Journal of Innovative Technologies
https://journalinnotech.com/
Vol 2(2) 2026

Static Structure versus Edge-Order Representation for Drive-by-Download Detection in Linux Audit Graphs: A Reproducible Evaluation

Xinyu Zhang[1], Jiale Gao[2], Yinchu Wu[3,*]

[1]Applied Computing, University of Toronto, Toronto, ON, Canada
[2]Applied Analytics, Columbia University, New York, NY, USA
[3] Computer Science, Shanghai Jiao Tong University, Shanghai, China

* Corresponding Email: wu.yinchu.cs@qq.com

Abstract

The public StreamSpot corpus is widely used to evaluate anomaly detection over Linux audit graphs, yet its released form removes timestamps while retaining the original edge sequence. This study asks whether that residual ordering improves drive-by-download detection beyond static structural frequencies. We compare three representations under one isolation-based detector: static typed-edge frequencies, static features augmented with edge-type transitions, and static features augmented with full typed-edge transitions. The verified corpus contains 600 graphs across five benign activities and one attack activity. Ten stratified repetitions allocate 300 benign graphs to training, 75 to threshold selection, and 125 benign plus all 100 attack graphs to testing. Static features already provide perfect mean ranking performance, with AUPRC and AUROC both equal to 1.0000. Full typed-edge order yields AUPRC 0.9987±0.0017 and AUROC 0.9991±0.0012. Its mean F1 rises from 0.9635 to 0.9689 and its false-positive rate falls from 0.0608 to 0.0520, but paired tests give p=0.375 and p=0.416, respectively. Gmail false positives increase from 0.156 to 0.228. Edge order offers no stable aggregate benefit on this corpus and expands the mean feature space from 121.0 to 3,634.7 dimensions. The result supports simpler static representations for this narrowly controlled setting.

***Keywords*:** provenance graphs; audit logs; anomaly detection; edge order

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/)

1. Introduction

1.1. Motivation and Scope

Operating-system audit streams record interactions among processes, files, sockets, and other kernel-visible entities. Their graph form preserves relations that a flat log sequence cannot express, while their arrival order may reveal execution regularities that aggregate counts discard. StreamSpot established a compact streaming treatment of heterogeneous audit graphs and released a benchmark containing repeated benign activities and a drive-by-download attack [1]. The benchmark remains attractive for a short empirical study because its six balanced scenarios permit controlled sampling without synthetic anomalies.

The released corpus also imposes a precise boundary on temporal claims. Its maintainers mapped node and edge types to compact symbols, collapsed consecutive block reads, and removed timestamp values after sorting the raw records. The line sequence is retained, but elapsed time, concurrency, and inter-event duration cannot be recovered. This paper studies edge order rather than continuous time. That distinction matters because sequential log analysis can exploit event order and event counts without representing graph causality [2], while provenance-based detection links actions to the entities that produced or received them [3]. Treating the public file as a timestamped event stream would overstate what the data contains.

Prior graph research gives credible reasons to expect both positive and negative results. Neighborhood aggregation can summarize local structure and generalize to unseen nodes [4]. Continuous-time attention can preserve temporal constraints that static aggregation ignores [5]. StreamSpot graphs, by contrast, represent short controlled executions whose attack class differs sharply in size and typed-edge composition from several benign classes. Static statistics may already separate the attack, leaving little room for order features. A richer transition vocabulary can also dilute useful frequencies and increase estimator variance when only 300 training graphs are available.

1.2. Research Questions and Contributions

A. Research Questions

The evaluation addresses three questions. RQ1 asks whether edge-order augmentation improves ranking quality, measured by area under the precision-recall curve and area under the receiver-operating-characteristic curve. RQ2 asks whether it improves thresholded decisions, measured by precision, recall, F1, Matthews correlation coefficient, and false-positive rate. RQ3 asks whether any aggregate change is stable across benign activities or merely transfers false alarms from one scenario to another. These questions separate score ranking, operational thresholding, and scenario-specific reliability.

B. Contributions

The study contributes a checksum-verified reconstruction of all 600 released graphs, a controlled comparison in which representations share the same detector and data partitions, and a ten-split assessment with paired tests. It also reports an intermediate edge-type transition view, isolating whether coarse order is preferable to high-dimensional typed-edge transitions. The comparison does not propose a new detector and does not claim real-time temporal inference. Its narrower aim is to measure the information retained by the released sequence and to identify when added order complexity is unsupported by the evidence.

