SIADS: Innovation-Based Anomaly Detection in Streaming Communication Graphs

Tanish Verma
Indian Institute of Technology Dharwad
Dharwad, India
cs24mt014@iitdh.ac.in

***Abstract*—Detecting anomalous interactions in streaming communication networks is a fundamental challenge in information security and network forensics. Existing methods rely on structural heuristics such as frequency sketches, random walk proximity, or matrix factorization, which degrade when attack intensity naturally drifts over time. We propose SIADS (Streaming Innovation-Based Anomaly Detection System), a fully online method that scores each incoming edge by the *innovation* it induces in learned node representations, directly borrowing the Kalman filter notion of residual-normalized-by-running-variance. A lightweight MLP encoder with gated aggregation maintains compact node states, while per-node and per-edge Exponential Moving Averages (EMA) normalize the anomaly score against historical graph evolution. In the reported setup, SIADS uses fixed randomly initialized weights, operates in a prequential snapshot-streaming setting, and adapts to distributional drift without retraining. We evaluate on three real-world communication network datasets under both uniform and time-window drift injection scenarios against four competitive streaming baselines. SIADS achieves up to 22.3% higher AUC and 3.4× higher Average Precision than the best baseline, while maintaining throughput of up to 47K edges/second.**

***Index Terms*—Anomaly Detection, Streaming Graphs, Network Security, Innovation Scoring, Graph Neural Networks**

I. Introduction

Modern communication networks, including social messaging platforms, organizational email systems, and internet routing infrastructures, generate continuous streams of interactions modeled as temporal edge streams over graphs. Detecting anomalous or malicious edges within these streams is central to network security, intrusion detection, and forensic analysis [1], [2]. An anomalous edge is one that would not be expected given the historical interaction structure: a novel message link, an unexpected routing path, or communication between previously unconnected parties [3].

Effective anomaly detection in this setting must satisfy three requirements simultaneously. First, anomaly labels are unavailable at detection time, so the detector must operate unsupervised. Second, anomaly rates drift continuously as attack campaigns intensify, pause, and re-emerge [4], and methods that assume a stationary anomaly distribution degrade under realistic conditions. Third, large-scale networks demand high-throughput inference to keep pace with the stream.

Existing streaming detectors address only parts of this challenge. Frequency-based methods [5] and distributional sketch methods [6] are fast but insensitive to structural novelty. Random walk methods [7] are structurally expressive but orders of magnitude slower. Factorization methods [8] capture temporal interaction rates but do not adapt to drift. Deep learning approaches [9]–[11] achieve strong detection but require an offline training phase, precluding fully-streaming deployment. No existing streaming method satisfies all three requirements simultaneously.

We propose **SIADS** (Streaming Innovation-Based Anomaly Detection System), a fully online edge anomaly detector grounded in the Kalman filter innovation principle [12]. For each arriving edge, SIADS measures the change it induces in learned node state representations and normalizes this change by a per-node running Exponential Moving Average (EMA) scale. This self-calibrating innovation score adapts automatically as typical graph dynamics evolve, providing drift robustness without retraining.

Our contributions are:

1) An innovation-based edge scoring mechanism where each state change is normalized by its per-node EMA scale, yielding scores that are self-calibrated across nodes with varying activity levels.
2) Complementary node and edge innovation signals capturing different aspects of anomalous behavior: behavioral novelty at the endpoint level and interaction intensity deviation at the pair level.
3) A fully-online, no-training streaming architecture with bounded memory that naturally adapts to concept drift without parameter updates.
4) An empirical demonstration that SIADS achieves the best detection quality across all evaluated methods on three real-world datasets under both injection scenarios, while maintaining practical throughput for real-time deployment.

II. Related Work

Streaming graph anomaly detection. Graph anomaly detection has been surveyed by Akoglu et al. [1] and, for dynamic graphs, by Ekle and Eberle [2]. Early work by Aggarwal et al. [3] formalized outlier detection in graph streams using sketch-based summaries. Count-Min Sketch [5] flags bursty edges but misses structural anomalies. SpotLight [6] operates at the snapshot level and misses individual anomalous edges. SedanSpot [7] scores edges via Personalized PageRank; it is structurally expressive but slow. F-FADE [8] captures temporal

frequency patterns but does not adapt to distributional shift. StreamSpot [13] addresses heterogeneous streaming graphs; our injection protocol follows the unseen-edge model of NetWalk [14].

