Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 23, No. 2, 2026

Frozen-Reference Graph Context for Inductive Network-Flow Anomaly Detection

Violet Ghosh, Samuel Green*

Department of Computer Science, Rutgers University–New Brunswick, USA

Corresponding author

Abstract: We study inductive network-flow anomaly detection using a frozen training-only graph reference rather than edges among future observations. Category-conditioned random-projection retrieval yields 12 neighbors per query; cosine-weighted context and elementwise disagreement are inherited from the DAS-GNN disagreement-aware graph framework. We compare all-record and label-vetted benign reference banks, prototype shrinkage, and compact uniform-mean residual features. A fully executable study uses two independently generated realizations of 48,000 synthetic flows each, three neural seeds, validation-selected thresholds, LightGBM, and saved per-record predictions. On the first synthetic test set, uniform graph averaging obtains 0.9052 ROC-AUC, 0.8243 average precision and 0.7330 F1, versus 0.8737, 0.7683 and 0.6787 for a matched MLP. On the independent generator realization, its average precision is 0.8153 versus 0.7645 for the MLP. More elaborate disagreement and prototype variants do not consistently improve on simple averaging. The result is an auditable, strict-inductive method comparison rather than a claim of optimality or accuracy on recorded production traffic.

Keywords: Graph anomaly detection; Inductive graph learning; Network flows; Reference memory; Disagreement; Streaming inference; Synthetic benchmark.

1. Introduction

Network-flow anomaly detection has traditionally been formulated as classification or novelty estimation from one record at a time. That abstraction omits whether a flow is compatible with peers that share protocol, service, and connection state. Reviews of deep outlier detection [1] and graph anomalies [2], [3] emphasize that attribute deviation and relational inconsistency need not coincide. Public intrusion corpora such as UNSW-NB15 [4] provide a useful motivation, although constructing a flow graph is a separate modeling choice: the benchmark table is not a native host-communication graph. Cost-sensitive adversarial training for cloud intrusion detection demonstrates the value of examining detection performance under strategically challenging traffic conditions, complementing conventional anomaly-ranking metrics [28].

Graph convolution [5], inductive neighborhood learning [6], and attention [7] offer general ways to mix relational evidence. For a production stream, however, the training regime must not depend on unobserved future traffic. Moreover, repeated edge expansion can be expensive when labels and detection thresholds change more frequently than the reference behavior. Precomputed propagation [8] and multiview feature caching [9] suggest that a neighborhood representation can be separated from the optimization loop. Temporal logical attention for log-based anomaly detection illustrates how structured operational context can strengthen fault interpretation in distributed systems, a useful methodological connection for relational flow analysis [29].

This study asks two related questions. Can disagreement-aware graph features be constructed with only historical, training-side references? And does restricting that reference memory to known benign traffic help preserve a recognizable normal context? We develop Reference-Anchored Inductive Disagreement (RAID), a deliberately small detector with a frozen lookup index, local residuals, categorical prototypes, and query-distance-dependent context shrinkage. To make the contribution independently checkable, all numerical evidence is generated from executable synthetic traffic rather than borrowed benchmark scores.

The contributions are (i) a strict-inductive implementation with bounded candidate retrieval and reproducible batch-invariance checks; (ii) two reference-memory regimes and explicitly tested disagreement, prototype, and shrinkage features; and (iii) auditable three-seed experiments on two independent realizations of a declared synthetic generator, with a matched LightGBM comparator and paired test-sample intervals. These experiments assess the value and fragility of inductive relational context without asserting performance on captured production traffic.

2. Reference-anchored problem setup

Let the labeled training records be $T=\{(x_i,c_i,y_i)\}$, with numeric $x_i \in R^{24}$, category triple c_i=(protocol,service,state), and $y_i \in \{0,1\}$. The positive class denotes a simulated attack. A deployment query q has x and c but never a known target. The reference bank R is fixed before any validation or test prediction; q is allowed to read R and its stored per-category statistics but cannot add itself or another held-out observation to the index. This separation makes inference inductive even when many arrivals are scored in one call.

The implementation directly adopts the central, executable graph computations of DAS-GNN [10]: category-conditioned multi-view Gaussian projection ordering, shortlist distance selection, cosine-gated neighborhood aggregation, and the explicit absolute node–context disagreement. These are the components that make DAS-GNN's neighborhood construction scalable, since sorted projections bound the candidate set, and disagreement-aware, since the node–context residual exposes contextual anomalies; they are kept unchanged in every graph variant studied here. DAS-GNN evaluates them on a full-feature batch graph, and the present

study asks whether they remain effective when every query is attached to a frozen training-only memory. Accordingly, the projection window is enlarged from ±3 to ±6, the acquisition-order shortcut is omitted, and an optional bank contains only training-labeled benign flows. Each difference is an explicit implementation switch, so its effect can be isolated.

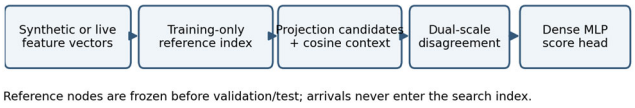


Fig. 1 Frozen-reference feature construction and dense scoring; no held-out record enters the reference memory.