2. Related Work

2.1. Static and Temporal Graph Analysis

A. Static Structural Representation

Static graph learning usually maps a fixed neighborhood or complete snapshot to a vector before prediction. This approach is computationally convenient, but it merges interactions that may have occurred in different orders. Dynamic extensions can evolve graph convolution parameters across snapshots, allowing the same architecture to adapt when topology changes [6]. Such snapshot methods still require defensible temporal boundaries, which the public StreamSpot file cannot supply. Our static representation instead summarizes graph size, endpoint types, edge types, typed-edge triples, and degree dispersion over each completed graph.

B. Order-Aware Representation

Order-aware graph analysis ranges from evolving embeddings to local sequence encoders. NetWalk updates node representations and anomaly scores as a network changes, avoiding repeated processing of the complete history [7]. Attention-based temporal convolution combines current structure with recurrent history for anomalous-edge detection [8]. Structural-temporal methods extract enclosing subgraphs at several snapshots and use recurrent units to represent their evolution [9]. Transformer-based detection further couples spatial and temporal encodings for unattributed dynamic graphs [10]. These approaches operate on richer temporal observations than StreamSpot provides. Our transition counts test a smaller question: whether adjacent released edge tokens add discriminative information after static composition is known.

2.2. One-Class Graph Anomaly Detection

Security evaluation often has abundant normal traces and few representative attacks. A one-class protocol reflects that asymmetry by fitting only benign graphs and reserving attack labels for assessment. Its credibility depends on a threshold chosen without test attacks. We fit an isolation-based ensemble on benign training graphs and select the 95th percentile of benign validation

scores. The detector is intentionally fixed across feature views, so measured differences arise from representation and split variation rather than from classifier selection.

This protocol also separates graph-level and node-level claims. Each StreamSpot graph is one completed activity execution with a graph ID. The public release supplies scenario labels, not labels for individual processes, files, edges, or timestamps. A graph predicted as anomalous indicates that the activity batch differs from the learned benign distribution; it does not localize the responsible event. This limitation prevents direct comparison with malicious-entity tracing on DARPA provenance corpora.

2.3. Positioning of the Study

The paper is a representation ablation on a single public benchmark. It does not compare large neural architectures because the sample size is 600 graphs and the attack mechanism is singular. The strongest design holds graph reconstruction, benign partitions, anomaly estimator, threshold rule, and metrics constant. Only order features change. This focus complements work that learns informative spatial-temporal node encodings but often changes several components at once. A narrow negative result is useful when it identifies a condition under which added temporal complexity fails to improve reproducible detection.

3. Experimental Setup and Methodology

3.1. Dataset and Integrity Audit

The experiment uses all.tar.gz from the official sbustreamspot-data repository. The downloaded archive contains 87,860,166 bytes and has SHA-256 digest 1551d7eba96fdea4958b9c0b6c15c773983a5ebdf9f1ffe4430cbb7ea85d23cf. Its extracted all.tsv contains 2,288,207,702 bytes. A streaming parser verified six tab-separated fields per record and strictly increasing grouped graph IDs from 0 through 599. The reconstructed corpus contains 89,770,902 edges. Table 1 reports statistics computed directly from this verified file; the means reproduce the rounded values published with the benchmark.

The drive-by-download scenario represents a visit to a malicious URL that exploits a Flash vulnerability and gains root access. The remaining scenarios capture routine browsing, email, gaming, and downloading activity. No fraud, money-laundering, or node-level attack labels are present. Recent semi-supervised dynamic anomaly detection demonstrates how sparse labels and unlabeled interactions can be combined [11], while cross-distribution work emphasizes that temporal results may vary with data diversity [12]. Our attack class is held out completely, retaining a stricter one-class setting but a narrower threat distribution.

Table 1. Verified StreamSpot corpus statistics.

