BEAT THE COUNTER FIRST: A BASELINE FOR TEMPORAL-GRAPH ANOMALY MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Progress in streaming, edge-level graph anomaly detection (GAD) has been marked by increasingly elaborate architectures, from count-min-sketch chi-square tests to memory-augmented attention networks. Yet the empirical gains attributable to this added complexity have not been systematically evaluated. We propose SIMPLECOUNT, a reference with no learning- or gradient-descent-based parameter fitting that selects one scalar feature per dataset from a fixed pool of counts, recencies, first-occurrence indicators, and count-derived transforms. We compare SIMPLECOUNT with two temporal-graph detector models and an ISOFOREST control fitted to the complete feature vector across five public datasets and one synthetic dataset. SIMPLECOUNT *matches or exceeds SLADE - a state-of-the-art GAD method - on three of six datasets and exceeds ISOFOREST on all six*. We report paired statistical tests and five-seed SLADE evaluations. SLADE requires *23 to 133× more wall-clock time* than SIMPLECOUNT. On synthetic datasets, pre-event structural scores recover the planted signal at AUC up to 0.955, while all evaluated GAD models remain near random. SIMPLECOUNT is not a replacement GAD model, rather, its purpose is to highlight that the benefit of complexity is dataset-dependent, and every claimed gain should be reported against a strong one-feature reference together with its compute cost.

Keywords: temporal graphs, transformers, dynamic graph anomaly detection, graph attenton, shortcut learning, simple baselines, compute-efficient graph learning, benchmark evaluation.

1 INTRODUCTION

Streaming-graph anomaly detection - the task of flagging suspicious edges (u, v, t) as they arrive in a temporal interaction graph - has followed a familiar trajectory in the literature: each method enriches the per-edge representation, from count-min-sketch chi-square tests (MIDAS-F Bhatia et al. (2022)) through temporal-graph attention (TGAT Xu et al. (2020)) and memory networks (TGN Rossi et al. (2020)), to recent self-supervised detector models such as SLADE Lee et al. (2024). While the improvements are real, how much of each gain comes from the added architectural complexity is unclear.

Elsewhere in graph learning, strong baselines have challenged similar gains. EDGEBANK Poursafaei et al. (2022) showed that a simple memorization rule—score 1 if an edge has been seen before, 0 otherwise—rivals TGN, TGAT, CAWN, and JODIE on temporal link prediction. An MLP mixer Cong et al. (2023) later matched attention-heavy models on standard temporal link-prediction benchmarks. GADBench Tang et al. (2023) found that tree ensembles with simple neighbourhood aggregation outperform specialised GNN anomaly detectors on *static* graphs. All three results concern tasks like link prediction and static node-level anomaly detection, where recurring edges or structural patterns reward low-complexity features. Neither settles the case for *temporal edge-level* anomaly detection, where labels target atypical dynamic behaviour rather than future-edge existence, and where the temporal dimension might plausibly demand the full machinery of attention and memory.

SIMPLECOUNT puts that assumption to a direct test. We compare the count-sketch-based MIDAS-F and self-supervised temporal-graph attention network SLADE against a low-complexity reference

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

that selects one scalar feature per dataset. We also fit IsoForest to the complete 14-feature vector to test whether a nonlinear combination improves over selecting one feature. Note that SimpleCount is not intended as an off-the-shelf model; it is a diagnostic reference meant to understand the utility of a complex model on a given dataset. Comparing learned models against SimpleCount reveals the incremental predictive value delivered by their learned representations and the compute required to obtain it. We report a separate two-dataset GraphMixer check (Cong et al., 2023) because its downstream classifier uses anomaly labels.

Our results connect to a growing body of work, including an analysis by Heeg et al. (2025) showing that temporal GNNs can fail to distinguish temporal structures. While strong performance from simple endpoint and pair features is consistent with *shortcut learning* (Geirhos et al., 2020), it does not establish that SLADE or other complex models rely only on such shortcuts. It shows instead that these benchmarks do not require deeper temporal structure for strong performance.

Our central finding is that a reference with no learning- or gradient-descent-based parameter fitting is competitive with a state-of-the-art temporal-graph detector model at one to two orders of magnitude less compute. Concretely, we show that:

- From a performance perspective, across five public datasets and one synthetic dataset, SimpleCount matches or beats SLADE on three datasets and exceeds IsoForest on all six in terms of AUC.
- From a computational cost perspective, SimpleCount uses 23–133$\times$ less wall time than SLADE, with a mean of 72$\times$ across all six datasets.
- From a simplicity and explainability perspective, SimpleCount detections are directly traceable to an observed count, recency, or first-occurrence value, without requiring an explanation method such as SHAP (Lundberg and Lee, 2017). Models based on feature ensembles or learned latent node states do not provide this one-feature attribution.

Our central claim is not one of suggesting yet another new model, but that **the benefit of complexity is dataset-dependent, and every claimed gain should be reported against a strong one-feature reference together with its compute cost.** We distil the audit into five recommendations for benchmark authors and reviewers in Appendix A.

2 Related Work

This work connects two lines of research: temporal-graph anomaly detection and benchmark studies showing the value of strong simple baselines. We apply this baseline-first perspective to temporal edge-level anomaly detection and add paired inference and compute-cost accounting.

Temporal-graph anomaly detection. The streaming-graph anomaly-detection subfield has produced MIDAS-F Bhatia et al. (2020; 2022) (count-min-sketch chi-square), SedAnSpot Eswaran and Faloutsos (2018), FFADE Chang et al. (2021), and most recently SLADE Lee et al. (2024), which combines TGN-style node memory Rossi et al. (2020) with a GRU updater, a TGAT memory generator Xu et al. (2020), and node-wise self-supervised objectives. It is formulated for dynamic node anomaly detection; its reference evaluator produces one anomaly score per interaction from the source node's memory-recovery and memory-drift similarities. The SLADE study evaluates Wikipedia, Reddit, Bitcoin-Alpha, and Bitcoin-OTC Kumar et al. (2019; 2016).

Simple baselines. The closest prior result is EdgeBank Poursafaei et al. (2022), which showed that a hash-set of seen edges matches or outperforms TGN, TGAT, CAWN, and JODIE on multiple temporal link-prediction benchmarks. In our setting, the EdgeBank-∞ rule enters as the is_new_pair feature. Cong et al. (2023) extended this critique to broader temporal networks, showing that an MLP mixer matches attention-based methods on standard temporal link-prediction benchmarks. Recent studies likewise found that recency and popularity heuristics rival neural models (Cornell et al., 2025) and that negative-sampling artifacts can inflate temporal-graph link-prediction results (Daniluk and Dąbrowski, 2023). These works target link prediction. We study temporal edge-level anomaly detection, where labels identify atypical dynamic behaviour and compute efficiency directly affects detection latency.