3. Inductive raid method

3.1. Training-fitted transforms and reference purity

Only training values determine the 5th–95th percentile robust scaling of the 24 numeric coordinates; transformed values are clipped to [−8,8]. Category triples map to 72 known signatures (4 protocols × 6 services × 3 states); the exact group is included as the same 72-dimensional one-hot feature for every detector, preventing unequal exposure to category information. Baseline reference search retains all 28,000 training rows. Clean-reference search retains only y=0 training rows (22,517 in this run). Labels are used for that reference selection and for supervised model fitting, never to rank neighbors or create validation/test features.

3.2. Fixed-degree candidate construction

The 12 coordinates of largest training variance form the search subspace. Four Gaussian vectors are drawn once from seed 41. Within each category, training reference nodes are sorted by scalar projection. For each arriving query and projection, a binary search locates the insertion point; offsets −6 through +5 provide at most 48 preliminary positions. Duplicate candidates and the query itself (when computing training features) are discarded. Twelve nearest remaining candidates under squared Euclidean distance in the selected subspace become the directed query-to-reference neighborhood N(q). No test–test edges or label-conditioned candidate ranking are produced.

$$N(q) = \mathrm{TopK}_r\{\|xq - xr\|^2_S : r \in Q(q),\ c_r = cq\},\quad K=12. \quad (1)$$

Algorithm 1. frozen-reference scoring

Stage	Implemented operation
Fit	Training-only scaling, signatures, anchors
Index	Twelve high-variance axes; four projections
Query	Per-category search of ±6 projection positions
Select	Remove duplicates/self; retain 12 nearest
Cache	Gated context, residual, prototype, trust
Score	Dense classifier; validation-calibrated threshold

The reference index is built once, with sorting cost $O(Rn_r \log n_r)$. A new record requires R binary searches, at most 48 candidate comparisons in a 12-dimensional subspace, and O(Kd) context computations. For fixed R, K, d and window width, this is $O(R \log n_r + 48d_s + Kd)$ time per query and $O(n_r(d+R)+BK)$ working storage for a query batch of size B. This statement describes the implemented static index, not an incrementally maintained stream graph.

3.3. Local context and disagreement

Cosine similarities between q and its 12 reference neighbors are converted to temperature-0.5 softmax weights. The weighted mean g(q) is a local normal-behavior estimate only to the extent that its reference nodes are representative. The implementation also computes the elementwise absolute residual |x−g|, a simple uniform neighborhood mean, normalized gate entropy, cosine mean and standard deviation, and mean and standard deviation of candidate distance.

$$a(q,r) = \mathrm{softmax}_r(\cos(xq,xr)/0.5),\ \ g(q)=\Sigma_r a(q,r)xr. \quad (2)$$

An additional coordinatewise normalized local disagreement divides |x−g| by the standard deviation of the selected neighbor features plus 0.2. It is intended to distinguish unusually large residuals from dimensions that naturally vary within a neighborhood. Local disagreement may still be uninformative when a query closely resembles contaminated or shifted anchors; thus source labels used at training time become relevant through bank selection.

3.4. Prototype shrinkage and dual scales

For each category signature c, let μc and σc denote the coordinatewise mean and clipped standard deviation (minimum 0.3) of its reference examples. These are precomputed without future observations. A mean query-to-neighbor distance Δ(q) defines a bounded trust scalar τ(q)=clip(exp[−Δ(q)/2.5],0.02,0.98). The blended context is h(q)=τ(q)g(q)+(1−τ(q))μc. Intuitively, an isolated query trusts an individual shortlist less and a stable category prior more; the coefficient is a fixed heuristic, not a learned probability or causal trust estimate.

$$h(q) = \tau(q)g(q) + [1 - \tau(q)]\mu c. \quad (3)$$

RAID concatenates x, the category one-hot vector, h, |x−h|, |x−g|, |x−g|/(sd(N)+0.2), |x−μc|/σc, the five neighborhood statistics, and τ into a dense vector. A two-layer feed-forward network (96 and 48 hidden units, ReLU, dropout 0.1) produces the attack logit. Neighbor operations are entirely outside neural optimization; the network never traverses an adjacency list during an epoch. This particular combination is intentionally compared with uniform context, the original local absolute disagreement, no prototype, no trust, and no local residual to avoid crediting unused components.

3.5. Supervised training and calibration

All neural variants share the identical training records, one-hot categories, hidden widths, early-stopping criterion, and AdamW optimizer (learning rate 0.002, weight decay 10^{-4}, batch size 2,048). Positive-class weighted binary cross entropy uses the training-only ratio n_0/n_1. At most 12 epochs are run, and best validation average precision with patience three selects the checkpoint. For each run, an F1-maximizing threshold is chosen from the validation precision–recall curve and frozen for test scoring. No test label enters reference construction, stopping decisions or threshold selection. An exploratory second-phase feature search followed the initial test analysis; its validation-based choice and independent generator replication are described separately to distinguish exploratory from confirmatory evidence.

3.6. Reference lifecycle and corner cases

Two operational choices deserve separate treatment. First, the benign-only anchor filter uses labels that may be delayed or disputed in a live monitoring service. A realistic

deployment would obtain vetted reference labels from an earlier incident-response period, freeze a reference snapshot with a version identifier, and score new records against that snapshot until an authorized refresh. The present test implements only the frozen snapshot. It does not claim to evaluate acceptance rules, delayed-label feedback, or cache refresh under concept drift. Adaptive reinforcement learning for cybersecurity incident response provides a practical downstream perspective on how anomaly evidence can inform risk-sensitive response decisions [30].