Scenario	Label	Graphs	Sum of per-graph nodes	Mean nodes	Total edges	Mean edges
YouTube	Benign	100	829,179	8,291.79	11,322,897	113,228.97
Gmail	Benign	100	682,730	6,827.30	3,738,210	37,382.10
VGame	Benign	100	863,734	8,637.34	11,295,753	112,957.53
Attack	Attack	100	889,080	8,890.80	2,842,345	28,423.45
Download	Benign	100	883,146	8,831.46	31,081,383	310,813.83
CNN	Benign	100	899,009	8,990.09	29,490,314	294,903.14

Source: computed from the official StreamSpot archive, identified by the SHA-256 digest reported above. Node counts are summed within graphs and do not imply globally unique node identities.

3.2. Representation and Detection Protocol

A. Static Structural Features

For graph (g), the static vector includes log-transformed edge count, node count, and maximum undirected incident degree; mean incident degree; degree coefficient of variation; normalized source-node-type frequencies; normalized destination-node-type frequencies; normalized edge-type frequencies; and normalized typed-edge frequencies. A typed edge is the categorical triple (source type, edge type, destination type). Counts are divided by the number of edges in (g), except the five size and degree statistics. This design preserves heterogeneous composition without using node identifiers, which would not transfer across graph executions. Residual-based attributed graph analysis supports measuring departures that cannot be explained by regular structure [13].

B. Edge-Order Features

The coarse order view appends normalized frequencies of adjacent edge-type pairs, transition entropy, and the fraction of repeated edge types. The full order view also appends adjacent typed-edge pairs, typed-transition entropy, and the typed-edge repetition fraction. Adjacent means consecutive records belonging to the same graph in all.tsv; it carries no duration information. Table 2 separates the three views. Joint attribute selection and anomaly scoring can suppress irrelevant dimensions [14], yet feature selection would introduce another learned component. We retain every transition observed in the benign training partition and assess whether the resulting expansion helps without such intervention.

Table 2. Feature-view definitions and fitted dimensionality.

Representation	Shared static content	Added order content	Mean fitted dimension	Range across splits
Static	Size, degree, endpoint type, edge type, typed-edge frequencies	None	121.0	120–122
Edge order	All static features	Edge-type transitions and two order summaries	608.1	604–614
Typed order	All static and edge-order features	Typed-edge transitions and two typed-order summaries	3,634.7	3,595–3,682

Source: experiment artifacts generated from each benign training partition. Vocabulary fitting uses training graphs only, which causes the small dimensional variation across repetitions.

All views feed the same isolation-based ensemble. The estimator builds 500 trees, samples at most 256 training graphs per tree, uses all fitted features, and receives the repetition seed. Its anomaly score is the negative of the estimator's normality score, so larger values indicate stronger deviation. No standardization, supervised tuning, attack-informed contamination setting, or representation-specific parameter search is applied. Multi-scale contrastive work shows that multiple local views can expose different anomalies [15]; the present design uses a single fixed estimator to prevent that flexibility from obscuring the feature comparison.

3.3. Evaluation Design

A. Splits and Threshold Selection

Each of ten seeds independently shuffles the 100 graphs in each benign scenario. Per scenario, 60 graphs enter training, 15 enter validation, and 25 enter testing. Every repetition contains 300 benign training graphs, 75 benign validation graphs, and a test set of 125 benign plus 100 attack graphs. The attack graphs never affect vocabulary fitting, estimator fitting, or threshold selection. The decision threshold is the linearly interpolated 95th percentile of the 75 validation anomaly scores. Using the same indices for all three representations creates paired observations and removes split composition as an explanation for representation differences.

Table 3. Controlled evaluation protocol.

Component	Fixed setting
Repetition seeds	7, 19, 43, 101, 211, 307, 401, 503, 607, 809
Benign split per scenario	60 train, 15 validation, 25 test
Counts per repetition	300 train, 75 validation, 225 test
Test composition	125 benign graphs and 100 attack graphs
Threshold	95th percentile of benign validation scores
Estimator	500 trees; maximum sample size 256; all features
Primary metrics	AUPRC, F1, MCC, and FPR
Secondary metrics	AUROC, precision, recall, TP, FP, TN, and FN

Source: the executable experiment configuration. The test attack prevalence is 100/225 and is identical for every representation and repetition.

