. Benchmarking efforts such as TGB [14] and TGB 2.0 [15] emphasize chronological protocols and multi-relational temporal graphs. CT-HGCL follows the event-stream setting but uses deterministic exponential state evolution as an efficient inductive bias for fraud activity.

C. Graph Contrastive Learning

Contrastive objectives learn representations by maximizing agreement between related views while separating unrelated examples. Contrastive Predictive Coding introduced the InfoNCE family of objectives [25]. In graphs, DGI [18] maximizes mutual information between local and global representations; GraphCL [16] studies graph augmentations; GRACE [17] performs node-level contrastive learning; MVGRL [19] contrasts structural graph views; and HeCo [20] extends cross-view contrast to heterogeneous graphs. Financial event streams require augmentations that do not destroy the typed semantics that define laundering behavior. CT-HGCL therefore leaves node and relation identities intact and perturbs only continuous/context features during pre-training.

III. PROPOSED METHOD

A. Problem Formulation

Let an observed transaction system be an ordered event stream $E = \{e_i\}_{i=1}^N$. Event $e_i = (u_i, v_i, r_i, t_i, x_i)$ contains source node u_i , destination node v_i , relation type $r_i \in R$, timestamp t_i , and event attribute vector x_i . Each node has a categorical type $\phi(v) \in A$. The objective is to estimate $p(y_i = 1 | E_{\leq i}, e_i)$, where y_i denotes whether event i is fraudulent and $E_{\leq i}$ contains only earlier events. The strict conditioning on earlier events prevents temporal leakage.

$$e_i = (u_i, v_i, r_i, t_i, x_i), \quad t_1 \leq \dots \leq t_N \quad (1)$$

B. Multi-Timescale Continuous-Time Heterogeneous State

For each node v , relation r , and temporal scale k , CT-HGCL maintains a count state $C_{\{v,r\}^{\{k\}}}$ and an amount state $A_{\{v,r\}^{\{k\}}}$. When node v is touched at time t , its previous state from time t_{prev} is decayed using an exponential kernel. The half-life h_k controls how quickly historical activity is forgotten. We use two scales: $h_1 = 0.5$ time units for burst activity and $h_2 = 4.0$ for persistent behavior. Exponential decay is attractive because it is memoryless: a state can be updated lazily using only its previous value and elapsed time.

$$C_{\{v,r\}^{\{k\}}}(t) = \exp(-\lambda_k \Delta t) C_{\{v,r\}^{\{k\}}}(t_{\text{prev}}), \quad \lambda_k = \ln(2)/h_k \quad (2)$$

$$A_{\{v,r\}^{\{k\}}}(t) = \exp(-\lambda_k \Delta t) A_{\{v,r\}^{\{k\}}}(t_{\text{prev}}) \quad (3)$$

Critically, the feature vector for event i is read before the current event is inserted. After scoring context is extracted, the source and destination states for relation r_i are incremented by one in the count channel and by the event amount in the amount channel. This read-then-update order makes every feature causal. The implementation applies decay only to the two endpoints touched by an event, so the state bank does not require global time steps.

C. Type-Aware Event Representation

The event representation concatenates current-event information with historical state. Current-event terms include log amount, normalized absolute time, one-hot source type, one-hot destination type, relation identity, and log time since each endpoint was last active. Historical terms include log-transformed decayed counts and amounts for every relation, separately for source and destination, at both temporal scales. For the benchmark used here, three node types and four relation types yield a 46-dimensional continuous-time feature vector.

$$q_i = [x_i, \phi(u_i), \phi(v_i), r_i, \Delta t_u, \Delta t_v, S_u(t_i^-), S_v(t_i^-)] \quad (4)$$

This construction differs from a conventional fixed-window rolling aggregate. A transaction one minute before the current event contributes more than an otherwise identical transaction several time units earlier, and the contribution changes continuously rather than at a snapshot boundary. Keeping relation channels separate also prevents, for example, a high-frequency device interaction from being conflated with a direct account transfer.