Algorithm 1 Chronological simple-feature extraction.

IN Time-sorted edges $(u_k, v_k, t_k, b_k)_{k=1}^{n}$ and unseen-recency sentinel M.

STATE For key type $x \in \{p, s, d\}$ (pair, source, destination): lifetime count C_x, current-bucket count $C_x^{(b)}$, and last-seen time T_x. Initialize all counts to 0, all times to $\perp$, and $S, m \leftarrow 0$, $b_{\text{prev}} \leftarrow \perp$.

for $k = 1, \ldots, n$ **do**
  Set $q_p = (u_k, v_k)$, $q_s = u_k$, and $q_d = v_k$.
  if $b_k \neq b_{\text{prev}}$: reset every $C_x^{(b)}$ and set $b_{\text{prev}} \leftarrow b_k$.
  for each $x \in \{p, s, d\}$: $n_x \leftarrow [C_x[q_x] = 0]$ and $\Delta_x \leftarrow t_k - T_x[q_x]$ if seen, else M.
  Set $\bar{C} \leftarrow S/\max(1, m)$.
  emit $\phi_k = (n_p, n_s, n_d,\ C_p[q_p], C_s[q_s], C_d[q_d],$
  $C_p^{(b)}[q_p], C_s^{(b)}[q_s], C_d^{(b)}[q_d],\ \Delta_p, \Delta_s, \Delta_d,$
  $\log(1 + C_p^{(b)}[q_p]),\ C_p^{(b)}[q_p]/\max(1, \bar{C}))$.
  Update $S \leftarrow S + C_p^{(b)}[q_p]$ and $m \leftarrow m + 1$.
  for each $x \in \{p, s, d\}$: increment $C_x[q_x]$ and $C_x^{(b)}[q_x]$; set $T_x[q_x] \leftarrow t_k$.
end for

Corrective benchmark critiques in graph learning. A parallel line of work has repeatedly found that simple methods rival specialized models under careful evaluation. GADBench (Tang et al., 2023) showed that tree ensembles with simple neighbourhood aggregation outperform specialized GNN anomaly detectors on static graphs. Gu and Zou (2024) exposed benchmark limitations in node-level graph anomaly detection and found that message passing can reduce performance. Most relevant here, Heeg et al. (2025) showed that temporal GNNs can fail to distinguish certain temporal structures, complementing our finding that SLADE does not recover the planted multi-hop patterns in our synthetic probes.

Parsimony audits in machine learning. Lipton and Steinhardt (2019) warned that empirical gains attributed to new methods often stem from hyperparameter tuning rather than the methods themselves. We apply the same principle by pairing each accuracy gain with the compute spent to obtain it (Section 5, Figure 1).

3 Methodology

Our measurement protocol has four components: (1) a low-complexity one-feature reference, (2) chronological feature selection that prevents optimistic bias in that reference, (3) paired significance tests, and (4) synthetic probes that test whether the evaluated detector models identify known planted graph patterns.

3.1 One-feature reference pipeline

For every incoming edge (u, v, t), a single chronological pass maintains three types of state. First, it maintains lifetime counts for source u, destination v, and the ordered source–destination pair (u, v). Second, it maintains counts for the same source, destination, and pair within the current time bucket; these counts reset when the stream enters a new bucket. Third, it records the most recent timestamp at which each source, destination, and pair appeared. All features are read before the current edge updates this state, so the feature vector uses only prior edges.

For the five real interaction datasets and synthetic dataset, b_k assigns each timestamp to one of $B = 1000$ equal-width intervals spanning the observed timestamp range. In all cases, every feature for edge k is computed before edge k updates the maintained state, so the feature vector depends only on earlier edges.

When a source, destination, or pair appears for the first time, its recency is undefined because no previous timestamp exists. We encode this missing value as 10^9 to keep the feature vector finite and numerical. Separate binary first-occurrence indicators distinguish this case from a previously observed key with a long time gap. The value 10^9 is an implementation convention, not a model

parameter. Feature extraction processes the chronologically ordered edge stream once, computes all features before updating the state, and therefore runs in $O(|E|)$ time.

For each of the six datasets, we (i) compute the training-slice AUC of every candidate feature; (ii) select the feature with the highest effective training AUC, defined as $\max(\text{AUC}_{\text{train}}, 1 - \text{AUC}_{\text{train}})$, and (iii) report that feature's orientation-invariant test separability as $\max(\text{AUC}_{\text{test}}, 1 - \text{AUC}_{\text{test}})$. Effective AUC is a diagnostic measure of separability rather than a deployment-realizable score. The directed score instead applies the training-selected orientation without test-side maximization.

3.2 Paired Inference

Within each dataset, the methods being compared assign scores to the same held-out test edges and are evaluated against the same labels. Their ROC-AUC estimates are therefore statistically dependent rather than independent. We account for this dependence with paired DeLong correlated-ROC tests (DeLong et al., 1988) and a paired permutation test. Section 4 describes their application.

3.3 Synth-Triangle Generator

Synth-Triangle tests whether a detector model can recover a known structural pattern. The common-neighbor score verifies that the planted pattern is detectable; it is not part of the detector-model comparison.

We generate a temporal network with $n = 2000$ nodes and $m = 80{,}000$ edges. Edges arrive according to a homogeneous Poisson process, and nodes have equal expected interaction rates. After a warm-up period, we replace a fraction $p = 0.01$ of the stream with $pm = 800$ planted anomalies.

Each anomalous edge (a, c) closes an existing two-hop path $a - b - c$. We verify that this pattern is recoverable using the pre-event common-neighbor score

$$s_{\text{CN}}(a, c, t) = |N_{<t}(a) \cap N_{<t}(c)|\,,$$

where $N_{<t}(x)$ contains only neighbors observed before time t. The score uses no anomaly labels or future edges. The experiment therefore tests whether the evaluated detector models recover a planted structure that is known to be available in the graph history.

4 Experimental Setup

All methods are evaluated on the same edge streams, chronological split, and evaluation metric. We use "pp" to denote percentage points. Evaluation code and synthetic generators are available at `https://anonymous.4open.science/r/beat_the_counter_first-A000/README.md`.