Deep learning methods for dynamic graph anomaly. Several recent methods apply deep learning to dynamic graph anomaly detection but require an offline training phase. AddGraph [9] uses an attention-based temporal GCN trained on historical snapshots. TADDY [10] applies a transformer over temporal graphs. RustGraph [11] jointly learns structural and temporal dependencies. SLADE [15] is self-supervised but trained on a chronological prefix, making it incompatible with fully-streaming prequential evaluation.

Temporal graph representation learning. Methods such as JODIE [16], DyRep [17], TGAT [18], and TGN [19] develop temporal node representations but target downstream prediction tasks rather than anomaly detection. SIADS adapts the state-memory intuition from this line of work for unsupervised, real-time scoring.

III. Proposed Method: SIADS

Fig. 1 illustrates the SIADS pipeline, organized into four stages: node states and edge features are retrieved from State Memory and encoded by a fixed MLP with gated neighborhood aggregation; the aggregated representation drives a tanh RNN state update; the resulting state change and embedding norm are EMA-normalized into complementary Node and Edge Innovation scores; and the two scores are combined into a final anomaly score.

Notation. Each node u maintains a state vector $\mathbf{s}_u \in \mathbb{R}^d$, initialized to $\mathbf{0}$, and a scalar EMA innovation scale σ_u, initialized to 1. Each observed pair (u, v) maintains an EMA edge magnitude ρ_{uv}, initialized to 1. All weight matrices $(\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_a, \mathbf{W}_s, \mathbf{W}_x)$ are randomly initialized and kept fixed throughout the stream.

A. State Memory and Input

SIADS maintains a persistent lookup table of node states; upon edge (u, v, t) arrival, $\mathbf{s}_u$ and $\mathbf{s}_v$ are retrieved (initialized to $\mathbf{0}$ for first-time nodes) along with edge features $\mathbf{f}_{uv}$. After the recurrent update (Section III-D), updated states are written back, forming the feedback loop in Fig. 1.

B. MLP Encoder

The retrieved state vectors and edge features are concatenated into a joint input and passed through a two-layer MLP ϕ to produce a shared edge embedding $\mathbf{h}_{uv}$:

$$\mathbf{h}_{uv} = \phi\big([\mathbf{s}_u \| \mathbf{s}_v \| \mathbf{f}_{uv}]\big) = \mathbf{W}_2\,\mathrm{ReLU}(\mathbf{W}_1[\mathbf{s}_u \| \mathbf{s}_v \| \mathbf{f}_{uv}] + \mathbf{b}_1) + \mathbf{b}_2 \tag{1}$$

Concatenating both endpoint states, rather than averaging them, preserves directionality: a message from an active hub to a newly seen node produces a structurally distinct embedding from the reverse direction, allowing SIADS to capture asymmetric anomaly patterns.

C. Gated Neighborhood Aggregation

Within each snapshot, multiple edges may arrive simultaneously, each contributing a message to shared endpoint nodes. To aggregate these messages, SIADS computes a per-dimension sigmoid gate from each edge embedding and performs a normalized weighted sum:

$$\boldsymbol{\omega}_{uv} = \sigma(\mathbf{W}_a \mathbf{h}_{uv}), \quad \mathbf{x}_u^t = \frac{\sum_{v \in \mathcal{N}_u^t} \boldsymbol{\omega}_{uv} \odot \mathbf{h}_{uv}}{\sum_{v \in \mathcal{N}_u^t} \boldsymbol{\omega}_{uv}} \tag{2}$$

where $\sigma(\cdot)$ denotes element-wise sigmoid and $\mathcal{N}_u^t$ is the set of neighbors of u in the current snapshot. Per-dimension gating provides selective weighting of each edge's contribution, which is important for hub nodes that interact simultaneously with many peers of varying relevance [18].