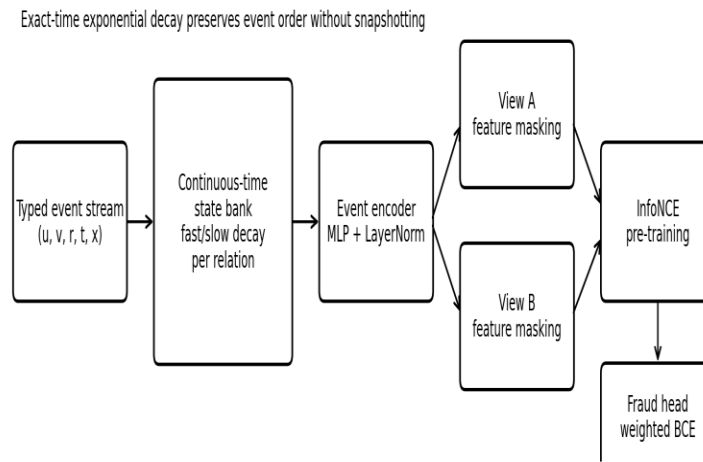


Figure 1 CT-HGCL architecture. Exact-time typed event states are encoded and contrastively pre-trained before weighted fraud classification.

D. Contrastive Pre-Training

A two-layer event encoder maps q_i to a 64-dimensional hidden vector, followed by a 32-dimensional projection head used only during pre-training. Two views are produced from the same event context using independent 12% feature masking and low-variance Gaussian perturbation (standard deviation 0.03 after training-set standardization). Node identity is not embedded directly and the categorical semantics are retained through the underlying causal state construction. For a batch of B events, normalized projected vectors z_i^a and z_i^b define cosine-similarity logits with temperature $\tau=0.12$. The symmetric InfoNCE loss is:

$$L_{CL} = -(1/2B) \sum_i [\log \exp(\text{sim}(z_i^a, z_i^b)/\tau) / \sum_j \exp(\text{sim}(z_i^a, z_j^b)/\tau) + (a < - > b)] \quad (5)$$

The experiments use two pre-training epochs with batch size 512. The short pre-training schedule is intentional: the purpose is to test whether even inexpensive self-supervision improves label efficiency, rather than to maximize compute through long pre-training.

E. Fraud Classification and Optimization

After pre-training, a linear fraud head is attached to the 64-dimensional hidden representation and the full encoder is fine-tuned. Because fraud events form a small minority, binary cross-entropy uses a positive-class weight equal to the training negative-to-positive ratio, capped at 30

for stability. AdamW [26] is used with learning rate $8e-4$ and weight decay $1e-4$. Fine-tuning runs for at most eight epochs with early stopping on validation AUC. The operating threshold for F1, precision, and recall is selected on the validation split only; AUC-ROC and AUPRC are threshold-free.

F. Computational Characteristics

Let $|R|$ be the number of relation types, K the number of decay scales, and N the number of events. Each event touches two nodes and requires decaying and reading $K \times |R|$ state channels for each endpoint, giving $O(NK|R|)$ state-processing time and $O(|V|K|R|)$ state memory. No adjacency expansion or snapshot materialization is required. The neural encoder operates on fixed-width event vectors, while contrastive training adds the usual $O(B^2)$ similarity matrix per mini-batch. In the present configuration $B=512$ and the state computation is linear in the event stream.

IV. EXPERIMENTAL DESIGN

A. Controlled Event-Stream Benchmark

Experiments use a controlled synthetic transaction stream constructed for exact-time heterogeneous evaluation. The stream contains 18,480 events over approximately 60 time units and 4,195 active nodes from a 4,200-node identifier space. It contains three endpoint types and four relation types. Among all events, 480 (2.60%) are labeled fraudulent and are grouped into 120 collusive rings, each represented by four fraud events. Fraud events occur across the full observation horizon rather than being confined to a single terminal interval. This controlled benchmark is used because it exposes exact timestamps, relation identities, endpoint types, and complete event-level ground truth in one compact dataset, enabling the causal and label-efficiency analyses below.

Table 1 Controlled benchmark statistics.