4.1 Datasets

We evaluate on five public datasets and one generated probe (Table 1). Wikipedia, MOOC, and Reddit come from the JODIE releases Kumar et al. (2019); Bitcoin-Alpha and Bitcoin-OTC are signed trust networks Kumar et al. (2016); and Synth-Triangle is generated as defined in Section 3.3. The public datasets combine bipartite user–item interaction graphs with signed trust networks, span 24K to 672K edges, and range from 0.05% to 7.22% anomalous edges. Synth-Triangle and the additional Synth-Quad experiment provide controlled probes with planted structural patterns.

4.2 Statistical tests

We use the paired DeLong test for Wikipedia, Bitcoin-Alpha, Bitcoin-OTC, MOOC, and Synth-Triangle. Reddit requires separate treatment because the direction of the one-feature score reverses on the test slice. We therefore use a paired permutation test for Reddit because the reported statistic is the difference in orientation-invariant effective AUC; details are given in Appendix D. Both tests compare methods on identical test edges and labels. Their implementation is described in Appendix C.

Table 1: Datasets. Positive rate is the fraction of labelled anomalous edges in the full stream. *Selected feature* is chosen on the chronological training slice (Section 3.1). All six datasets use $B = 1000$ equal-width intervals. Their sizes range from 24K to 672K edges, and their anomaly rates range from 0.05% to 7.22%.

Dataset	$\lvert E\rvert$	Pos rate	Selected feature
Wikipedia	157 474	0.14 %	c_u (inverted)
Bitcoin-Alpha	24 186	3.61 %	c_v
Bitcoin-OTC	35 592	7.22 %	Δt_v (inverted)
MOOC	411 749	0.99 %	c_u (inverted)
Reddit	672 447	0.05 %	c_u
Synth-Triangle	80 000	1.00 %	c_v

The positive rates in our benchmarks span two orders of magnitude (0.05%-7.22%). ROC-AUC is the standard ranking metric for unsupervised anomaly detection under such imbalance, and our paired rank-based inference (Section 3.2) remains valid regardless of base rate because it operates on per-edge score orderings rather than threshold-dependent metrics.

4.3 Compared methods

We evaluate four methods on every dataset: SimpleCount, MIDAS-F, SLADE, and IsoForest. Random guessing provides a chance reference in the supplementary artefact. Beyond the one-feature reference, the evaluation emphasizes diversity of mechanisms rather than dense coverage of one model family: streaming sketch statistics with MIDAS-F, classical unsupervised feature-space isolation with IsoForest, and self-supervised temporal memory and attention with SLADE. GraphMixer provides a separate supervised model check.

- **SimpleCount**: one scalar feature selected from the fixed 14-feature count, recency, novelty, and count-derived pool on the chronological training slice (Section 3.1). The five real interaction datasets and Synth-Triangle use $B = 1000$ equal-width intervals over the observed timestamp range.
- **SLADE**: the self-supervised temporal-graph detector proposed by Lee et al. (2024). We use the authors' implementation and each dataset's recommended configuration. Each run is trained for 10 epochs on one NVIDIA H100 GPU; reported AUCs are means across seeds 0–4.
- **MIDAS-F**: We use a NumPy port of the C++ MIDAS-F reference implementation (Bhatia et al., 2022). We hold the repository demo configuration fixed across all six datasets: two hash rows and 1024 columns per Count–Min Sketch, a filtering threshold of 10^3, and a temporal decay factor of 0.5.
- **IsoForest**: an unsupervised nonlinear baseline (Liu et al., 2008) fitted to the complete 14-dimensional feature vector on the chronological training slice and evaluated on the test slice. We use 200 trees with automatic subsampling and contamination settings. This baseline tests whether jointly modelling all engineered features improves over selecting one scalar feature.

Separate supervised GraphMixer check. We additionally evaluate GraphMixer (Cong et al., 2023) on Wikipedia and Reddit using the DyGLib dynamic-node-classification pipeline: temporal-link pretraining followed by a downstream classifier trained directly on anomaly labels. While SimpleCount and GraphMixer both use training labels, they use it for different purposes. SimpleCount uses them only to select one feature and its orientation from a fixed pool; its label-dependent output is one feature index and one sign bit, not a fitted continuous decision function. GraphMixer instead uses anomaly labels to fit a supervised downstream classifier. Accordingly, SimpleCount appears in the main audit as a diagnostic reference, whereas GraphMixer is reported separately as a supervised model check. The check spans one SimpleCount-favourable dataset and one unfavourable dataset: under effective AUC, GraphMixer trails by 2.3 points on Wikipedia and leads by 2.1 points on Reddit (Appendix E).

Table 2: Label usage and fitted parameters. SIMPLECOUNT uses labels for discrete selection from a fixed feature pool, whereas GraphMixer uses labels to fit a supervised decision function.

Method	Labels used for	Fitted parameters
SIMPLECOUNT	Choosing 1 of 14 features on the training slice	2 (index, sign bit)
GraphMixer	Fitting a supervised node classifier	$\sim$1.2M (estimated)
SLADE	Not used (self-supervised)	$\sim 8.5 \times 10^5$ (counted, default configuration)
MIDAS-F	Not used	4 hyperparameters plus sketch counters (streaming state)
ISOFOREST	Not used	$\sim 10^4$ split rules

Table 3: Effective test ROC-AUC under the chronological 85/15 protocol. Bold indicates the per-dataset best. SLADE is run at the repository's recommended configuration for each dataset; its values are five-seed means. For SIMPLECOUNT, effective and directed AUC agree on five datasets; on Reddit, the directed AUC is 0.382 rather than 0.618 (Appendix A1).

Dataset	MIDAS-F	ISOFOREST	SIMPLECOUNT	SLADE
Wikipedia	0.535	0.589	**0.887**	0.886
Bitcoin-Alpha	0.641	0.635	0.664	**0.766**
Bitcoin-OTC	0.655	0.602	0.691	**0.772**
MOOC	0.661	0.587	**0.727**	0.602
Reddit	0.514	0.551	0.618	**0.755**
Synth-Triangle	0.507	0.529	**0.568**	0.522

EDGEBANK equivalence. The EDGEBANK-∞ memorization baseline Poursafaei et al. (2022) corresponds to our is_new_pair feature up to orientation: EDGEBANK scores previously seen pairs high for link prediction, whereas is_new_pair scores novel pairs high for anomaly detection. Its per-dataset AUC appears in the per-feature breakdown (Figure A3).