D. Recurrent State Update

The aggregated neighborhood representation $\mathbf{x}_u^t$ is used to update node u's persistent state via a single-layer tanh recurrent cell:

$$\mathbf{s}_u^t = \tanh\big(\mathbf{W}_s\,\mathbf{s}_u^{t-1} + \mathbf{W}_x\,\mathbf{x}_u^t\big) \tag{3}$$

The tanh nonlinearity constrains states to $(-1, 1)^d$ regardless of stream length, preventing unbounded growth in node representations. The updated state $\mathbf{s}_u^t$ is written back to State Memory before the next snapshot is processed.

E. Innovation Scoring

SIADS computes two complementary EMA-normalized innovation scores that together characterize how anomalous the current interaction is.

Node Innovation. The ℓ_2 magnitude of the state change measures how much node u's representation shifted after processing the current snapshot. Following the Kalman filter innovation principle [12], this raw residual is normalized by its own running EMA:

$$\delta_u^t = \|\mathbf{s}_u^t - \mathbf{s}_u^{t-1}\|_2, \tag{4}$$

$$\sigma_u^t = \alpha\,\sigma_u^{t-1} + (1-\alpha)\,\delta_u^t, \tag{5}$$

$$z_u^t = \delta_u^t \,/\, (\sigma_u^t + \varepsilon) \tag{6}$$

The normalized score z_u^t answers not "how much did this node change?" but "how much relative to its own typical change?", making the signal comparable across nodes with very different activity levels.

Edge Innovation. In parallel, the ℓ_2 norm of $\mathbf{h}_{uv}$ captures interaction intensity and is EMA-normalized per stored pair to detect deviations from its own history:

$$m_{uv}^t = \|\mathbf{h}_{uv}^t\|_2, \tag{7}$$

$$\rho_{uv}^t = \beta\,\rho_{uv}^{t-1} + (1-\beta)\,m_{uv}^t, \tag{8}$$

$$z_{uv}^t = m_{uv}^t \,/\, (\rho_{uv}^t + \varepsilon) \tag{9}$$

The node scores z_u^t, z_v^t capture *behavioral novelty* at the endpoint level; z_{uv}^t captures *interaction intensity deviation* at the pair level. These signals are complementary: a node may behave abnormally without an unusual pair interaction, and vice versa.