B. Metrics and Reproducibility

AUPRC is the primary ranking metric because it focuses on the positive attack class, while AUROC provides a prevalence-insensitive secondary view. Thresholded behavior is assessed with precision, recall, F1, MCC, and FPR. Accuracy is omitted from the main analysis because its interpretation changes with the chosen attack prevalence. Means and sample standard deviations summarize ten paired repetitions. Two-sided Wilcoxon signed-rank tests compare full typed order with static structure; zeros are removed by the standard Wilcoxon rule. The tests describe stability across sampled benign partitions, not independent attack campaigns.

The parser, feature cache, per-seed results, scenario error files, figures, and metadata are retained with the manuscript. Reproduction begins from the official archive whose hash appears in Section 3.1. It validates graph grouping before computing features and fits categorical vocabularies after each training split. The evaluation can be rerun without re-reading the 2.29 GB text file once the checksum-bound graph feature cache exists. This workflow keeps raw-data verification separate from repeated statistical evaluation.

4. Results and Analysis

4.1. Aggregate Detection Performance

A. Ranking Quality

Table 4 shows that static structure already separates every attack graph ahead of every benign test graph across all ten partitions: mean AUPRC and AUROC are both 1.0000 with zero observed standard deviation. The full typed-order view reaches AUPRC 0.9987±0.0017 and AUROC 0.9991±0.0012. These values remain high, yet they are lower than the static baseline by 0.0013

and 0.0009. The edge-type-only order view retains perfect ranking, indicating that the small loss appears after the high-dimensional typed-transition vocabulary is added. Node-context contrast can be an effective anomaly signal [16], but this graph-level corpus already contains a dominant static separation.

Table 4. Detection performance across ten paired repetitions.

Representation	AUPRC	AUROC	Precision	Recall	F1	MCC	FPR
Static	1.0000±0.0000	1.0000±0.0000	0.9299±0.0232	1.0000±0.0000	0.9635±0.0123	0.9345±0.0221	0.0608±0.0211
Edge order	1.0000±0.0000	1.0000±0.0000	0.9356±0.0316	1.0000±0.0000	0.9665±0.0170	0.9398±0.0305	0.0560±0.0295
Typed order	0.9987±0.0017	0.9991±0.0012	0.9402±0.0357	1.0000±0.0000	0.9689±0.0192	0.9441±0.0344	0.0520±0.0331

Source: metrics_by_seed.csv. Entries are mean±sample standard deviation, and thresholds are selected independently from benign validation scores.

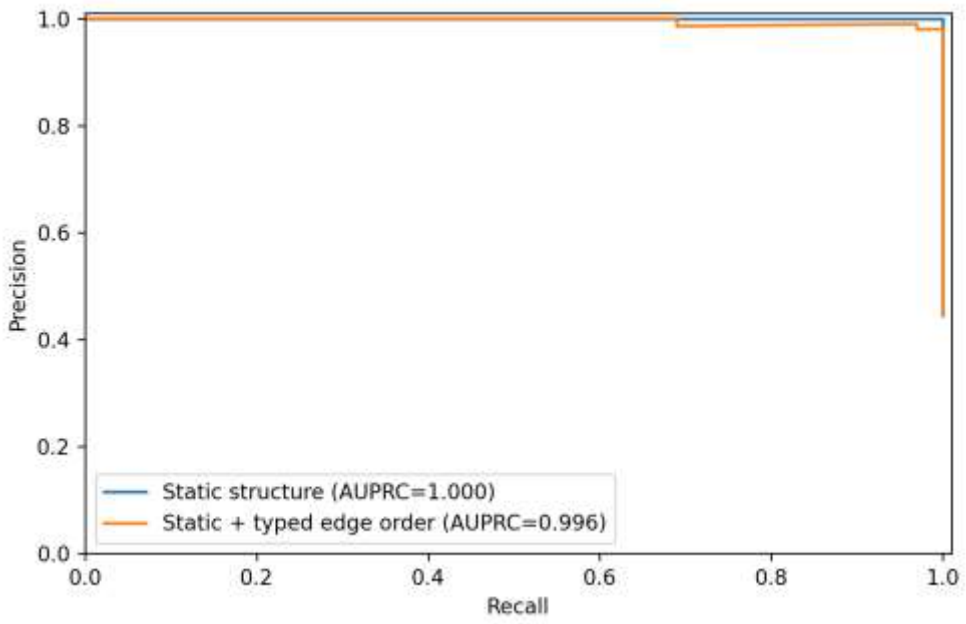


Figure 1. Precision-recall curves for the fixed seed-7 partition

The static representation attains AUPRC 1.000, while the full typed-order representation attains 0.996. Both curves sustain near-unit precision across most recall values, but typed order loses precision near complete recall. This partition is displayed for curve shape; Table 4 reports all ten repetitions.