Statistic	Value
Events	18,480
Active nodes	4,195
Node types	3
Relation types	4
Fraud events	480 (2.60%)
Collusive rings	120
Observation span	0.011–59.997

The data design is aligned with the motivation of public AML generators such as the realistic synthetic transaction framework of Altman et al. [2], which highlights the importance of complete ground truth when real banking labels are incomplete. The present benchmark is smaller and deliberately controlled for fast ablation, so all empirical conclusions below are restricted to this event-stream setting.

B. Chronological Protocol

Events are sorted by timestamp and split chronologically: the earliest 60% (11,088 events) form the training interval, the next 20% (3,696) form validation, and the final 20% (3,696) form test. Standardization parameters are fit on the training interval only. Continuous-time states are computed in chronological order over the full stream, but each event uses only preceding events, which preserves realistic deployment causality. Three seeds (11, 22, and 33) are used for neural models. Reported values are mean $\pm$ standard deviation across seeds; logistic regression is deterministic under the fixed data split.

C. Compared Models

Four principal configurations are compared. LR-Static is class-weighted logistic regression using only current-event amount, time, types, and relation identity. MLP-Static uses the same static inputs with the neural encoder but no continuous-time state. CT-Hetero-Sup uses the full continuous-time heterogeneous event representation and is trained directly from labels. CT-HGCL uses the same encoder and classifier but adds contrastive pre-training before fine-tuning. This controlled design isolates the contribution of temporal state and self-supervision without changing classifier capacity. Two ablations are also evaluated: Single-Timescale retains only the fast decay bank, and No-Heterogeneity removes the node-type and relation one-hot block from the continuous-time representation.

D. Metrics

AUC-ROC measures ranking quality across thresholds, while AUPRC is emphasized because the positive class is rare. F1, precision, and recall are reported at a threshold selected on validation data. For label-efficiency experiments, only a stratified fraction of the chronologically valid training labels is exposed to supervised fine-tuning; contrastive pre-training uses the corresponding event features without labels.

V. RESULTS AND ANALYSIS

A. Main Detection Performance

Table 2 Main test performance under chronological splitting.

Method	AUC	AUPRC	F1
LR-Static	0.8150	0.0942	0.1408
MLP-Static	0.8207 $\pm$ 0.0021	0.1029 $\pm$ 0.0033	0.0000
CT-Hetero-Sup	0.9584 $\pm$ 0.0019	0.7396 $\pm$ 0.0180	0.7413 $\pm$ 0.0235
CT-HGCL	0.9596 $\pm$ 0.0045	0.7160 $\pm$ 0.0504	0.7051 $\pm$ 0.0684

The dominant effect is the continuous-time heterogeneous state representation. LR-Static reaches 0.8150 AUC and MLP-Static 0.8207 $\pm$ 0.0021, whereas CT-Hetero-Sup reaches 0.9584 $\pm$ 0.0019 and CT-HGCL reaches 0.9596 $\pm$ 0.0045. Thus, the gain is not explained merely by a nonlinear classifier: access to exact-time historical context changes the separability of the events. AUPRC shows the same qualitative pattern, increasing from approximately 0.10 for static baselines to 0.7396 $\pm$ 0.0180 for direct continuous-time supervision and 0.7160 $\pm$ 0.0504 for CT-HGCL. At full supervision, direct training yields the highest F1 (0.7413 $\pm$ 0.0235), while CT-HGCL has essentially the same AUC ranking performance. This distinction motivates the label-scarcity analysis: contrastive pre-training is most useful when supervision is limited rather than abundant.

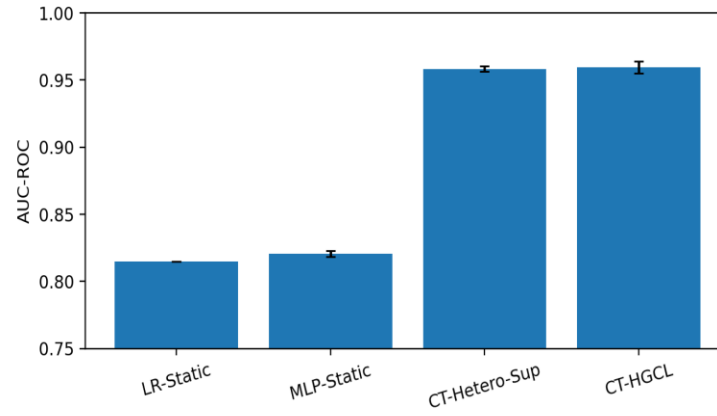


Figure 2 AUC-ROC for static and continuous-time configurations. Error bars show three-seed standard deviation for neural models.