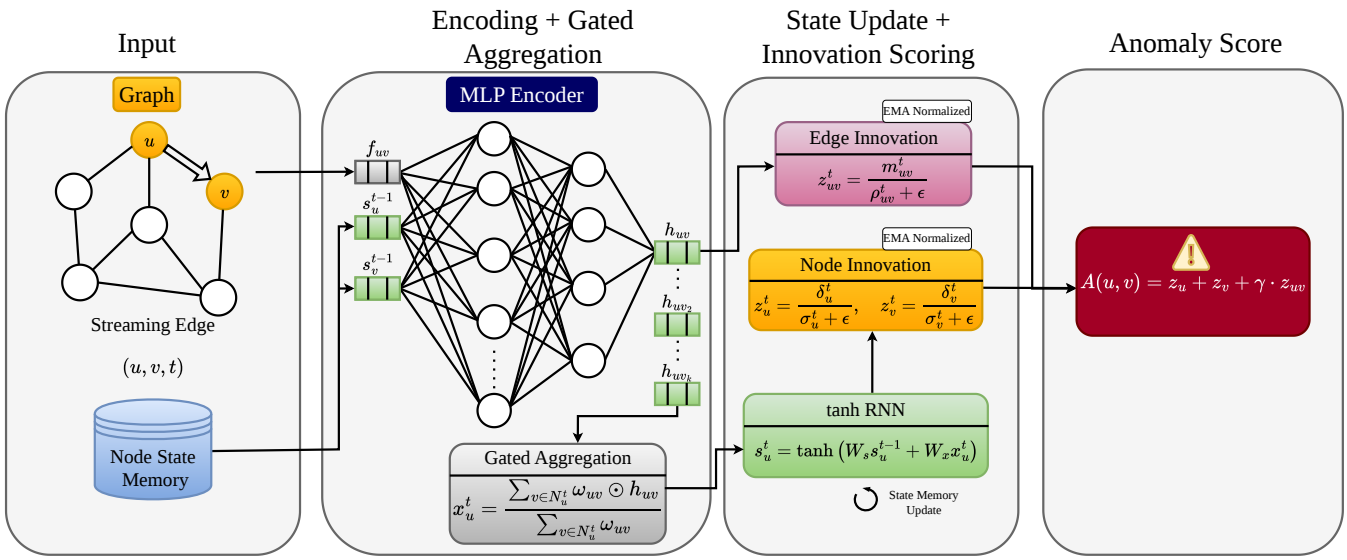

Fig. 1. SIADS pipeline for a streaming edge (u, v, t). **Input**: prior node states $\mathbf{s}_u^{t-1}, \mathbf{s}_v^{t-1}$ and edge features $\mathbf{f}_{uv}$ are retrieved from Node State Memory. **Encoding + Gated Aggregation**: a fixed MLP encoder produces edge embeddings $\mathbf{h}_{uv}$; gated aggregation pools neighbor embeddings into $\mathbf{x}_u^t$. **State Update + Innovation Scoring**: the tanh RNN updates node state $\mathbf{s}_u^t$; the EMA-normalized state change yields Node Innovation z_u^t, z_v^t, and the EMA-normalized embedding norm yields Edge Innovation z_{uv}^t. **Anomaly Score**: $\mathcal{A}(u, v) = z_u + z_v + \gamma \cdot z_{uv}$.

F. Anomaly Score

The final anomaly score combines both signals:

$$\mathcal{A}(u, v)^t = z_u^t + z_v^t + \gamma \cdot z_{uv}^t \tag{10}$$

The weight $\gamma \geq 0$ controls the contribution of the pair-level signal; setting γ=0 reduces the score to pure node innovation. Each edge is scored using node states from before the current snapshot update, ensuring strict prequential ordering with no lookahead. The ablation in Section V confirms that γ>0 consistently improves detection, particularly on datasets where pairwise interaction history is a strong anomaly signal.

IV. Experimental Setup

A. Datasets

We evaluate on three real-world temporal communication networks drawn from KONECT [20]: UCI Messages [21], Email-DNC [20], and AS-Topology [22]. Together they cover social messaging, email communication, and internet routing, spanning both medium-scale dense interaction graphs and a large sparse topology graph. Table I summarizes their statistics.

TABLE I
Dataset Statistics

Dataset	Nodes	Edges	Domain
UCI Messages	1,899	59,835	Student social messages
Email-DNC	1,891	43,546	Political email network
AS-Topology	34,761	214,819	Internet AS routing

B. Anomaly Injection Protocol

Since ground-truth anomalies are not available in these datasets, we inject synthetic anomalies following the widely adopted *unseen-edge* model [7], [14]: random node pairs that have never co-occurred in the observed stream are inserted as anomalous edges, making them structurally incongruent with historical graph structure.

We evaluate under two scenarios:

- **Baseline**: Uniform 10% anomaly injection rate throughout the stream.
- **Time-Window Drift (TW-Drift)**: Varying injection rates across five equal-length windows: $5\% \rightarrow 15\% \rightarrow 2\% \rightarrow 20\% \rightarrow 8\%$. This simulates realistic attack campaigns that intensify, pause, and re-emerge over time.

All experiments use 3 random seeds (42, 52, 62); results are reported as mean $\pm$ std.

C. Baselines and Evaluation

We compare against **CMS** [5], **SpotLight** [6], **SedanSpot** [7], and **F-FADE** [8], re-implementing each baseline's core scoring mechanism in our unified prequential framework where every edge is scored before updating internal state. SIADS hyperparameters are fixed across all experiments: state dimension d=128, EMA decay rates α=β=0.88, score weight γ=1.0, and edge-scale table budget 200K entries, giving an effective innovation window of $1/(1-\alpha) \approx 8$ snapshots. We report **AUC-ROC**, **Average Precision (AP)**, and **throughput** (edges/sec).