B. Thresholded Decisions

Every representation detects all 100 attack graphs in every repetition, yielding recall 1.0000 and mean FN 0.0. Differences arise solely from benign false alarms. Static structure produces mean TP 100.0, FP 7.6, TN 117.4, and FN 0.0. Typed order produces TP 100.0, FP 6.5, TN 118.5, and FN 0.0. Its precision is 0.9402 rather than 0.9299, and its F1 is 0.9689 rather than 0.9635. Generative and contrastive objectives can combine attribute and structural evidence [17]; here a much simpler transition augmentation changes only 1.1 false decisions per 225-graph test set on average.

4.2. Stability and Paired Evidence

The mean thresholded changes favor typed order, but their variation is larger than their magnitude. F1 increases by 0.0053 with p=0.375, MCC increases by 0.0096 with p=0.375, and FPR decreases by 0.0088 with p=0.416. None supplies stable evidence of a threshold-level benefit across benign partitions. The AUPRC reduction of 0.0013 gives p=0.0156 because static AUPRC is exactly 1.0000 in every split and typed order is lower in each non-tied split. This statistical result indicates a consistent direction among nonzero differences, while its practical magnitude remains small.

Table 5. Paired comparison of full typed order and static structure.

Metric	Typed order minus static	SD of paired difference	Wilcoxon statistic	Two-sided p
AUPRC	-0.0013	0.0017	0.0	0.0156
AUROC	-0.0009	0.0012	0.0	0.0156
F1	+0.0053	0.0168	18.0	0.3750
MCC	+0.0096	0.0301	18.0	0.3750
FPR	-0.0088	0.0291	19.0	0.4160

Source: paired calculations from the ten rows per representation in metrics_by_seed.csv. Negative ranking differences favor static structure; negative FPR differences favor typed order.

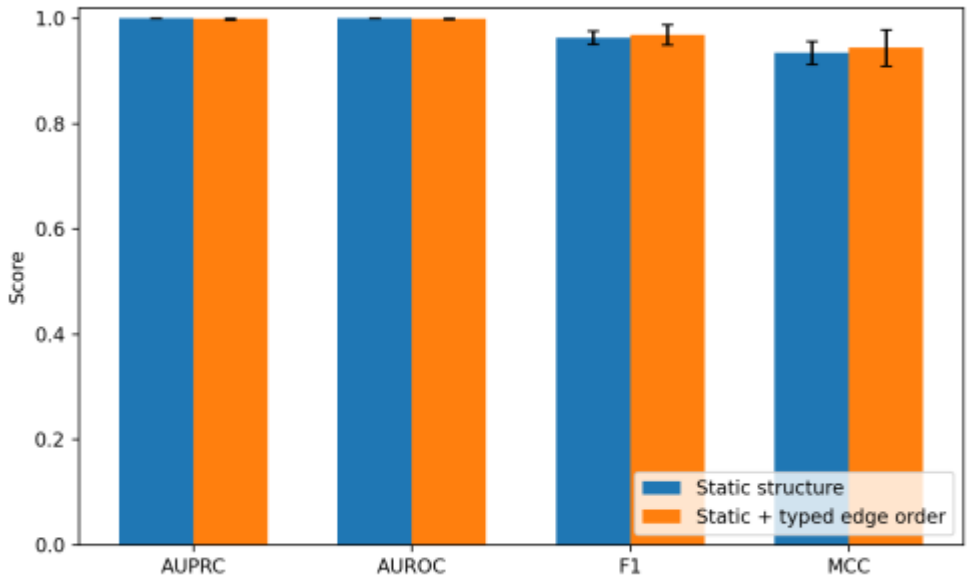


Figure 2. Aggregate detection metrics across ten paired repetitions

Static structure records AUPRC and AUROC of 1.0000±0.0000, compared with 0.9987±0.0017 and 0.9991±0.0012 for typed order. Typed order raises F1 from 0.9635±0.0123 to 0.9689±0.0192 and MCC from 0.9345±0.0221 to 0.9441±0.0344, while the wider error ranges show that these gains depend on the benign partition. One-class homophily research similarly stresses that an aggregate anomaly score can conceal local structure differences [18].