Second, fixed-degree lookup requires enough eligible anchors per category. With small or previously unseen signature buckets, the deterministic candidate pool can contain fewer than K distinct neighbors. The executed synthetic generator supplies many examples of all 72 signatures, so the failure condition does not occur. On a real stream it should trigger an explicit fallback (e.g., hierarchical protocol-level references or abstention), not silently fill a neighborhood with unrelated flows. A trust coefficient estimated from distance cannot repair a missing or mislabeled signature on its own.

The local prototype is most plausible for unimodal category-conditioned normal behavior. If a service has several legitimate operating regimes, shrinkage toward a single μc can erase useful within-group structure. The no-prototype and no-trust experiments quantify this risk only for the chosen simulator, and the no-local-residual variant tests feature redundancy. More expressive density-aware prototype banks could be investigated later, but no learned clustering or online prototype update is part of this implementation.

4. Relation to existing work

Sampling-oriented graph scalability takes several forms. FastGCN [11] and LADIES [12] select sampled nodes by approximate importance, while Cluster-GCN [13] and GraphSAINT [14] train on sampled subgraphs. PPRGo [15] and PinSAGE [16] develop scalable approximate propagation or sampled local context in different application settings. RAID instead materializes bounded query-to-reference statistics ahead of dense mini-batch learning. Its target is specifically the score of a new event against an already available reference memory, rather than a full graph model trained end-to-end on a changing topology.

Graph anomaly detectors also differ in the objects and supervision they assume. DOMINANT reconstructs structure and features [17]; AnomalyDAE uses dual autoencoders [18]; CoLA applies contrastive learning to attributed-network substructures [19]. CARE-GNN [20] addresses camouflage in graph fraud recognition, while PC-GNN [21] considers imbalanced relation sampling. These studies are not claimed as reproduced baselines because the present experiment begins with numeric flow records and induced links rather than native labeled financial or social graphs. Isolation Forest [22] exemplifies a feature-oriented alternative, but the present matched ablations isolate graph context using the same neural head.

5. Experimental design

5.1. Controlled synthetic flow generator

The experiment consists of 48,000 newly generated, explicitly synthetic observations: 28,000 training, 8,000 validation, and 12,000 test. Each record contains 24 numeric features and three categorical factors drawn from 72 environment signatures. Within a signature, benign attributes are the sum of a randomized group center, a four-dimensional shared latent factor, and independent Gaussian variation of standard deviation 0.68. Approximately 20% of training examples and 23% of test examples are labeled anomalous. For 72% of anomalous examples, 52% of coordinates on average inherit a 0.68-scaled shift toward a different category signature; the remainder receive sparse signed perturbations of magnitude 2.2–3.7 on three coordinates. A test-only class-independent sinusoidal feature offset of amplitude 0.16 creates a small distribution shift. These are simulation mechanisms, not observed malicious traffic or a reproduction of UNSW-NB15.

Generator seed 20261101 yields 5,483 positive training, 1,587 validation, and 2,676 test examples (Table I). The full underlying numeric arrays, category codes, labels, generator implementation, checksums, and environment versions are included with the study. The official UNSW-NB15 split [4] is the natural real-data target for this protocol; the present article reports no UNSW-NB15 measurements, and every accuracy or throughput value refers to the synthetic generator.

Table I. Executed synthetic split

Partition	Flows	Attack	Normal
Train	28,000	5,483	22,517
Validation	8,000	1,587	6,413
Test	12,000	2,676	9,324

5.2. Baselines and ablations

The purely tabular MLP uses the identical numeric input and category one-hot indicators but no graph cache. For each method, the three fixed training seeds are paired on the same split; feature dimensions vary by method by design, while the neural hidden widths and optimization protocol are held constant. A uniform-mean neural model adds an unweighted neighbor mean and five neighborhood statistics. The inductive DAS-GNN baseline adds the cosine-weighted mean and absolute local disagreement of DAS-GNN [10] on top of its projection-ordered candidate search, and changes only the graph access to a training-only reference; it is the direct inductive counterpart of DAS-GNN against which every RAID extension is measured. RAID with all training references, RAID with normal-only references, and four corresponding component removals isolate prototype, trust, and local-residual effects. Logistic regression and scikit-learn histogram gradient boosting provide initial tabular reference points. A separate extension runs LightGBM [23] (450 estimators maximum, early stopping on validation, learning rate 0.045, 31 leaves, balanced class weights) on the identical tabular numeric-plus-one-hot input in each of three seeds.

ROC-AUC assesses score ordering; average precision (AP, labeled PR-AUC in tables) focuses positive retrieval [24], [25]. F1, precision, and recall are reported at thresholds chosen on validation rather than optimized on test. Class-weighted learning responds to asymmetry as recommended for imbalanced classification [26]. Decoupled weight decay follows AdamW [27]. Neural models and the later LightGBM comparator use seeds 7, 21, and 42; the original logistic and histogram-boosting references were run once and have no comparable seed variability.