Additional evaluation details are given in Appendix C.

5 RESULTS

5.1 PER-DATASET AUC ACROSS ALL METHODS

Table 3 reports the effective test ROC-AUC of the four methods on every dataset across our six-benchmark suite.

Across these six benchmarks, SIMPLECOUNT matches or exceeds SLADE on Wikipedia, MOOC, and Synth-Triangle (3/6 datasets), while SLADE wins on Bitcoin-Alpha, Bitcoin-OTC, and Reddit. With the training-selected orientation applied to test edges, SIMPLECOUNT's directed Reddit AUC is 0.382 rather than its effective AUC of 0.618. Reddit is already a SLADE win, so the 3/6 match-or-exceed result and per-dataset win/loss pattern are unchanged. SIMPLECOUNT also exceeds ISOFOREST on all six datasets, showing that a nonlinear combination of the complete 14-feature vector does not improve on the selected scalar feature in this suite. Neither MIDAS-F nor ISOFOREST is the top-performing method on any dataset. The Wikipedia result is effectively a tie (0.887 versus SLADE's 0.886), and every comparison uses SLADE at its documented high-performance setting. For the SIMPLECOUNT-versus-SLADE inference, five datasets use paired DeLong tests; Reddit uses the paired permutation test. Wikipedia and Synth-Triangle are not significant; the four significant comparisons survive a Holm–Bonferroni correction. MIDAS-F is not significantly different from SIMPLECOUNT on Bitcoin-Alpha. Figure A1 reports the paired effect sizes and confidence intervals for the five datasets evaluated with DeLong.

5.2 COMPLEXITY-BENEFIT: ACCURACY GAIN VERSUS COMPUTE PREMIUM

Figure 1 reports the central complexity-benefit finding. SLADE uses 23–133$\times$ the wall time of SIMPLECOUNT, with a mean of 72$\times$ across all six datasets. On the three datasets where SLADE leads, the mean cost is 91$\times$ for a mean AUC gain of 10.66 pp. These totals measure the practitioner

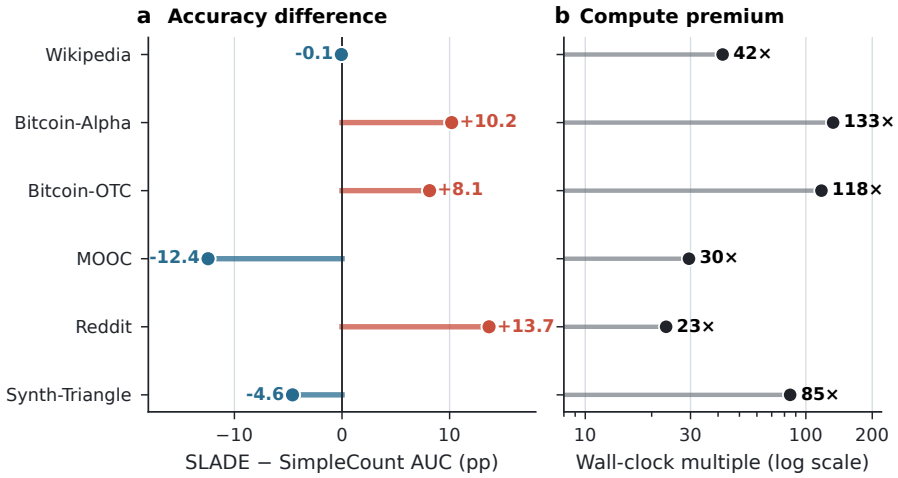


Figure 1: Complexity-benefit per dataset. Left: AUC gap (SLADE − SimpleCount, percentage points; positive values indicate SLADE is ahead). Right: wall-clock cost multiple (SLADE / SimpleCount, log scale; mean across six datasets is 72×). Wall-clock cost is end-to-end and includes SLADE training; SLADE runs on an H100 GPU and SimpleCount on one CPU core.

wall-clock cost of running the released implementations in their default configurations and therefore include SLADE's training. SLADE is timed on an NVIDIA H100 GPU, while SimpleCount runs on one CPU core. The comparison therefore gives SLADE its favourable hardware rather than forcing both methods onto the CPU.

5.3 Multi-seed variance

Across five seeds, SLADE's sample standard deviation ranges from 0.002 to 0.012 (Figure A2). Winner direction is stable on five datasets; on near-tied Wikipedia, the numerical ordering between SLADE and SimpleCount changes across seeds.

5.4 Recoverable local structure that tested methods miss

On Synth-Triangle, an elementary two-hop neighbourhood score separates the anomalies, but every method we tested (MIDAS-F, SimpleCount, SLADE, IsoForest) operates near chance (0.50–0.57; Figure 2). SimpleCount reaches 0.568 from a residual $\deg \geq 1$ generator bias. Because the planting mechanism is known, we define a pre-event common-neighbor score

$$\text{score}(u, v, t) = |N(u, t) \cap N(v, t)|,$$

where $N(x, t)$ contains interactions strictly before time t. The score uses no labels or future edges; it is generator-informed because its functional form is chosen with knowledge of the planted closure pattern. It attains AUC $= 0.666$ (mean common-neighbor count 3.66 on positives versus 2.64 on negatives). Thus, the planted two-hop structure is recoverable from prior edges. The $+14$ pp gap shows that SLADE does not extract this local multi-hop structure.

A second closure pattern: 4-cycles. To test whether the generator-informed score result is specific to triangle closure, we repeat the construction with 4-cycle-closing anomalies (*Synth-Quad*: 8000 nodes, 40 000 edges, 2% positives; each anomaly $a \to d$ closes a sampled length-3 path $a-b-c-d$). Every tested feature-based and neural model is again near chance (SimpleCount 0.557, MIDAS-F 0.521, IsoForest 0.528, SLADE 0.506). Crucially, the 2-hop common-neighbor score that separates Synth-Triangle is now *also* near chance (0.508): a 4-cycle leaves no 2-hop signature. A generator-informed pre-event length-3 path-count score,