TABLE II
AUC-ROC (↑) AND AVERAGE PRECISION (↑) ACROSS DATASETS AND SCENARIOS (MEAN ± STD OVER 3 SEEDS). BEST RESULT PER COLUMN IN **BOLD**.

Method	UCI Messages				Email-DNC				AS-Topology			
	Baseline		TW-Drift		Baseline		TW-Drift		Baseline		TW-Drift	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
SIADS (ours)	**.915** ±.005	**.426** ±.045	**.904** ±.004	**.436** ±.052	**.953** ±.013	**.597** ±.009	**.933** ±.019	**.572** ±.017	**.855** ±.005	**.392** ±.120	**.860** ±.010	**.413** ±.123
SedanSpot	.877 ±.000	.286 ±.000	.877 ±.001	.300 ±.002	.934 ±.000	.427 ±.002	.932 ±.000	.429 ±.000	.632 ±.000	.115 ±.000	.632 ±.000	.118 ±.000
F-FADE	.804 ±.001	.172 ±.001	.804 ±.001	.183 ±.001	.748 ±.001	.142 ±.000	.747 ±.001	.145 ±.000	.406 ±.002	.067 ±.000	.406 ±.003	.069 ±.000
CMS	.580 ±.002	.141 ±.001	.573 ±.001	.147 ±.000	.816 ±.001	.277 ±.001	.816 ±.001	.283 ±.001	.472 ±.001	.079 ±.000	.465 ±.000	.080 ±.000
SpotLight	.359 ±.006	.061 ±.000	.365 ±.005	.066 ±.000	.314 ±.007	.058 ±.001	.318 ±.007	.060 ±.001	.630 ±.009	.104 ±.003	.631 ±.010	.108 ±.003

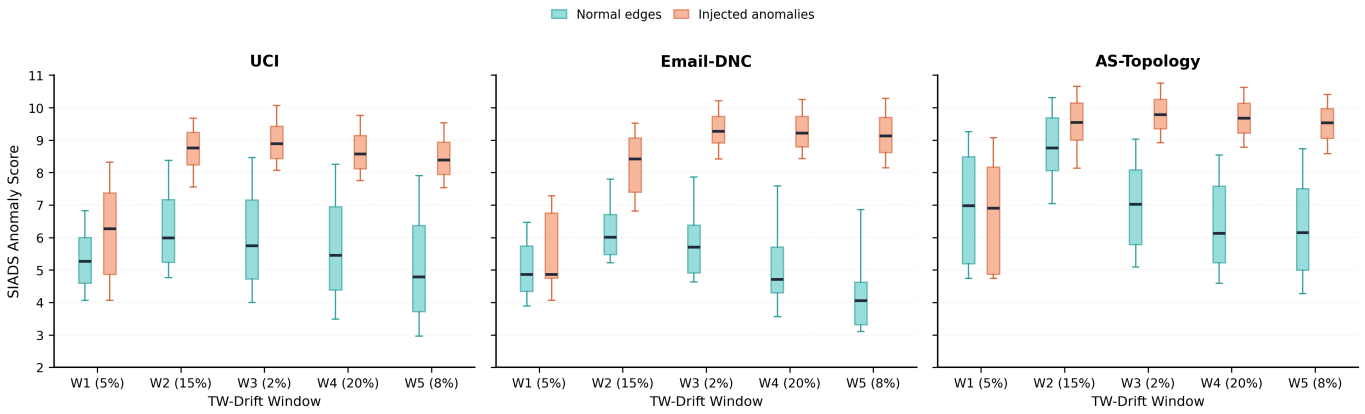

Fig. 2. Distribution of SIADS anomaly scores for normal and injected anomalous edges across the five TW-Drift windows on UCI, Email-DNC, and AS-Topology. Each window shows paired box plots for normal and anomalous edges; higher anomaly boxes indicate stronger score separation. Separation is weakest in the earliest window and strongest on Email-DNC, while remaining visible across all three datasets.

