THE SCORE IS NOT THE MODEL: FOUR PROTOCOL FAILURE MODES IN OPEN GRAPH BENCHMARK REPORTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Leaderboard scores are read as measurements of a model. On the Open Graph Benchmark (OGB) that reading fails in four independent ways, and we isolate each one by holding the model fixed and changing only the reporting protocol. **P1, tail reading:** for link prediction Hits@K is, by construction, the fraction of positive pairs that outscore the K-th negative pair, so its value is set by the extreme upper tail of the negative distribution rather than by ranking quality; on ogbl-ddi the $K{=}20$ threshold (a Common-Neighbour score of 472) sits above the median positive score (358), and a Common-Neighbour scorer with AUC 0.952 reports Hits@$20 = 0.1773$. **P2, the undeclared sampler:** holding the scorer and the number of negatives fixed, swapping OGB's negative set for uniformly drawn pairs drives the same scorer from 0.1773 to 0.0000, and sweeping K from 20 to $95{,}599$ drives it to 0.9997. **P3, the two-route pipeline:** a pipeline that computes features for positive pairs and a literal zero block for negative pairs scores Hits@$20 = 1.000000$, and still scores 1.000000 when the features are replaced by the constant 1; the official evaluator returns the value without complaint. **P4, reused splits:** on ogbn-products validation accuracy exceeds test accuracy by 22.1 points, enabling validation edges as message-passing input raises ogbl-ddi from 0.5800 ± 0.0461 to 0.7792 ± 0.0518, and the validation spread between post-processing configurations is up to $14\times$ narrower than the test spread they produce. We calibrate the instrument before using it: our structural heuristics reproduce published ogbl-ddi values to the precision at which those values are published. Every number is reproducible with the released toolkit, and the paper contributes no new architecture—deliberately so.

1 INTRODUCTION

A leaderboard number is usually read as a property of a model. In practice it is the composition of six decisions—how the split is consumed, how features are computed, which model is trained, which post-processing is applied, how configurations are selected, and how the score is computed. If any one is undeclared or degenerate, the result is still a real number, still comparable in the reader's mind to every other number in the table, and still not a statement about the model. This paper isolates four places where that happens on the Open Graph Benchmark (OGB) (Hu et al., 2020), among the most carefully constructed graph benchmarks in use.

Our method is substitution, not ablation: we fix the scorer and change one stage of the protocol, so the change is attributable to the protocol and not to the model. The scorers are zero-parameter structural heuristics or single logistic regressions, which makes them cheap to reproduce and impossible to attribute to tuning, and all four findings use the *official* OGB evaluator, so no claim depends on our re-implementation of a metric.

We name the four modes so that they can be checked by name; Figure 1 states each one and the number that establishes it.

Sections 4 to 7 isolate the four modes in turn, each by holding the scorer fixed and changing one stage of the protocol.

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.

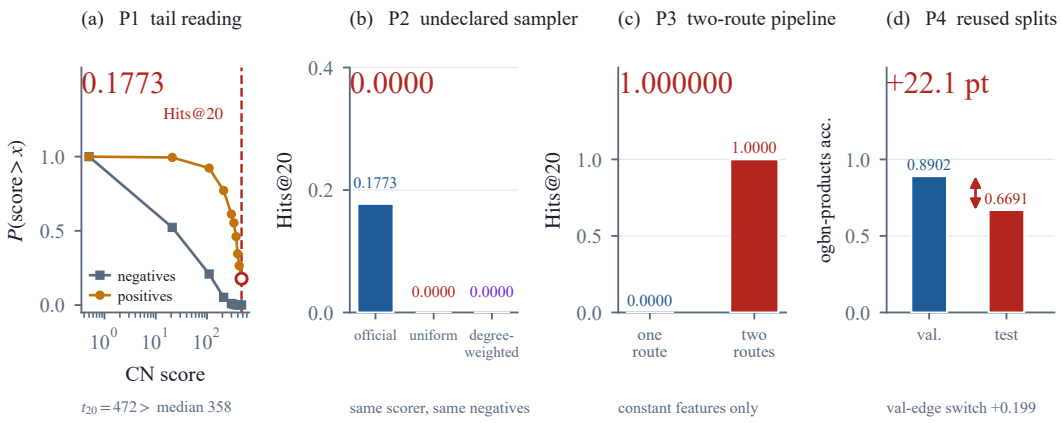

Figure 1: A reported number is not the model. Each panel names one failure mode and gives the number that establishes it, all from the official evaluator on the official split: **(a)** tail reading, where the $K{=}20$ threshold sits above the median positive score; **(b)** the undeclared sampler, which alone moves the value to 0.0000; **(c)** the two-route pipeline, which scores 1.000000 on constant features; **(d)** reused splits, where the arrow spans the validation-to-test gap.