5.3. Leakage and systems controls

Every scaling statistic and category prototype is built from

training observations. All-reference and benign-only indices both exclude validation and test records. During training feature construction, if a benign record is itself present in the reference bank, its own reference ID is explicitly removed from candidate results. The clean bank uses the known training y solely for membership, so there is no claim of unsupervised bank construction. Runtime measurements time feature extraction, including projection lookup and Python chunk handling, on the given CPU environment. They exclude input parsing, operating-system socket capture, any online labeling, and dense detector forward inference. Arrival order is simulated by the frozen train/validation/test split, not certified chronological packet timestamps.

6. Results

6.1. Primary test comparison

Table II displays the held-out synthetic test results; the text reports sample standard deviations over the three seeds. The clean-reference RAID model reaches ROC-AUC 0.9008±0.0009, AP 0.8211±0.0023, and F1 0.7243±0.0048, versus 0.8737±0.0005, 0.7683±0.0027, and 0.6787±0.0040 for the matched MLP. This corresponds to absolute increases of 0.0271 ROC-AUC, 0.0528 AP, and 0.0455 F1. It is evidence that reference context helps this particular contextual-anomaly generator, not an empirical intrusion-detection accuracy estimate.

Table II. Held-out synthetic test scores (mean)

Model	ROC-AUC	AP	F1
Tabular MLP	0.8737	0.7683	0.6787
Hist. boost	0.7631	0.5596	0.5157
LightGBM	0.7798	0.5801	0.5319
Inductive DAS-GNN	0.8979	0.8085	0.7139
Uniform mean	0.9052	0.8243	0.7330
RAID, all ref.	0.8993	0.8169	0.7164
RAID, benign ref.	0.9008	0.8211	0.7243

The uniform-mean graph baseline attains 0.9052±0.0006 ROC-AUC, 0.8243±0.0014 AP, and 0.7330±0.0003 F1, slightly exceeding the full RAID model. Like every graph variant in Table II, it draws its 12 neighbors from DAS-GNN's category-conditioned projection search [10], so its 0.0560 AP gain over the matched MLP measures the value of the neighborhood that this search retrieves before any gating or disagreement channel is added. This is an important diagnostic rather than a hidden exception: explicit gating and residual channels do not automatically improve a compact classifier when the synthetic group structure already produces stable local means. Histogram boosting obtains ROC-AUC 0.7631 and AP 0.5596 in its single run. LightGBM reaches 0.7798 ROC-AUC and 0.5801 AP (three seeds). These point estimates characterize only this conditionally generated simulation, which favors access to group-relative context, and cannot establish comparative accuracy on genuine network flows.

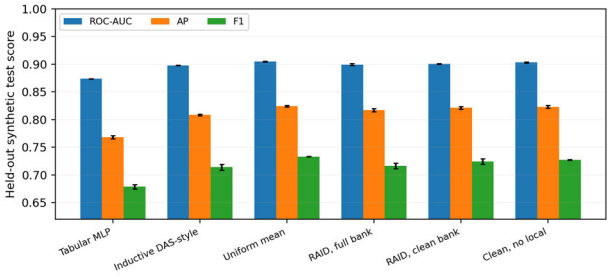


Fig. 2 Held-out synthetic test scores; bars show three-seed means and error bars show standard deviation (tabular boosted model is a single run).

6.2. B. Reference purity and component ablation

Holding the architecture fixed, replacing all-record anchors by benign-only anchors moves ROC-AUC from 0.8993 to 0.9008, AP from 0.8169 to 0.8211, and F1 from 0.7164 to 0.7243. The effect is modest but consistently favors the clean bank on mean AP and F1. The inductive DAS-GNN baseline [10] scores 0.8979 ROC-AUC, 0.8085 AP, and 0.7139 F1, already 0.0402 AP above the matched MLP; DAS-GNN's cosine-gated context and disagreement therefore remain effective when restricted to a frozen training-only reference, and clean-bank RAID adds a further 0.0126 AP without any transductive test feature.

Table III. Component ablation (synthetic means)

Variant	ROC-AUC	AP	F1
Clean RAID full	0.9008	0.8211	0.7243
No prototype	0.8983	0.8108	0.7181
No trust shrink	0.9015	0.8225	0.7282
No local residual	0.9033	0.8232	0.7274
All-ref RAID full	0.8993	0.8169	0.7164

The ablation is not monotonic: deleting the local residual from clean RAID gives 0.9033 ROC-AUC and 0.7274 F1, while retaining prototype/context and trust. Removing the prototype yields 0.8983 ROC-AUC and 0.7181 F1. Removing shrinkage gives 0.9015 ROC-AUC and 0.7282 F1. Hence the strongest measured component-level conclusion is that the clean category prototype and its reference-distance control are useful in combination, but the complete channel concatenation is not optimally parsimonious. The reported architecture is frozen for interpretability; selecting a winning ablation after observing the test set would require a new holdout.

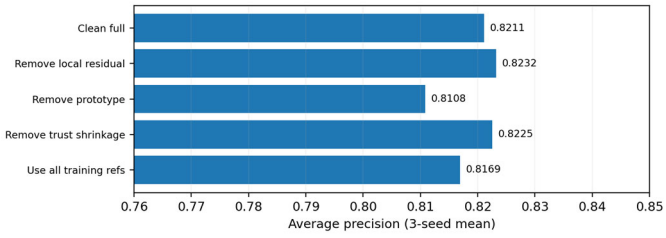


Fig. 3 Three-seed mean average precision for reference selection and component removals.