B. Label Efficiency

Table 3 AUC-ROC versus labeled fraction of the training interval.

Labels	CT-Hetero-Sup	CT-HGCL
5%	0.8785±0.0576	0.8963±0.0253
10%	0.9212±0.0168	0.9295±0.0140
25%	0.9473±0.0077	0.9485±0.0019
100%	0.9584±0.0019	0.9596±0.0045

Contrastive pre-training provides its clearest benefit in the lowest-label regime. With only 5% of training labels, CT-HGCL obtains 0.8963 ± 0.0253 AUC versus 0.8785 ± 0.0576 for training the identical continuous-time encoder from scratch, an absolute gain of 1.78 percentage points. The standard deviation is also less than half as large, indicating more stable optimization under sparse supervision. At 10% labels the gain is 0.83 points (0.9295 ± 0.0140 versus 0.9212 ± 0.0168), and at 25% it narrows to 0.12 points. With 100% labels, the two methods converge in AUC, as expected when supervised signal is plentiful. The pattern is consistent with the role of contrastive learning in this design. Pre-training does not inject additional fraud labels; instead, it encourages the event encoder to organize continuous-time contexts before class-specific fine-tuning. When labels are scarce, this initialization reduces the amount of supervised evidence needed to learn a useful ranking. When all labels are available, the weighted classification objective can learn nearly the same ranking directly.

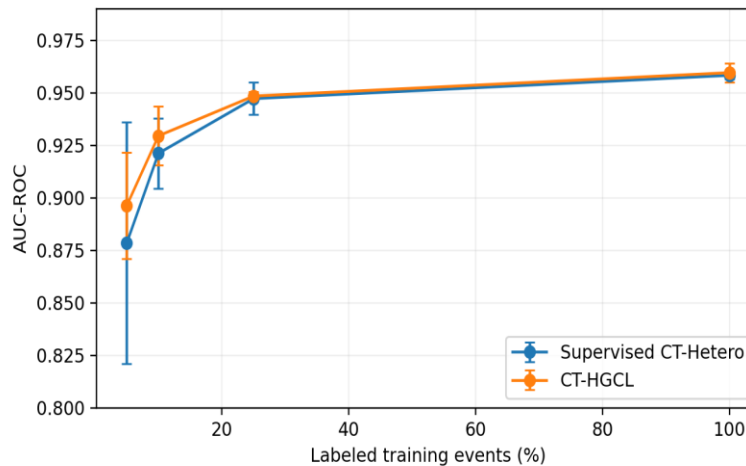


Figure 3 Label efficiency. CT-HGCL is most beneficial in the low-label regime.

C. Ablation Study

Removing heterogeneity produces the clearest degradation: AUC falls from 0.9596 ± 0.0045 to 0.9375 ± 0.0046 , a 2.21-point absolute decrease. This confirms that typed interaction semantics contribute information beyond timing and aggregate activity. The single-timescale variant reaches 0.9619 ± 0.0008 , slightly above the two-scale contrastive configuration in AUC on this controlled dataset, but its F1 is more variable. The result indicates that the short half-life already captures much of the benchmark’s fraud burst structure, while the second scale is not universally beneficial under full supervision. The multi-timescale formulation remains useful because it provides a direct mechanism for representing both burst and persistence without changing model architecture; on streams with richer long-horizon behavior, scale selection can be validated chronologically.