We then turn the findings into something usable: seven checks that a reader or reviewer can run on any reported number (Section 8), a five-benchmark re-measurement whose purpose is to be reproducible rather than competitive, and a released toolkit with one script per probe.

What this paper is not. It is not a claim that OGB is broken, and it is not a claim that any particular published entry is wrong. Each probe targets a decision that the reporting format leaves implicit; the benchmark's construction is the reason the failures are informative rather than trivial, because OGB's splits, negative sets and evaluators are otherwise unusually well specified.

2 Related Work

Benchmark validity. Shchur et al. (2018) showed that GNN evaluation is sensitive to splits and to hyperparameter budgets, and Li et al. (2023) catalogued pitfalls specific to link prediction, including the sensitivity of Hits@K to negative sampling; P1 supplies the identity behind that sensitivity and P2 quantifies it on a fixed scorer and a fixed negative-set size. The same fragility has been measured in neighbouring fields. Bouthillier et al. (2021) decompose the variance of a benchmark comparison into its seed, hyperparameter and data components and show that the seed term alone can exceed the differences being reported; Reimers & Gurevych (2017) and Lucic et al. (2018) find that reporting a distribution rather than a single number changes which method appears to win; Henderson et al. (2018) find the ranking of reinforcement-learning agents to depend on seeds and on the evaluation protocol; and Gorman & Bedrick (2019) show that a widely used standard split can be replaced without anyone noticing. Melis et al. (2018) and Dodge et al. (2019) showed that reported differences shrink or reverse once tuning budgets are equalised, which is the phenomenon P4 measures on a graph benchmark, and Kapoor & Narayanan (2023) and D'Amour et al. (2022) systematised the leakage taxonomy and the unreported-pipeline argument that P4 instantiates. What the four probes below add is not the existence of the problem but a decomposition of it on one benchmark, where each stage can be held fixed while the next is varied.

The benchmark and its post-processing. OGB (Hu et al., 2020) supplies standardised splits, evaluators and leaderboards; the properties we probe follow from design choices made for good reasons, not from errors. Hits@K against a fixed negative set is a sound tail statistic, and the validation-edge switch exists because ogbl-collab's split is time-based and its validation edges are genuinely available at test time. Correct & Smooth (Huang et al., 2021) is the post-processing step used in P4, and the sign flip we report is consistent with work showing that label propagation and simple models are strong but dataset-dependent baselines (Wu et al., 2019; Gasteiger et al., 2019). This paper contributes no new architecture and makes no state-of-the-art claim: Section 6 argues that the absence

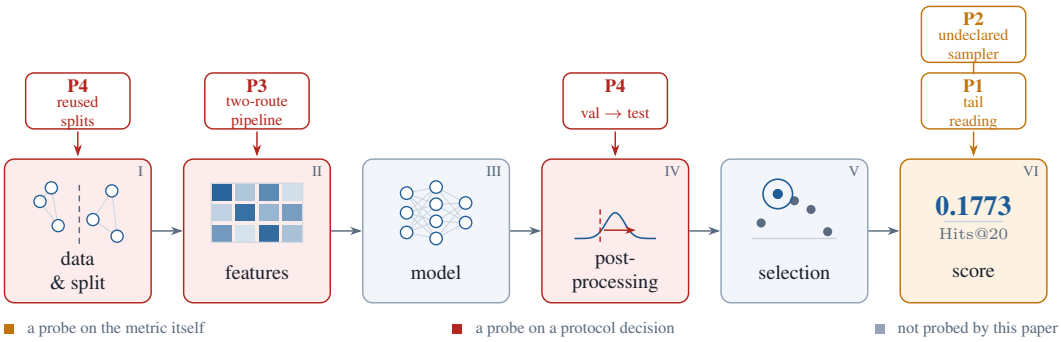

Figure 2: A reported number is the composition of six stages. The four probes each fix all other stages and change exactly one. P3 and P4 concern decisions that are usually left implicit; P1 and P2 concern the metric itself.