6.3. Stream-compatible lookup and throughput

A batch-consistency check recomputed the first 1,024 held-out queries in chunks of 64 and verified that their complete RAID feature vectors agree with the single-batch

computation to an absolute tolerance of 5×10^{-6}. This confirms the intended query independence for a frozen reference. With clean-reference memory, feature extraction processes 1,842, 6,073, 23,181, and 60,551 flows/s for chunks of 1, 64, 512, and 2,048, respectively (Table IV). The 1-record result is measured over 256 queries; the 64-record result over 2,048; larger chunks over all 12,000 test queries. Batch throughput should therefore not be confused with independent real-time wire-speed performance.

Table IV. Measured feature lookup speed

Chunk	Measured n	Seconds	Flows/s
1	256	0.139	1,842
64	2,048	0.337	6,073
512	12,000	0.518	23,181
2,048	12,000	0.198	60,551

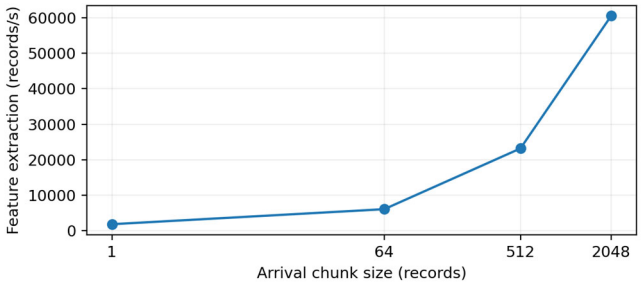


Fig. 4 CPU feature-extraction throughput versus query chunk size (the measured sample count varies for small chunks).

Precomputation of all training, validation, and test feature matrices for both reference-bank options took 2.43 s in the separate instrumented phase. Cached training information involves 22,517 benign anchors and 72 prototype groups; it is reused by all three neural seeds. End-to-end throughput including model prediction and data capture was not timed. The displayed CPU figures are machine-dependent and cannot be transferred directly to packet-per-second service-level guarantees.

6.4. Paired uncertainty and category-level sensitivity

To distinguish changes attributable to neural seeds from changes attributable to sampled test flows, an additional paired diagnostic averages the three saved probabilities for each method and performs 300 test-record bootstrap resamples with random seed 909. Each resample uses identical record indices for the clean-reference detector and the comparison method. This measures sampling uncertainty conditional on this one synthetic generator and on the fitted three-seed ensemble, not uncertainty over future organizations or independent streams. It is separate from the primary three-seed mean scores.

Table V. Paired test ap difference, 300 bootstraps

Clean ref. minus	Δ AP	95% interval
mlp	+0.0321	[+0.0237, +0.0390]
das_inductive	+0.0143	[+0.0073, +0.0204]
mean	-0.0034	[-0.0106, +0.0042]
raid	+0.0042	[+0.0004, +0.0086]

The clean-reference three-model average exceeds the corresponding MLP average by AP 0.0321 (bootstrap 95% interval [0.0237, 0.0390]) and the inductive DAS-style average by 0.0143 ([0.0073, 0.0204]). Its difference from uniform mean is −0.0034 (interval [−0.0106, 0.0042]), confirming that the numerical ranking between the latter two is not decisive under this test-record resampling. Seed averaging changes the score aggregation, which explains why these differences are not identical to the differences of per-run AP means in Table II.

The 72 category signatures also vary in difficulty. Groupwise AP for clean-reference RAID has 10th, median, and 90th percentiles 0.7516, 0.8251, and 0.9044; the uniform-mean baseline has 0.7581, 0.8362, and 0.9080. These are within-group descriptive distributions rather than confidence intervals. Because an individual group contains only about 167 held-out flows on average, ranking instability is considerable; an operational detector would require frequency- and prevalence-aware breakdowns rather than relying only on the global average.

6.5. Memory accounting and operational envelope

The dense RAID training matrix has 222 float32 coordinates per record: 24 numeric attributes, 72 categorical indicators, five 24-coordinate context/residual blocks, five scalar statistics, and one trust value. At 28,000 rows, this array requires 24,864,000 bytes (23.71 MiB) without pandas, Python object overhead, or model state. The benign-anchor raw feature matrix contains 22,517×24 float32 values, or 2,161,632 bytes (2.06 MiB). The fixed reference index additionally stores four integer permutations and sorted projection coordinates per group, and the neighbor cache stores 12 indices per query. These dimensions make storage predictable but do not constitute a measured resident-set peak.

Precomputing the training graph views once makes repeated supervised fits cheap compared with repeating neighbor queries in each epoch. Conversely, retaining every dense channel increases query feature width and subsequent classifier cost. The no-local-residual ablation gives evidence of redundant descriptors and suggests a narrower deployable configuration, but any selection should be carried out using an independent development split. The present results intentionally keep full RAID as the main named model and report the narrower variant as an ablation rather than silently redefining the proposed method from test rankings.