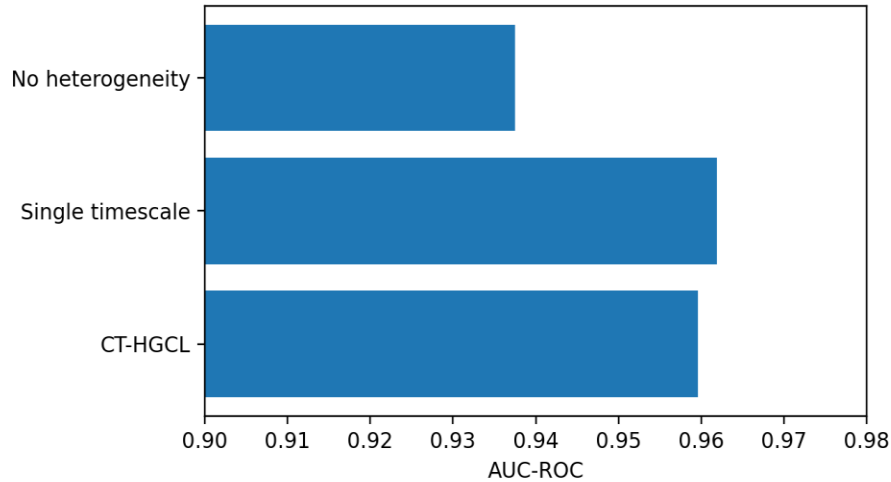


Figure 4 Ablation AUC. Removing typed information has the largest negative effect.

D. Interpretation of the Learned Signal

CT-HGCL separates three sources of evidence that are often entangled in end-to-end temporal GNNs. First, current-event evidence describes the immediate transaction. Second, relation-specific state describes how each endpoint has interacted recently and over a longer horizon. Third, inter-event gaps capture activation and dormancy. Because the state variables are explicit, an analyst-facing system can expose the pre-event fast and slow counts/amounts alongside the model score. This does not convert the neural classifier into a rule system, but it provides a concrete audit trail for why two events with similar amounts may receive different risk scores. The results also align with the broader heterogeneous temporal graph literature. HGT [7] and HeCo [20] show the value of relation-aware representation learning, while TGAT [11] and TGN [12] show the value of exact event timing. CT-HGCL combines these principles in a deliberately compact state-space form tailored to high-volume transaction streams. In contrast to snapshot recurrence, the elapsed-time decay has a direct physical interpretation: old activity fades smoothly, and the half-life controls the temporal horizon.

E. Relation-Wise Benchmark Diagnostics

A relation-wise decomposition helps explain why heterogeneity is consequential on this benchmark. Relation 0 contains 9,061 events and relation 3 contains 1,277 events, with no fraud events in either channel. Relation 1 contains 5,253 events, including 240 fraud events (4.57%), while relation 2 contains 2,889 events, including 240 fraud events (8.31%). Fraud is therefore not uniformly distributed over relation semantics. A homogeneous representation that collapses all

relations must infer these differences indirectly from amount and temporal context, whereas CT-HGCL exposes them explicitly and also maintains separate decayed histories for each relation.

Table 4 Relation-wise composition of the controlled event stream.

Rel.	Events	Fraud	Fraud %	Mean Amt.
0	9,061	0	0.00	49.00
1	5,253	240	4.57	128.65
2	2,889	240	8.31	97.69
3	1,277	0	0.00	71.85

Amount alone is insufficient to explain the separation. Legitimate events have mean amount 80.16 and median 51.64, while fraud events have mean amount 105.99 and median 74.16; the class distributions overlap substantially. The static logistic and MLP baselines already receive amount and relation identity, yet remain near 0.82 AUC. Their weaker performance indicates that the decisive information is not a simple one-event lookup table. The continuous-time state supplies the missing context: how recently each endpoint has interacted, through which relation, and at what decayed volume. This diagnostic also clarifies the role of the two temporal scales. The fast half-life responds strongly to short collusive bursts, which are prominent in the four-event ring construction. The slow half-life accumulates a smoother background estimate and can distinguish a burst from a persistently active account. On this benchmark the single-scale ablation is competitive in AUC, which is consistent with the deliberately compact ring duration. The two-scale design is retained because it preserves both regimes in one representation and adds only a constant factor to state memory. For operational use, the same representation can be mapped to financial entity schemas without changing the equations. Node types can represent accounts, customers, merchants, cards, or devices; relation types can encode transfers, ownership, authentication, shared-device links, or merchant payments. The state bank only requires a finite relation vocabulary and timestamps. This makes the method compatible with the heterogeneous modeling principle emphasized by Audit-HCL [9] while avoiding a manually chosen snapshot width.