$$\text{score}(u, v, t) = \sum_{b \in N(u,t)} |N(b, t) \cap N(v, t)|,$$

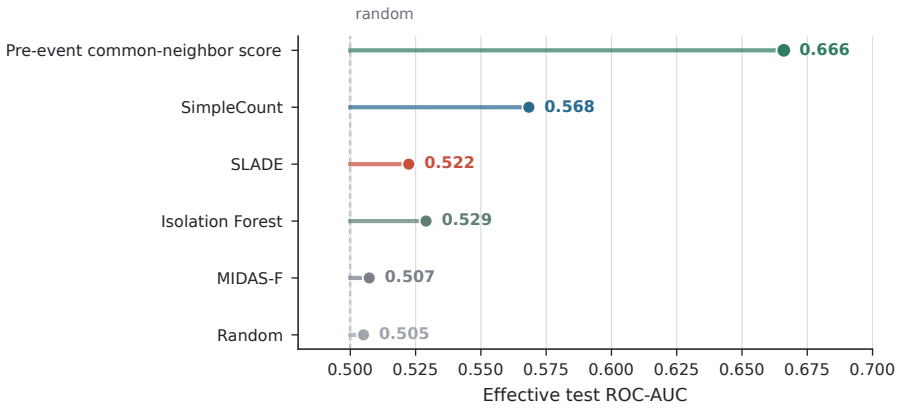


Figure 2: Synth-Triangle: feature-based and neural models (Random, MIDAS-F, IsoForest, SLADE, SimpleCount) all operate at or near chance (0.50–0.57). A generator-informed pre-event common-neighbor score attains AUC $= 0.666$ (top row), confirming that the planted closure structure is recoverable even when SLADE does not recover it. The reference line indicates random chance.

recovers the anomalies at AUC $= 0.955$, a paired-DeLong margin of $+0.45$ over SLADE ($z = 23.6$, $p < 10^{-4}$) and $+0.40$ over SimpleCount ($z = 20.9$). Like the common-neighbor score, this score uses only prior edges and no labels; its form is chosen from knowledge of the generator. Thus, each planted closure pattern is recoverable from prior edges, while SLADE recovers neither. The common-neighbor score that separates triangles does not separate 4-cycles.

5.5 Per-feature ablation and operational metrics

The per-feature AUC heatmap (Appendix Figure A3) shows that source count c_u is selected on Wikipedia, MOOC, and Reddit, while destination count c_v is selected on Bitcoin-Alpha and Synth-Triangle. Wikipedia and MOOC use the inverted count direction; Reddit selects source count c_u. In implementation-level inference-only throughput measurements, SimpleCount processes approximately 14 000–50 000 edges/sec on one CPU core, while SLADE processes approximately 670–1 500 edges/sec on an H100 GPU. SimpleCount is therefore approximately 9–75$\times$ faster at inference alone.

6 Discussion

6.1 When is the complexity premium justified?

The value of added complexity varies by dataset. On Bitcoin-Alpha and Bitcoin-OTC, SLADE gains 8–10 pp AUC and the graphs contain only 24K–35K edges. On Wikipedia, MOOC, and Synth-Triangle, the added compute does not improve AUC. Reddit presents the strongest trade-off: SLADE gains 13.7 pp at a 23$\times$ wall-clock premium. These results show where the additional compute buys accuracy and where it does not.

Dataset structure. Figure 3 shows a split at destination-activity Gini 0.55: SLADE leads on the three higher-Gini datasets, while SimpleCount matches or exceeds it on the three lower-Gini datasets. A plausible explanation is that concentrated destination activity creates hub-related interaction patterns available to attention mechanisms but absent from a one-dimensional feature. Appendix B gives the formal definition.

6.2 Why does MOOC favour SimpleCount

The MOOC dropout signal is concentrated in low-activity users. The selected score is the inverted lifetime source interaction count, $-c_u$, which assigns higher anomaly scores to interactions from

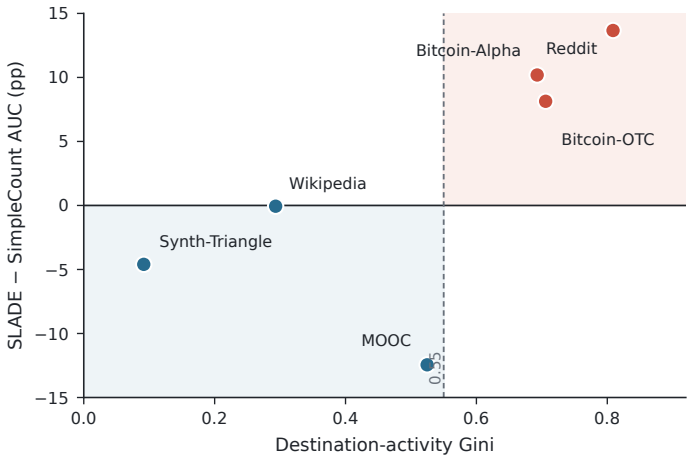


Figure 3: AUC gap (SLADE − SimpleCount, percentage points) against destination-activity Gini. Positive values indicate that SLADE leads; negative values indicate that SimpleCount leads.

users with fewer prior interactions. Users who leave after very few sessions are labelled as dropouts, so this feature captures the label pattern directly. SLADE's contrastive objective encourages node representations to drift minimally over time and may suppress these activity differences. This interpretation is consistent with the per-feature ablation (Appendix Figure A3).

7 Conclusion

We asked whether the added complexity of a temporal-graph anomaly detector model is justified across five public datasets and one generated dataset. We test whether a nonlinear model of all 14 features improves over selecting one feature, how much accuracy MIDAS-F and SLADE gain over a reference with no parameter fitting, and how much compute they spend for that gain. The reference exceeds IsoForest on all six datasets, matches or exceeds SLADE on three, and is within 13.7 percentage points of SLADE on all six. SLADE requires 23 to 133 times the reference wall time, with a mean of $72\times$ across all six datasets. Across the three SLADE wins, the means are $91\times$ and $+10.66$ pp AUC. On the synthetic datasets, every method remains near chance, while pre-event structural scores recover planted patterns at AUC up to 0.955.

Complexity earns its cost on the small Bitcoin graphs, where SLADE recovers signal absent from the selected feature. Across all six datasets, its mean wall-clock cost is $72\times$ that of SimpleCount. The benefit of complexity is therefore dataset-dependent. What SimpleCount offers is not a replacement model for deployment, but rather, we recommend that every claimed gain on a deployed model be reported against a strong one-feature reference like SimpleCount, together with its compute cost.