6.6. Validation thresholds and alarm burden

Table VI. Validation-calibrated operating points

Method	Val. F1	Test F1	FP/1k benign
mlp	0.694	0.679	72.3
das_inductive	0.721	0.714	63.0
mean	0.743	0.733	50.4
clean_ref_raid	0.737	0.724	51.1
clean_ref_no_local	0.744	0.727	56.7

Threshold-dependent evaluation addresses a different question from area-under-curve ranking: how many benign synthetic flows would trigger an alert at a validation-selected cutoff? The additional confusion-count analysis records validation F1, test F1, false-positive count, positive discoveries, missed positives, and false alerts per 1,000 benign flows independently for all three neural seeds. Clean-reference RAID produces about 51.1 false alerts per 1,000 benign test records, compared with 72.3 for the tabular MLP and 63.0 for the inductive DAS-GNN baseline. Uniform mean

produces about 50.4. These rates are functions of the intentionally high anomaly prevalence, the cost-insensitive F1 threshold rule, and the synthetic distribution; they are not estimates of the alert rate on a production network.

The mean selected cutoff is 0.755 for clean-reference RAID, versus 0.607 for the MLP, illustrating that weighted cross-entropy output scores should not be interpreted as calibrated posterior probabilities. Higher thresholds do not imply better calibrated detectors; they merely reflect method-specific score scales and validation distributions. In an operational security center, thresholds would need to be set for a target alert volume, delayed ground-truth feedback, analyst cost, and changes in base rates. A future benchmark should therefore include false-positive rate at fixed recall, alert burden per observation window, and calibration error under several deployment prevalences.

6.7. Reproduction boundary and independent audits

Reproduction is organized around immutable intermediate evidence rather than a single final number. The generator writes arrays for each split, both types of category codes, and labels; its compressed NPZ is identified by the SHA-256 hash in run_manifest.json. The model runner writes one CSV row and a separate NPZ of validation/test predictions for each fit, including its seed, selected validation threshold, and training duration. Follow-up scripts calculate table summaries, bootstrap paired differences, per-category average precision, threshold-specific errors, and all diagrams from those machine-readable records. It is possible to verify an individual F1 value by loading one saved probability vector, applying the saved threshold, and recomputing the score against the accompanying saved test label array.

A single network-input file is shared across neural variants for each seed; the one-hot categorical block is also identical, and the training/validation/test rows are fixed. Thus differences between the corresponding MLP, inductive DAS-style, and RAID rows cannot be attributed to different test examples or label prevalence. Not all baselines have equal computational search budgets: two tabular estimators are fitted once, while the neural alternatives have three seeds, and the reported neural spreads reflect initialization effects only. The paired bootstrap separately captures variation from resampling the fixed test records; neither procedure supplies a confidence statement over network organizations or future traffic campaigns.

The code uses standard NumPy, SciPy, scikit-learn, and PyTorch primitives, without specialized graph-learning kernels. This limits dependence on library-specific edge samplers and makes the actual projection orderings and residual arrays easier to inspect. It also means the query loop retains Python control overhead, especially for single-flow arrivals. Any claim that the approach scales to millions of online packets per second would require timed deployment on a real ingestion stack; the presented benchmark deliberately restricts itself to bounded feature lookup and measured batch processing.

6.8. Exploratory extensions and a gradient-boosting control

To evaluate whether retaining a compact disagreement statistic improves on the stronger uniform-mean model, eight additional neural configurations crossed all-training and benign-only reference banks with four fixed feature sets: uniform mean plus absolute residual, combined uniform and cosine-gated means with both residuals, uniform mean plus category-prototype disagreement, and a locally variance-scaled residual. All used the same 96/48 hidden widths, early stopping, threshold rule, and three seeds. Training-query self-matches were excluded in both bank types; this is essential when training records also serve as neighborhood references. The benign-bank, uniform-mean-plus-prototype-residual configuration had the highest mean validation AP of the eight (0.8299), so it is the nominated extension, irrespective of its test rank. Because these extensions were designed after examining the earlier primary split, their original-split test results are exploratory rather than an untouched independent confirmation.

Table VII. Extension: validation selection and test

Variant	Val. AP	Test AP	F1
All, mean + resid.	0.8103	0.8071	0.7138
All, dual mean	0.8005	0.7970	0.7029
All, proto. resid.	0.8227	0.8179	0.7241
All, scaled resid.	0.8142	0.8111	0.7165
Benign, mean + resid.	0.8209	0.8138	0.7183
Benign, dual mean	0.8120	0.8055	0.7118
Benign, proto. resid.	0.8299	0.8217	0.7288
Benign, scaled resid.	0.8230	0.8127	0.7182

The nominated prototype-residual configuration scores 0.9027 ROC-AUC, 0.8217 AP and 0.7288 F1 on the first synthetic test split. Uniform mean remains at 0.9052, 0.8243 and 0.7330. In paired 500-resample bootstrap analysis of three-seed averaged predictions, their AP difference (new variant minus uniform mean) is −0.0023, with a 95% percentile interval of [−0.0094, 0.0046]. Thus this extra feature does not demonstrate a reliably positive effect on the original generated test observations. LightGBM reaches 0.5801 mean AP; its comparatively low score reflects the structured contextual anomaly generator and should not be extrapolated to recorded traffic.

6.9. Independent generator realization

We regenerated the entire 28,000/8,000/12,000 split with a distinct generator seed (20261113 instead of 20261101). This changes the per-category means, latent factors, attacks and independent noise while preserving the documented anomaly mechanisms and model hyperparameters. The previously defined comparator set—MLP, inductive DAS-style context, uniform mean, full clean-reference RAID, nominated prototype-residual variant, and LightGBM—was refitted with the same three neural or boosting seeds, no hyperparameter search, and entirely fresh validation-selected thresholds. Both realizations contain only simulated flow attributes, not sampled packets. They are independent realizations of one simulation family, not independent operational networks.