4.3. Scenario Errors and Representation Ablation

A. Benign-Scenario Error Distribution

False positives are not distributed uniformly. Static FPR is highest on Gmail at 0.156, followed by Download at 0.060, YouTube and CNN at 0.036 each, and VGame at 0.016. Typed order removes observed false positives from YouTube and VGame and reduces Download to 0.008 and CNN to 0.024, yet Gmail rises to 0.228. Gmail accounts for 5.7 of the 6.5 typed-order false positives per repetition, or 87.7%. Under static structure, it contributes 3.9 of 7.6, or 51.3%. Pseudo-anomaly generation can improve discrimination when only normal labels are known [19], but the present errors show that a single threshold interacts differently with each benign activity.

Table 6. Mean false-positive rate by benign scenario.

Benign scenario	Static FPR	Edge-order FPR	Typed-order FPR	Typed minus static
YouTube	0.036	0.000	0.000	-0.036
Gmail	0.156	0.276	0.228	+0.072
VGame	0.016	0.000	0.000	-0.016
Download	0.060	0.000	0.008	-0.052
CNN	0.036	0.004	0.024	-0.012

Source: scenario_fpr_summary.csv. Each scenario contributes 25 test graphs per repetition, so an FPR change of 0.040 corresponds to one mean false alarm.

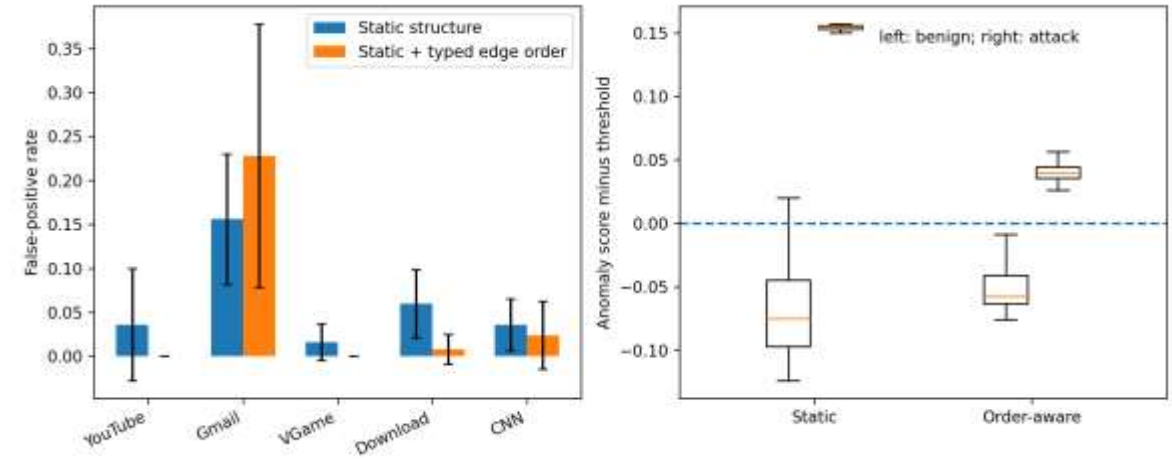


Figure 3. Scenario-specific false positives and seed-7 anomaly-score margins

Panel (a) reports mean false-positive rates and their split variability for the five benign activities. Typed order lowers four scenario rates but raises Gmail from 0.156 to 0.228. Panel (b) subtracts each representation's validation threshold from seed-7 test scores. Static benign and attack medians are -0.075 and 0.155, while typed-order medians are -0.058 and 0.040; the narrower typed-order margin is consistent with its slightly lower AUPRC.