AI use statement

In this work, we used generative AI tools to assist with language editing, code development, and manuscript organization. It was not used for idea generation and automatic research. Any GenAI output was verified by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

The anonymized github repository for this work is available at `https://anonymous.4open.science/r/beat_the_counter_first-A000/README.md`

References

Siddharth Bhatia, Rui Liu, Bryan Hooi, Minji Yoon, Kijung Shin, and Christos Faloutsos. Real-time anomaly detection in edge streams. *ACM Transactions on Knowledge Discovery from Data*, 16 (4):75:1–75:22, 2022. doi: 10.1145/3494564.

Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations (ICLR)*, 2020.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. arXiv preprint arXiv:2006.10637, 2020. Presented at the ICML 2020 Workshop on Graph Representation Learning.

Jongha Lee, Sunwoo Kim, and Kijung Shin. Slade: Detecting dynamic anomalies in edge streams without labels via self-supervised learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1506–1517, 2024. doi: 10.1145/3637528.3671845.

Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022. URL https://papers.nips.cc/paper_files/paper/2022/hash/d49042a5d49818711c401d34172f9900-Abstract-Datasets_and_Benchmarks.html.

Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? In *International Conference on Learning Representations (ICLR)*, 2023.

Jianheng Tang, Fengrui Hua, Ziqi Gao, Peilin Zhao, and Jia Li. GADBench: Revisiting and benchmarking supervised graph anomaly detection. In *Advances in Neural Information Processing Systems (NeurIPS Datasets and Benchmarks)*, volume 36, 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/5eaafd67434a4cfb1cf829722c65f184-Abstract-Datasets_and_Benchmarks.html.

Franziska Heeg, Jonas Sauer, Petra Mutzel, and Ingo Scholtes. Weisfeiler and Leman follow the arrow of time: Expressive power of message passing in temporal event graphs. *arXiv preprint arXiv:2505.24438*, 2025.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NIPS)*, volume 30, 2017.

Siddharth Bhatia, Bryan Hooi, Minji Yoon, Kijung Shin, and Christos Faloutsos. Midas: microcluster-based detector of anomalies in edge streams. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3242–3249, 2020. doi: 10.1609/aaai.v34i04.5724.

Dhivya Eswaran and Christos Faloutsos. Sedanspot: Detecting anomalies in edge streams. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 953–958. IEEE, 2018. doi: 10.1109/ICDM.2018.00117.

Yen-Yu Chang, Pan Li, Rok Sosic, M. H. Afifi, Marco Schweighauser, and Jure Leskovec. F-fade: Frequency factorization for anomaly detection in edge streams. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 589–597, 2021. doi: 10.1145/3437963.3441806.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1269–1278, 2019. doi: 10.1145/3292500.3330895.

Srijan Kumar, Francesca Spezzano, V. S. Subrahmanian, and Christos Faloutsos. Edge weight prediction in weighted signed networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 221–230, 2016. doi: 10.1109/ICDM.2016.0033.

Filip Cornell, Oleg Smirnov, Gabriela Zarzar Gandler, and Lele Cao. On the power of heuristics in temporal graphs. In *ICLR Workshop "I Can't Believe It's Not Better" (ICBINB), PMLR*, volume 296, pages 37–46, 2025.

Michał Daniluk and Jacek Dąbrowski. Temporal graph models fail to capture global temporal dynamics. *arXiv preprint arXiv:2309.15730*, 2023. doi: 10.48550/arXiv.2309.15730.

Jing Gu and Dongmian Zou. Three revisits to node-level graph anomaly detection: Outliers, message passing and hyperbolic neural networks. In *Proceedings of the Second Learning on Graphs Conference*, volume 231 of *Proceedings of Machine Learning Research*, pages 14:1–14:29. PMLR, 2024. URL https://proceedings.mlr.press/v231/gu24a.html.

Zachary C Lipton and Jacob Steinhardt. Troubling trends in machine learning scholarship. *Queue*, 17(1):45–77, 2019. doi: 10.1145/3317287.3328534.

Elizabeth R DeLong, David M DeLong, and Daniel L Clarke-Pearson. Comparing the areas under two or more correlated receiver operating characteristic curves: a nonparametric approach. *Biometrics*, 44(3):837–845, 1988. doi: 10.2307/2531595.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008. doi: 10.1109/ICDM.2008.17.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

A Recommendations for benchmark authors

Our audit suggests five additions to future temporal-graph anomaly-detection evaluations.

1. **Report a chronologically selected one-feature baseline.** A single feature selected from a fixed pool of counts, recencies, novelty indicators, and count-derived transforms computed from prior interactions (Section 3.1) is a compact reference. It matched or beat SLADE on three of our six datasets. We recommend reporting it alongside learned models.
2. **Report paired significance in addition to seed CIs.** The paired DeLong test DeLong et al. (1988), or a paired permutation test when score direction is part of the reported statistic, turns "method A scores higher" into "method A is significantly better on this dataset" (Section 3.2).
3. **Publish wall-clock cost alongside AUC.** A method that wins by $+10.66$ pp at $91\times$ the compute is a different proposition from one that wins at parity. Per-percentage-point cost (Figure 1) makes the trade-off explicit.
4. **Include a nonlinear full-feature control.** This distinguishes gains from combining engineered features from gains due to a temporal-graph architecture.
5. **Validate architectural claims on controlled synthetic probes with generator-informed pre-event structural scores.** Our Synth-Triangle probe with a common-neighbor score (Section 3.3) shows that "our model exploits k-hop structure" is a testable claim, not an assumption: the planted structure was recoverable from prior edges (AUC $= 0.666$) yet was not recovered by SLADE.

B Destination-activity Gini

For each node v that appears at least once as a destination, let

$$c_v = \sum_{(u,w,t)\in E} \mathbf{1}[w = v]$$

Table A1: Directed test ROC-AUC using the orientation selected on the chronological training slice, with no test-side maximization. The SLADE scores are inherently directed and therefore equal those in Table 3. Only SimpleCount on Reddit changes relative to its effective AUC.

Dataset	SimpleCount (directed)	SLADE
Wikipedia	0.887	0.886
Bitcoin-Alpha	0.664	0.766
Bitcoin-OTC	0.691	0.772
MOOC	0.727	0.602
Reddit	**0.382**	0.755
Synth-Triangle	0.568	0.522

be its number of destination-side interaction events. Repeated interactions are counted separately, and nodes that never appear as destinations are excluded. For the n observed destination nodes, we define

$$G_{\mathrm{dst}} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |c_i - c_j|}{2n \sum_{i=1}^{n} c_i}.$$

A value of 0 indicates equal destination activity, while larger values indicate greater concentration.