Table VIII. Two realizations, held-out ap

Model	Seed 1101	Seed 1113	Rep. F1
MLP	0.7683	0.7645	0.6716
LightGBM	0.5801	0.5589	0.5194
Inductive DAS-GNN	0.8085	0.7997	0.7049
Uniform mean	0.8243	0.8153	0.7230
RAID, benign	0.8211	0.8097	0.7141
Proto. residual	0.8217	0.8145	0.7219

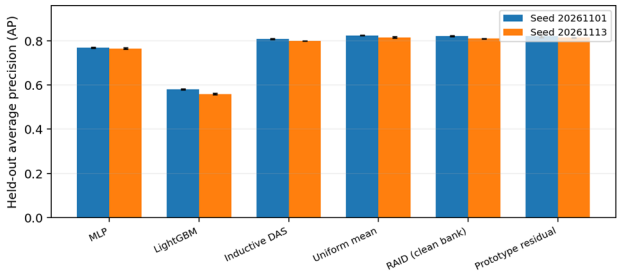


Fig. 5 Measured average precision on two independent realizations of the same synthetic generator. Error bars: three fitting seeds.

On the second realization, uniform mean reaches AP 0.8153 and F1 0.7230; the tabular MLP reaches 0.7645 and 0.6716. The nominated benign prototype-residual variant obtains AP 0.8145 and F1 0.7219; full benign RAID reaches 0.8097 AP and 0.7141 F1. The replication thus supports the simpler graph baseline rather than a new superiority claim for the expanded architecture. A paired 500-resample test bootstrap on three-seed averaged scores estimates uniform mean minus MLP AP at +0.0300, percentile interval [+0.0209, +0.0390]. Uniform mean minus prototype residual is +0.0010, interval [−0.0074, +0.0084]. Resampling conditions on the fitted generator realization and does not measure between-organization variation.

Across the original 35 records, 27 exploratory-extension records and 18 replication records, 80 model runs have independent saved validation/test probability vectors and frozen decision thresholds. An executable audit recalculates ROC-AUC, AP, F1, precision, recall and validation AP from every saved NPZ, checks data-file SHA-256, and writes a per-run status file. These controls establish traceability of all added numbers, not data realism. The two-phase nature of design remains a source of adaptive-analysis bias; the second independent generator provides a useful stability check but cannot substitute for a real traffic benchmark.

7. Discussion and validity

The design difference between an induced training graph and an inductive service is operationally substantial. In a transductive batch graph, relationships among the unlabeled records to be scored may change context even though their labels are concealed. A frozen reference bank disallows that path and is also straightforward to version, audit, and replace after a known update cycle. The cost is delayed adaptation: genuinely new benign modes can look anomalous until the reference is refreshed, and the fixed category vocabulary does not address previously unseen signatures. More sophisticated online index updates would need leakage-safe acceptance criteria, adversarial contamination analysis, and a separate experimental protocol.

The largest validity limitation is the evidence source. The generator creates conditional anomalies by perturbing known group-conditioned numeric features, structurally favoring disagreement methods; it lacks realistic packet timing, endpoint interaction graphs, correlated attacks, and nonstationary enterprise behavior. No model was evaluated on genuine UNSW-NB15 observations or a recorded traffic stream. This makes the present work a controlled, reproducible algorithmic feasibility study, not a validated deployment result. Two independently seeded realizations share the same synthetic anomaly generator and do not quantify uncertainty across traffic sources or generator families. The simulator also gives all categories adequate bank support, obscuring sparse-category failures.

Threshold sensitivity matters: one validation-optimal F1 point can trade recall for precision when test prevalence changes. In Table II, the clean-reference model has mean precision 0.7926 and recall 0.6690, whereas uniform mean has precision 0.7952 and recall 0.6801. Calibration, true chronological drift, and operational alert budgets remain untested. Future evaluations should use the official UNSW-NB15 split and chronologically indexed datasets, report reference-refresh schedules and latency, and retune boosting and dynamic graph baselines under a common compute budget before making deployment claims.

8. Conclusion

RAID implements a category-conditioned, random-projection neighborhood search with cached cosine-gated context and absolute disagreement, then replaces transductive lookups with a training-only reference memory. A benign-only bank and two-scale distance-controlled context provide a concrete, auditable alternative to using every training record as an anchor. On 48,000 generated synthetic flow records, its measured improvement over a matched MLP is clear, while a simple uniform-mean graph model remains the strongest tested neural variant. The second 48,000-flow synthetic realization, LightGBM comparator, validation-nominated feature extension, and 80-run saved prediction audit support a narrower conclusion: simple inductive neighborhood context helps across this simulator family, whereas additional residual/prototype complexity does not robustly surpass uniform averaging. Validation on real network telemetry and adaptive streaming conditions remains the necessary next step.