B. Feature-Space Ablation

The coarse edge-order view is the strongest ranking-preserving augmentation. It uses 608.1 dimensions on average, roughly five times the static space, while retaining AUPRC and AUROC at 1.0000. Its F1 of 0.9665 and FPR of 0.0560 lie between the static and typed-order values. Adding typed transitions expands the mean dimension to 3,634.7, about 30 times the static dimension. F1 rises by another 0.0024 and FPR falls by 0.0040 relative to coarse order, while AUPRC drops to 0.9987. Tail-sensitive graph contrastive analysis shows that added neighborhood operations can redistribute performance across structural groups [20]. Our scenario audit reveals a related redistribution: lower error in four benign activities coexists with higher Gmail error.

The result rejects a simple monotonic relation between representation richness and detection quality. Static frequencies are sufficient to rank the released attack graphs perfectly. Coarse transitions preserve that ranking and slightly alter threshold calibration. Full typed transitions compress the score margin between attack and benign graphs, despite a small mean reduction in false alarms at one validation quantile. Given the single attack class and absent timestamps, the added space is not supported as a general replacement for static structure.

5. Discussion and Future Work

5.1. Interpretation and Practical Implications

The evidence favors static structure for the stated task. Its perfect ranking does not mean that drive-by-download detection is solved in realistic deployments. It means that this released attack scenario is strongly separable from the sampled benign activities using size, degree, and typed-edge composition. All three views detect every attack graph, and the remaining problem is calibration against benign variation. The typed-order view reduces mean FPR from 0.0608 to 0.0520, a difference of 0.0088 and 1.1 fewer false alarms among 125 benign test graphs, yet the paired p-value of 0.416 and the Gmail increase prevent a stable-benefit claim.

The negative ranking result is more informative than a headline comparison based only on F1. Static AUPRC is 1.0000, while full order reduces it to 0.9987. F1 moves in the opposite direction because each representation receives a separately estimated validation threshold. Ranking quality and threshold calibration answer different questions. A deployment with sufficient benign calibration data might prefer a compact static score and tune thresholds by activity class. An operator seeking one shared threshold should inspect scenario-conditioned errors before adding order features.

Representation cost also matters. The mean vocabulary grows from 121.0 to 3,634.7 dimensions, a 30-fold expansion, for no observed recall gain and a small, unstable FPR change. Coarse edge transitions offer a more restrained alternative at 608.1 dimensions, but even they do not improve AUPRC. In this setting, a simpler representation is easier to audit, store, and transfer across implementations. The study supports reporting compact baselines before attributing gains to richer sequence encoders.

5.2. Limitations and Future Work

The corpus contains one simulated drive-by-download mechanism and five controlled benign activities. Attack graphs are also much smaller in edge count than Download and CNN graphs, making static separation unusually easy. The test prevalence of 100 attacks among 225 graphs is useful for stable measurement but does not reflect operational prevalence. Precision values should not be transferred to production environments without recalibration.

The public preprocessing removes timestamps and collapses repeated block reads. Edge sequence preserves an ordering trace but cannot represent elapsed time, simultaneous activity, or causal delay. The analysis cannot evaluate continuous-time attention, time decay, or early detection. It also uses graph-level labels, so it cannot measure malicious-node localization or attack-path reconstruction. A stronger follow-up should use a timestamped provenance corpus, chronological train-test periods, several attack families, and node- or event-level ground truth.

Future work should retain the same controlled logic while expanding data coverage. Static and temporal encoders should share identical node and edge vocabularies, neighborhood budgets, training objectives, and threshold data. Evaluation should report ranking metrics, scenario-conditioned FPR, detection delay, memory, and throughput. Multiple attack campaigns are required before interpreting small metric differences as general temporal benefits. Until such evidence is available, the present result is confined to complete-graph drive-by-download detection on the verified public StreamSpot release.

References

[1] Manzoor, E., Milajerdi, S. M., & Akoglu, L. (2016). Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1035–1044). Association for Computing Machinery. https://doi.org/10.1145/2939672.2939783

[2] Meng, W., Liu, Y., Zhu, Y., Zhang, S., Pei, D., Liu, Y., Chen, Y., Zhang, R., Tao, S., Sun, P., & Zhou, R. (2019). LogAnomaly: Unsupervised detection of sequential and quantitative anomalies in unstructured logs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (pp. 4739–4745). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/658

[3] Wang, S., Wang, Z., Zhou, T., Sun, H., Yin, X., Han, D., Zhang, H., Shi, X., & Yang, J. (2022). THREATRACE: Detecting and tracing host-based threats in node level through provenance graph learning. *IEEE Transactions on Information Forensics and Security, 17*, 3972–3987. https://doi.org/10.1109/TIFS.2022.3208815

[4] Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems* (Vol. 30, pp. 1024–1034). Curran Associates.