V. RESULTS AND DISCUSSION

A. Detection Performance

Table II presents the full AUC-ROC and AP results. SIADS achieves the highest AUC and AP on every dataset in both scenarios. The most significant improvement is on **AS-Topology**, where SIADS achieves AUC of 0.855 vs. the next-best 0.632 (SedanSpot), a margin of **22.3 percentage points (pp)**. On Email-DNC, SIADS reaches AP of 0.597, compared to 0.427 for SedanSpot, a **40% relative improvement**. CMS, SpotLight, and F-FADE rank lowest because they target frequency bursts, snapshot-level changes, and temporal interaction rates respectively, none of which capture the structural novelty of the unseen-edge injection model; SedanSpot, the one structurally expressive baseline, achieves the second-best results across most settings.

Notably, the TW-Drift scenario does not meaningfully harm SIADS performance. AUC changes by less than 1.5 pp across all three datasets, demonstrating inherent adaptivity to concept drift via EMA normalization. Fig. 2 complements these aggregate metrics by showing SIADS score distributions for normal and injected anomalous edges across the five TW-Drift windows. In most windows, anomalous edges receive higher scores than normal edges, with the clearest separation on Email-DNC; the weaker separation in the earliest window is consistent with SIADS's cold-start behavior before EMA scales stabilize.

B. Ablation Study

Table III isolates each component's contribution (mean ± std over 3 seeds, Baseline scenario). Removing the edge score z_{uv}^t causes the largest drop, especially on UCI (.915→.611 AUC) and Email-DNC (.953→.776), showing that pair-specific interaction intensity is a critical signal. Removing EMA normalization preserves some average performance but greatly increases variance across seeds, indicating that per-node calibration is important for stable operation.

TABLE III
COMPONENT ABLATION (BASELINE, MEAN ± STD OVER 3 SEEDS).

Variant	UCI		Email-DNC		AS-Topology	
	AUC	AP	AUC	AP	AUC	AP
Full SIADS	**.915** ±.005	**.426** ±.045	**.953** ±.013	**.597** ±.009	**.855** ±.005	**.392** ±.120
No-Edge (γ=0)	.611 ±.036	.119 ±.021	.776 ±.012	.269 ±.006	.846 ±.027	.315 ±.097
No-EMA ($\sigma\equiv$1)	.839 ±.062	.459 ±.042	.812 ±.140	.519 ±.210	.771 ±.141	.372 ±.187

C. Throughput Analysis

Table IV reports throughput in edges per second. There is a clear inverse trend across methods: faster sketch methods (CMS, SpotLight) achieve the highest throughput but the lowest AUC and AP, while structurally expressive methods (SedanSpot, F-FADE) are slower but achieve higher detection quality. SIADS occupies the best position in this trade-off: it **strictly dominates SedanSpot** on all datasets, with >20×

higher throughput while also outperforming it on every detection metric. Against F-FADE, SIADS consistently achieves higher detection quality across all datasets; on UCI and Email-DNC it is also faster, though F-FADE has marginally higher throughput on AS-Topology (40K vs. 32K edges/sec) at the cost of dramatically lower AUC (0.406 vs. 0.855). Compared to CMS and SpotLight, SIADS is slower due to its MLP computation but achieves substantially higher detection quality, processing over 9K edges/sec on all datasets, sufficient for real-time network monitoring. This efficiency follows from SIADS's design: each edge incurs a single $O(d^2)$ MLP forward pass with no graph traversal, node memory is $O(|\mathcal{V}|)$, and the scalar EMA scale reduces per-node storage from $O(d^2)$ to $O(1)$ compared to a full covariance approach.

TABLE IV
THROUGHPUT IN EDGES/SECOND (BASELINE SCENARIO).