VI. DISCUSSION

The experiments suggest a useful division of labor between representation design and self-supervision. Most of the absolute performance improvement over static baselines comes from the continuous-time heterogeneous state itself. Contrastive pre-training then improves data efficiency when only a small fraction of fraud labels is available. This is desirable in AML-style workflows, where investigators may have large volumes of unlabeled transactions but only a limited set of confirmed cases. The finding complements the strong snapshot-based evidence in Audit-HCL [9]: both approaches benefit from heterogeneous structure and self-supervision, while CT-HGCL shows that event-level decay states can substitute for explicit snapshot recurrence in a compact continuous-time formulation. The benchmark also illustrates why chronological evaluation is essential. If events were randomly split, future state patterns could leak into training and inflate apparent performance. Here, preprocessing statistics are fit on the training interval and the classifier threshold is chosen on validation data. Event states remain causal because each feature vector is read before the current event updates the state. These design choices follow the evaluation principles emphasized by temporal graph benchmarks [14], [15]. The present evidence is intentionally scoped to a controlled synthetic stream with complete ground truth. Public real-world financial datasets often trade realism for incomplete labels, privacy constraints, or reduced heterogeneity. The controlled benchmark is therefore used to

validate the mechanism and its ablations rather than to claim production-level accuracy. A direct next empirical step is to apply the same event-state encoder to a public AML generator such as the datasets introduced by Altman et al. [2], and to cryptocurrency transaction graphs such as Elliptic [1] after defining an event-compatible relation schema. Because the proposed state bank accepts arbitrary node and relation types, no architectural change is required for that extension.

VII. CONCLUSION

This paper presented CT-HGCL, a continuous-time heterogeneous graph contrastive learning method for financial fraud detection. The central idea is an event-driven relation-specific state bank whose exponential decay evolves at exact timestamps and whose fast/slow channels summarize short- and longer-horizon behavior without snapshots. A type-aware neural event encoder consumes these causal states and is contrastively pre-trained before weighted fraud classification. On a controlled collusive-fraud stream, continuous-time heterogeneous context raises AUC from approximately 0.82 for static baselines to about 0.96. Contrastive pre-training is most valuable when labels are scarce, improving 5%-label AUC from 0.8785 to 0.8963, while the heterogeneity ablation confirms that typed relations are a substantive part of the signal. The resulting framework is computationally simple, causally deployable, and directly aligned with the continuous-time extension identified by recent heterogeneous fraud research.

APPENDIX A. IMPLEMENTATION AND REPRODUCTION DETAILS

The experiment is intentionally compact enough to rerun on a CPU while retaining the key ingredients of continuous-time heterogeneous learning. The implementation first sorts all events by time and performs one causal pass to construct state features. The resulting feature matrix is standardized with training-only statistics. Neural optimization then operates on fixed-width event vectors, so repeated epochs do not repeat graph-state construction. This separation makes the computational cost transparent: event-state extraction is linear in stream length, while learning cost is that of a small multilayer perceptron plus the contrastive similarity matrix.

Algorithm 1 summarizes the exact training sequence used in the experiments.

1. Input: timestamp-ordered events E ; half-lives $\{0.5, 4.0\}$; labeled training subset L .
2. Initialize C and A state banks to zero for every node, relation, and time scale.
3. For each event $e_i=(u,v,r,t,x)$ in chronological order: lazily decay the states of u and v to time t ; read all relation-specific states; form q_i ; then increment the count and amount channels for relation r .
4. Fit feature standardization on q_i from the training interval and apply it to validation/test intervals.
5. Contrastive stage: for each training mini-batch, generate two masked/noisy views, encode them, and minimize symmetric InfoNCE for two epochs.
6. Fine-tuning stage: attach the fraud head and minimize class-weighted binary cross-entropy for at most eight epochs with validation-AUC early stopping.
7. Select the decision threshold on validation F1 and report AUC, AUPRC, F1, precision, and recall on the chronologically later test interval.