[5] Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., & Achan, K. (2020). Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJeW1yHYwH

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

[7] Yu, W., Cheng, W., Aggarwal, C. C., Zhang, K., Chen, H., & Wang, W. (2018). NetWalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2672–2681). Association for Computing Machinery. https://doi.org/10.1145/3219819.3220024

[8] Zheng, L., Li, Z., Li, J., Li, Z., & Gao, J. (2019). AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN. In *Proceedings of the Twenty-Eighth International Joint Conference*

on Artificial Intelligence (pp. 4419–4425). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/614

[9] Cai, L., Chen, Z., Luo, C., Gui, J., Ni, J., Li, D., & Chen, H. (2021). Structural temporal graph neural networks for anomaly detection in dynamic graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 3747–3756). Association for Computing Machinery. https://doi.org/10.1145/3459637.3481955

[10] Liu, Y., Pan, S., Wang, Y. G., Xiong, F., Wang, L., Chen, Q., & Lee, V. C. S. (2023). Anomaly detection in dynamic graphs via Transformer. *IEEE Transactions on Knowledge and Data Engineering, 35*(12), 12081–12094. https://doi.org/10.1109/TKDE.2021.3124061

[11] Tian, S., Dong, J., Li, J., Zhao, W., Xu, X., Wang, B., Song, B., Meng, C., Zhang, T., & Chen, L. (2023). SAD: Semi-supervised anomaly detection on dynamic graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence* (pp. 2306–2314). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2023/256

[12] Yang, X., Zhao, X., & Shen, Z. (2025). A generalizable anomaly detection method in dynamic graphs. *Proceedings of the AAAI Conference on Artificial Intelligence, 39*(20), 22001–22009. https://doi.org/10.1609/aaai.v39i20.35508

[13] Li, J., Dani, H., Hu, X., & Liu, H. (2017). Radar: Residual analysis for anomaly detection in attributed networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence* (pp. 2152–2158). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2017/299

[14] Peng, Z., Luo, M., Li, J., Liu, H., & Zheng, Q. (2018). ANOMALOUS: A joint modeling approach for anomaly detection on attributed networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence* (pp. 3513–3519). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2018/488

[15] Jin, M., Liu, Y., Zheng, Y., Chi, L., Li, Y.-F., & Pan, S. (2021). ANEMONE: Graph anomaly detection with multi-scale contrastive learning. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 3122–3126). Association for Computing Machinery. https://doi.org/10.1145/3459637.3482057

[16] Liu, Y., Li, Z., Pan, S., Gong, C., Zhou, C., & Karypis, G. (2022). Anomaly detection on attributed networks via contrastive self-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems, 33*(6), 2378–2392. https://doi.org/10.1109/TNNLS.2021.3068344

[17] Zheng, Y., Jin, M., Liu, Y., Chi, L., Phan, K. T., & Chen, Y.-P. P. (2023). Generative and contrastive self-supervised learning for graph anomaly detection. *IEEE Transactions on Knowledge and Data Engineering, 35*(12), 12220–12233. https://doi.org/10.1109/TKDE.2021.3119326

[18] Qiao, H., & Pang, G. (2023). Truncated affinity maximization: One-class homophily modeling for graph anomaly detection. In *Advances in Neural Information Processing Systems* (Vol. 36). Curran Associates.

[19] Qiao, H., Wen, Q., Li, X., Lim, E.-P., & Pang, G. (2024). Generative semi-supervised graph anomaly detection. In *Advances in Neural Information Processing Systems* (Vol. 37). Curran Associates.

[20] Xu, Y., Peng, Z., Shi, B., Hua, X., Dong, B., Wang, S., & Chen, C. (2025). Revisiting graph contrastive learning on anomaly detection: A structural imbalance perspective. *Proceedings of the AAAI Conference on Artificial Intelligence, 39*(12), 12972–12980. https://doi.org/10.1609/aaai.v39i12.33415