C Evaluation protocol

Split. A chronological 85/15 train/test split is applied to each dataset. The simple-feature pipeline receives no parameter tuning beyond feature selection on the chronological training slice (Section 3.1).

Metric. Test-slice effective ROC-AUC is reported for all four methods in Table 3. ROC-AUC is the standard metric in temporal-graph anomaly detection; Appendix F gives a complementary precision-recall analysis. Effective AUC, $\max(\mathrm{AUC}_{\mathrm{test}}, 1 - \mathrm{AUC}_{\mathrm{test}})$, measures orientation-invariant ranking separability rather than a prospectively oriented deployment score.

Paired testing. DeLong correlated-ROC tests DeLong et al. (1988) use per-edge scores from both methods on the same test slice, as described in Section 3.2. For Reddit, we use a paired, orientation-invariant permutation test. We independently convert both score vectors to normalized midranks, randomly swap the paired method labels, and recompute the effective-AUC gap for each of 9,999 permutations. The SimpleCount intervals in Figure A2 use 500 bootstrap iterations, with per-iteration subsampling to $\leq 100\,000$ edges for computational tractability on the largest datasets (Reddit: 672 447 edges).

Multiple-testing correction. We apply Holm–Bonferroni correction (Holm, 1979) with a family-wise $\alpha = 0.05$ across the six SLADE-vs-SimpleCount comparisons (one per dataset).

D Reddit paired permutation test

We compare SimpleCount and SLADE with a paired Monte Carlo permutation test that recomputes the orientation-invariant effective-AUC gap. Scores are converted to normalized within-method midranks. In each of $B = 9{,}999$ replicates, the two ranked scores are independently exchanged within each test edge with probability $1/2$, using random seed 20260730. With the standard add-one correction, the two-sided p-value is

$$p = \frac{E + 1}{B + 1},$$

where E is the number of sampled gaps at least as large in absolute value as the observed gap. For Reddit, $E = 26$, giving $p = (26 + 1)/(9{,}999 + 1) = 0.0027$.

E GraphMixer comparison

We evaluate GraphMixer using the DyGLib dynamic-node-classification pipeline: temporal-link pretraining followed by a node classifier trained on anomaly labels. The comparison spans one SimpleCount-favourable dataset and one unfavourable dataset. Under effective AUC, GraphMixer trails SimpleCount by 2.3 percentage points on Wikipedia and leads by 2.1 points on Reddit. For the deployment-realizable directed comparison, Wikipedia is unchanged, while GraphMixer scores 0.639 against SimpleCount's 0.382 on Reddit, a lead of 25.7 points (Table A2).

Table A2: GraphMixer comparison. SimpleCount effective AUC is included for consistency with the headline results; its directed AUC applies the training-selected orientation without test-side maximization. GraphMixer's learned orientation is inherently directed. GraphMixer values are mean ± sample standard deviation over five seeds, and the gap uses directed AUC.

Dataset	SimpleCount eff.	SimpleCount dir.	GraphMixer dir.	Dir. gap
Wikipedia	0.887	0.887	0.864 ± 0.014	-2.3 pp
Reddit	0.618	0.382	0.639 ± 0.012	$+25.7$ pp

F Precision-recall robustness

ROC-AUC can be optimistic under extreme class imbalance, so we report average precision (AUPRC) on the same chronological test slices (Table A3). MIDAS-F, IsoForest, and SimpleCount use the same deterministic scores as the ROC-AUC comparison; SLADE values are five-seed means and the chance level equals the test-slice base rate. The three-of-six count from Table 3 is unchanged under AUPRC, but membership changes: SLADE leads Wikipedia, Bitcoin-Alpha, and Reddit, while SimpleCount leads Bitcoin-OTC, MOOC, and Synth-Triangle. The two highest test base rates are Bitcoin-OTC (0.088) and Bitcoin-Alpha (0.064). Bitcoin-OTC, a SLADE win on ROC-AUC, reverses in SimpleCount's favour under AUPRC. Wikipedia and Reddit sit near the base-rate floor, where every method's AUPRC is below 0.02 and absolute differences are small. SimpleCount exceeds IsoForest on five datasets; on Reddit, IsoForest scores 0.00124 versus 0.00118 for SimpleCount, an absolute difference below 0.0001.

Table A3: Average precision (AUPRC) on the chronological test slice. Bold indicates the per-dataset best across the four methods; SLADE is the five-seed mean and chance equals the test-slice base rate. The three-of-six pattern of Table 3 is preserved.

Dataset	Chance	MIDAS-F	IsoForest	SimpleCount	SLADE
Wikipedia	0.002	0.002	0.004	0.015	**0.016**
Bitcoin-Alpha	0.064	0.102	0.097	0.111	**0.148**
Bitcoin-OTC	0.088	0.146	0.128	**0.289**	0.205
MOOC	0.009	0.019	0.013	**0.023**	0.016
Reddit	0.001	0.001	0.001	0.001	**0.003**
Synth-Triangle	0.021	0.021	0.021	**0.026**	0.023

G Supplementary figures

The following figures support the analyses in the main text.