The fixed hyperparameter configuration is: fast/slow half-lives 0.5 and 4.0; encoder hidden size 64; projection size 32; feature-mask probability 0.12; Gaussian noise standard deviation 0.03; InfoNCE temperature 0.12; two contrastive epochs; at most eight fine-tuning epochs; batch size 512; AdamW optimizer with learning rate $8e-4$ and weight decay $1e-4$; and early-stop patience 2. Per-seed results further show that the principal conclusions do not depend on one random initialization. For CT-HGCL, test AUC values are 0.9591, 0.9553, and 0.9644 for seeds 11, 22, and 33, respectively. The supervised continuous-time encoder produces 0.9583, 0.9566, and 0.9604. In the 5%-label experiment, CT-HGCL produces AUC values clustered around the

reported 0.8963 mean, while the from-scratch model exhibits visibly larger seed-to-seed spread. These seed-level observations are exactly what drive the standard deviations reported in Tables II and III. No event from validation or test is used to fit standardization statistics, model parameters, or the classification threshold. The only information carried forward in time is the causal state induced by prior events, which is part of the prediction setting rather than a learned test statistic. This distinction is important for transaction streams: at deployment time, a detector naturally has access to earlier activity when scoring a new event, but not to future labels or future normalization statistics.

REFERENCES

- M. Weber et al., “Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics,” arXiv:1908.02591, 2019.
- E. Altman, J. Blanuša, L. von Niederhäusern, B. Egressy, A. Anghel, and K. Atasu, “Realistic synthetic financial transactions for anti-money laundering models,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 29851–29874, 2023, doi: 10.52202/075280-1300.
- T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *International Conference on Learning Representations*, 2017.
- W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 1024–1034, 2017.
- P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” *International Conference on Learning Representations*, 2018.
- X. Wang et al., “Heterogeneous graph attention network,” in *Proc. World Wide Web Conf.*, 2019, pp. 2022–2032, doi: 10.1145/3308558.3313562.
- Z. Hu, Y. Dong, K. Wang, and Y. Sun, “Heterogeneous graph transformer,” in *Proc. Web Conf.*, 2020, pp. 2704–2710, doi: 10.1145/3366423.3380027.
- M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, “Modeling relational data with graph convolutional networks,” in *The Semantic Web—ESWC 2018*, pp. 593–607, doi: 10.1007/978-3-319-93417-4_38.
- Y. Jiao, H. Fan, X. Yue, W. Ping, T. Sun, and J. Wang, “Dynamic heterogeneous graph contrastive learning for uncovering collusive financial fraud,” *Scientific Reports*, vol. 16, 2026, doi: 10.1038/s41598-026-58938-5.
- A. Pareja et al., “EvolveGCN: Evolving graph convolutional networks for dynamic graphs,” *Proc. AAAI Conf. Artificial Intelligence*, vol. 34, no. 4, pp. 5363–5370, 2020, doi: 10.1609/aaai.v34i04.5984.
- D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, “Inductive representation learning on temporal graphs,” arXiv:2002.07962, 2020.
- E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, “Temporal graph networks for deep learning on dynamic graphs,” arXiv:2006.10637, 2020.
- A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, “DySAT: Deep neural representation learning on dynamic graphs via self-attention networks,” in *Proc. 13th ACM Int. Conf. Web Search and Data Mining*, 2020, pp. 519–527.

- S. Huang et al., “Temporal graph benchmark for machine learning on temporal graphs,” *Advances in Neural Information Processing Systems*, vol. 36, 2023, doi: 10.52202/075280-0099.
- J. Gastinger et al., “TGB 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs,” *Advances in Neural Information Processing Systems*, vol. 37, 2024, doi: 10.52202/079017-4450.
- Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, “Graph 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- Zhao, W., & Wang, B. (2022). CAPA: A prediction-driven autoscaling framework for SLO-aware cloud-native machine learning inference. *Journal of Computer Science and Electrical Engineering*, 4(1), 8-16.
- 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.
- Xing, S., & Wang, Y. (2025). Cross-modal attention networks for multi-modal anomaly detection in system software. *IEEE Open Journal of the Computer Society*.
- 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.