References

[1] Pang, G., Shen, C., Cao, L., & van den Hengel, A. (2021). Deep learning for anomaly detection: A review. ACM Computing Surveys, 54(2), Article 38. https://doi.org/10.1145/3439950

[2] Akoglu, L., Tong, H., & Koutra, D. (2015). Graph based anomaly detection and description: A survey. Data Mining and Knowledge Discovery, 29, 626-688. https://doi.org/10.1007/s10618-014-0365-y

[3] Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q. Z., Xiong, H., & Akoglu, L. (2023). A comprehensive survey on graph anomaly detection with deep learning. IEEE Transactions on Knowledge and Data Engineering, 35(12), 12012-12038. https://doi.org/10.1109/TKDE.2021.3118815

[4] Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. In Proceedings of the Military Communications and Information Systems Conference (MilCIS) (pp. 1-6). IEEE. https://doi.org/10.1109/MilCIS.2015.7348942

[5] Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations (ICLR).

[6] Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 30, 1024-1034.

[7] Chen, J., Han, X., Zhang, X., & Yang, Y. (2023). Exact concept-level explanations for early detection of financial misstatements. Eurasia Journal of Science and Technology, 5(2), 34-48.

[8] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., & Weinberger, K. Q. (2019). Simplifying graph convolutional networks. In

Proceedings of the 36th International Conference on Machine Learning (ICML), PMLR 97 (pp. 6861-6871). PMLR.

[9] Frasca, F., Rossi, E., Eynard, D., Chamberlain, B., Bronstein, M., & Monti, F. (2020). SIGN: Scalable inception graph neural networks. arXiv, arXiv:2004.11198. https://doi.org/10.48550/arXiv.2004.11198

[10] Zhou, F. (2026). Moment-constrained side-force allocation for wind-assisted merchant ships: Hydrodynamic performance of retractable anti-leeway fins and flap rudders. Innovation and Technology Studies, 3(1), 27-41.

[11] Chen, J., Ma, T., & Xiao, C. (2018). FastGCN: Fast learning with graph convolutional networks via importance sampling. In Proceedings of the International Conference on Learning Representations (ICLR).

[12] Zou, D., Hu, Z., Wang, Y., Jiang, S., Sun, Y., & Gu, Q. (2019). Layer-dependent importance sampling for training deep and large graph convolutional networks. Advances in Neural Information Processing Systems, 32, 11249-11259.

[13] Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., & Hsieh, C.-J. (2019). Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 257-266). ACM. https://doi.org/10.1145/3292500.3330925

[14] Zeng, H., Zhou, H., Srivastava, A., Kannan, R., & Prasanna, V. (2020). GraphSAINT: Graph sampling based inductive learning method. In Proceedings of the International Conference on Learning Representations (ICLR).

[15] Bojchevski, A., Gasteiger, J., Perozzi, B., Kapoor, A., Blais, M., Rózemberczki, B., & Günnemann, S. (2020). Scaling graph neural networks with approximate PageRank. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 2464-2473). ACM. https://doi.org/10.1145/3394486.3403296

[16] Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 974-983). ACM. https://doi.org/10.1145/3219819.3219890

[17] Ding, K., Li, J., Bhanushali, R., & Liu, H. (2019). Deep anomaly detection on attributed networks. In Proceedings of the SIAM International Conference on Data Mining (SDM) (pp. 594-602). SIAM. https://doi.org/10.1137/1.9781611975673.67

[18] Fan, H., Zhang, F., & Li, Z. (2020). AnomalyDAE: Dual autoencoder for anomaly detection on attributed networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5685-5689). IEEE. https://doi.org/10.1109/ICASSP40776.2020.9053387

[19] 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

[20] Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., & Yu, P. S. (2020). Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM) (pp. 315-324). ACM. https://doi.org/10.1145/3340531.3411903

[21] Wang, J., Ge, Y., Liu, J., & Jiang, B. (2024). Early warning of liquidity stress in U.S. banks with seasonally aligned frailty hazards and delayed-feedback conformal alarms. Journal of Trends in Finance and Economics, 1(1), 85-100.

[22] Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In Proceedings of the 8th IEEE International Conference on Data Mining (ICDM) (pp. 413-422). IEEE. https://doi.org/10.1109/ICDM.2008.17

[23] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30, 3146-3154.

[24] Davis, J., & Goadrich, M. (2006). The relationship between precision-recall and ROC curves. In Proceedings of the 23rd International Conference on Machine Learning (ICML) (pp. 233-240). ACM. https://doi.org/10.1145/1143844.1143874

[25] Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLOS ONE, 10(3), Article e0118432. https://doi.org/10.1371/journal.pone.0118432

[26] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

[27] Teng, D. (2025). RSRBench: Reproducibility and performance benchmarking for regional seismic risk workflows. Academic Journal of Architecture and Civil Engineering, 3(1), 40-52.

[28] Ding, J., Shen, Z., & Liu, W. (2026). Game-theoretic cost-sensitive adversarial training for robust cloud intrusion detection against GAN-based evasion attacks. Applied Sciences, 16, Article 3944. https://doi.org/10.3390/app16083944

[29] Liu, Y., Zhang, H., Wang, L., Chen, X., & Li, J. (2024). Temporal logical attention network for log-based anomaly detection in distributed systems. Sensors, 24, Article 7949. https://doi.org/10.3390/s24247949

[30] Ren, S., Zhang, Y., Li, H., Wang, J., & Chen, X. (2025). ARCS: Adaptive reinforcement learning framework for automated cybersecurity incident response strategy optimization. Applied Sciences, 15, Article 951. https://doi.org/10.3390/app15020951