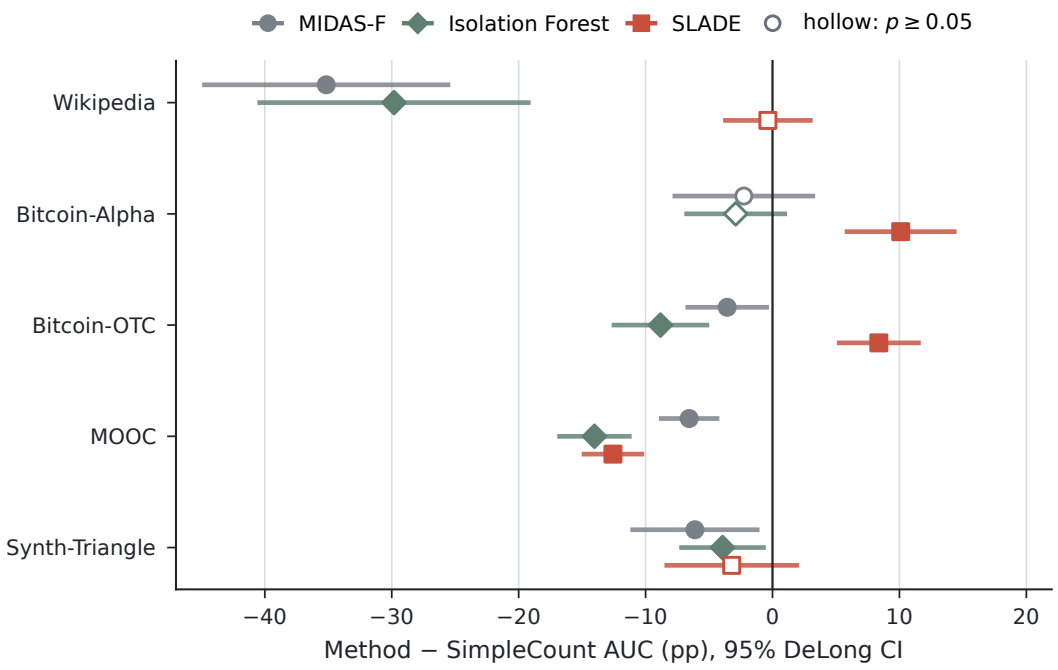


Figure A1: Paired DeLong forest for MIDAS-F, IsoForest, and SLADE on five datasets. Reddit uses the paired permutation test and is omitted. Intervals show effect size (method AUC − SimpleCount AUC) with 95 % CIs; hollow markers denote unadjusted $p \geq 0.05$.

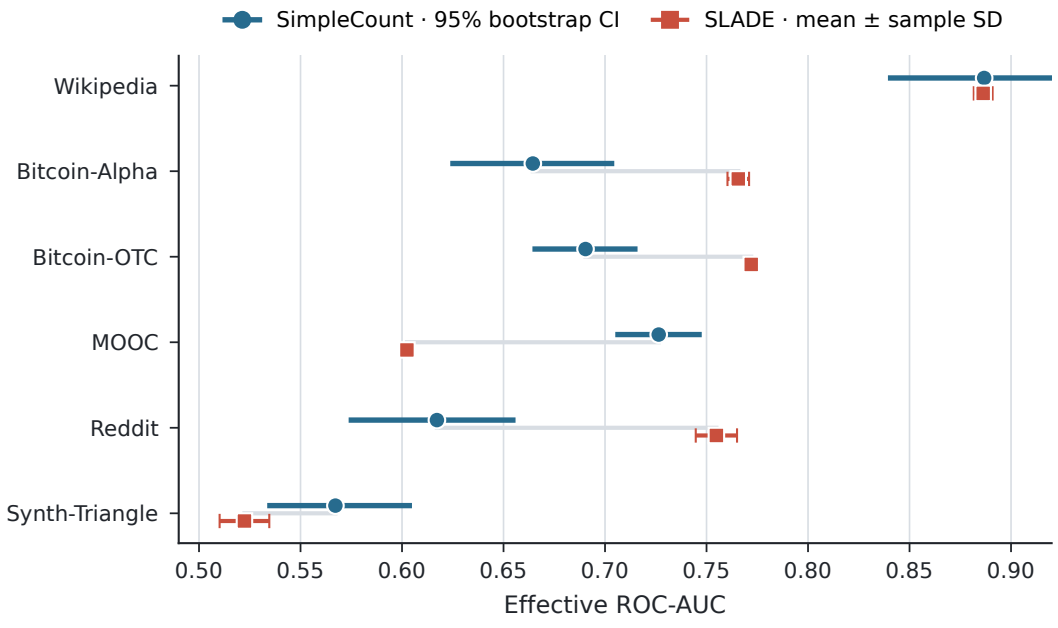


Figure A2: Uncertainty and seed variation. Circles show SimpleCount with 95 % bootstrap CIs; squares show SLADE's five-seed mean ± sample standard deviation. Gray connectors emphasize the per-dataset gap. SLADE sample standard deviation ranges from 0.002 to 0.012.

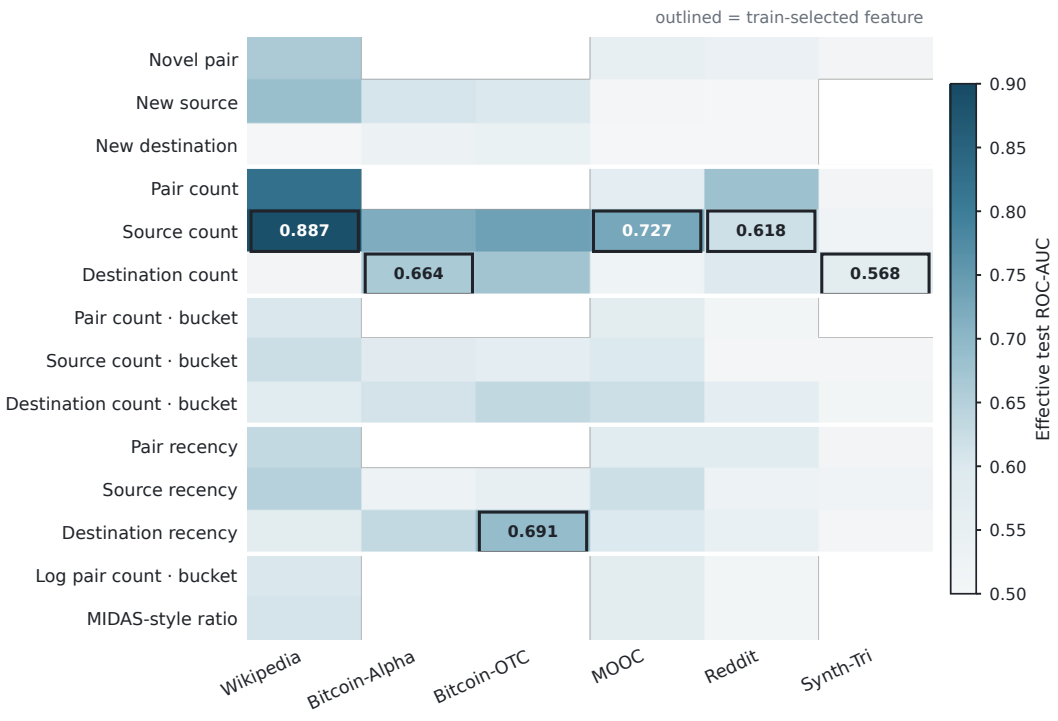


Figure A3: Effective test AUC for each chronological candidate feature. Outlined cells identify the feature selected on the chronological training slice; their values are printed. Blank cells indicate that the feature is constant on that test slice, so ROC-AUC is undefined. The selected feature varies by dataset and is a per-source or per-destination count or recency feature.