Method	UCI	Email-DNC	AS-Topology
CMS	28,860	96,161	94,027
SpotLight	14,450	67,443	69,602
SIADS	**9,594**	**46,761**	**32,055**
F-FADE	7,477	25,555	40,447
SedanSpot	157	1,069	869

VI. CONCLUSION

We presented SIADS, a streaming edge anomaly detector grounded in the Kalman filter innovation principle. The central idea is to normalize each state change by its per-node running variance rather than comparing raw magnitudes, yielding a detector that is simultaneously self-calibrating across nodes of different activity levels and robust to concept drift without retraining. On three real-world communication datasets, SIADS consistently outperforms four streaming baselines across all metrics and both injection scenarios, with the largest gains on AS-Topology where structural novelty is the dominant anomaly signal. One known limitation is a cold-start window of approximately $1/(1-\alpha)$ snapshots before EMA scales converge; edges in this early window receive partially suppressed scores regardless of their anomalousness. Future work includes extending SIADS to attributed and heterogeneous graphs [13] and developing warm-start initialization strategies to eliminate the cold-start window.

REFERENCES

[1] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015. [Online]. Available: https://doi.org/10.1007/s10618-014-0365-y

[2] O. A. Ekle and W. Eberle, "Anomaly detection in dynamic graphs: A comprehensive survey," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 8, pp. 1–44, 2024. [Online]. Available: https://doi.org/10.1145/3669906

[3] C. C. Aggarwal, Y. Zhao, and P. S. Yu, "Outlier detection in graph streams," in *2011 IEEE 27th International Conference on Data Engineering*. IEEE, 2011, pp. 399–409. [Online]. Available: https://doi.org/10.1109/ICDE.2011.5767885

[4] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 44:1–44:37, 2014. [Online]. Available: https://doi.org/10.1145/2523813

[5] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," *Journal of Algorithms*, vol. 55, no. 1, pp. 58–75, 2005.

[6] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra, "SpotLight: Detecting anomalies in streaming graphs," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2018, pp. 1378–1386.

[7] D. Eswaran and C. Faloutsos, "SedanSpot: Detecting anomalies in edge streams," in *Proceedings of the 18th IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 953–958.

[8] C. Chang, W.-Y. Wang, W.-C. Peng, and T.-F. Chen, "F-FADE: Frequency factorization for anomaly detection in edge streams," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2021, pp. 589–597.

[9] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 4419–4425.

[10] Y. Liu, S. Pan, Y. G. Wang, F. Xiong, L. Wang, Q. Chen, and V. C. S. Lee, "Anomaly detection in dynamic graphs via transformer," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 12 081–12 094, 2023. [Online]. Available: https://doi.org/10.1109/TKDE.2021.3124061

[11] J. Guo, S. Tang, J. Li, K. Pan, and L. Wu, "RustGraph: Robust anomaly detection in dynamic graphs by jointly learning structural-temporal dependency," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 7, pp. 3472–3485, 2024. [Online]. Available: https://doi.org/10.1109/TKDE.2023.3328645

[12] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960. [Online]. Available: https://doi.org/10.1115/1.3662552

[13] E. Manzoor, S. M. Milajerdi, and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2016, pp. 1035–1044.

[14] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "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 (KDD)*. ACM, 2018, pp. 2672–2681.

[15] J. Kim, M. Jeon, S. Han, J. Sung, and C. Park, "SLADE: A self-supervised learning framework for anomaly detection in dynamic graphs," in *Proceedings of the 30th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2024.

[16] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 1269–1278.

[17] R. Trivedi, M. Farajtabar, P. Biswal, and H. Zha, "DyRep: Learning representations over dynamic graphs," in *International Conference on Learning Representations*, 2019.

[18] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," in *International Conference on Learning Representations*, 2020.

[19] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, "Temporal graph networks for deep learning on dynamic graphs," *arXiv preprint arXiv:2006.10637*, 2020.

[20] J. Kunegis, "KONECT – the koblenz network collection," in *Proceedings of the 22nd International Conference on World Wide Web Companion*. Association for Computing Machinery, 2013, pp. 1343–1350. [Online]. Available: https://doi.org/10.1145/2487788.2488173

[21] P. Panzarasa, T. Opsahl, and K. M. Carley, "Patterns and dynamics of users' behaviour and interaction: Network analysis of an online community," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 5, pp. 911–932, 2009.

[22] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, pp. 2:1–2:41, 2007. [Online]. Available: https://doi.org/10.1145/1217299.1217301