is the point, because a pipeline can reach Hits@1 with constant features, so the defence has to be a protocol rather than a stronger model. The models that populate these leaderboards are the standard ones—graph convolution (Kipf & Welling, 2017), sampling-based inductive embedding (Hamilton et al., 2017), attention (Veličković et al., 2018; Brody et al., 2022), and message passing in general (Gilmer et al., 2017; Xu et al., 2019)—together with a smaller line of work that re-injects the structural signal message passing smooths away (Yun et al., 2021; Dong et al., 2024; Chamberlain et al., 2023). The benchmark itself has since been extended to graphs far larger than the five used here (Hu et al., 2021a).

3 What a Reported Number Is

Tasks and metrics. We use five OGB datasets: three node-classification tasks (ogbn-arxiv, ogbn-products, ogbn-proteins) scored by accuracy or ROC-AUC, and two link-prediction tasks (ogbl-ddi, ogbl-collab) scored by Hits@K with $K{=}20$ and $K{=}50$ respectively. Node features, splits and negative sets are taken from the official loaders without modification.

The reported value as a composition. Write a reported score as

$$\hat{v} \;=\; \mathcal{E}\big(\,\mathcal{S}\circ\Pi\circ\Theta\,(\mathcal{D},\;\mathcal{R})\,\big), \tag{1}$$

where $\mathcal{D}$ is the dataset, $\mathcal{R}$ the set of protocol decisions, Θ the model that is trained, Π the post-processing, $\mathcal{S}$ the selection rule that picks among candidate configurations, and $\mathcal{E}$ the evaluator. Figure 2 maps the four probes onto these stages; they are ordered by how little a reader of a paper typically knows about the stage, not by how often the stage goes wrong.

Hits@K **under the official evaluator.** For link prediction the OGB evaluator implements

$$\text{Hits@}K \;=\; \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} \mathbb{1}[\,s(p) > t_K\,], \qquad t_K \;=\; K\text{-th largest of } \big\{s(n) : n\in\mathcal{N}\big\}, \tag{2}$$

where $\mathcal{P}$ is the set of positive test pairs, $\mathcal{N}$ the fixed negative multiset, s the scorer, and t_K the K-th largest negative score. All link-prediction numbers in this paper are produced by `ogb.linkproppred.Evaluator`, including the diagnostics.

Calibrating the instrument. A probe can only be believed if the measuring device is known to work, so before using Hits@K to make a negative point we check that our implementation reproduces values already on the record. In Table 1 three of the four standard structural heuristics reproduce a published anchor to within one unit in its last printed digit. That small table has a large job: it is the reason the anomalies in Sections 4–7 can be attributed to the protocol rather than to our code.

Table 1: Instrument calibration on ogbl-ddi, official evaluator, official test split. Our value for each heuristic against the published anchor for the same heuristic and the same dataset. Agreement is stated at the precision at which each anchor is printed, so the rows are not compared at a common number of digits.

Scorer	This paper	Published anchor	Anchor source
Common Neighbour	0.177296	17.73 (=0.1773)	Li et al. (2023)
Adamic–Adar	0.186075	0.1861	Open Graph Benchmark (2026)
Resource Allocation	0.062267	6.23 (=0.0623)	Li et al. (2023)
Preferential Attach.	0.039262	not reported	—

Anchors are quoted as printed by their sources: the first and third as percentages, the second as a fraction. The fourth heuristic has no published anchor on this dataset and is reported for completeness. Our values are 17.7296, 0.186075, 6.2267 and 0.039262 respectively when printed to the same precision.

Table 2: Ogbl-ddi, official evaluator. The official Hits@20 and the fraction of positives scoring above the 20th largest negative score agree to machine precision for all four heuristics ($|\mathcal{N}| = 95{,}599$, $|\mathcal{P}| = 133{,}489$, 4,267 nodes). For Common Neighbour the threshold exceeds the median positive score, while ranking quality as measured by AUC is high.

Scorer	Hits@20	frac. $> t_{20}$	t_{20}	median positive	AUC	AP
Common Neighbour	0.177296	0.177296	472	358	0.952	0.9664
Adamic–Adar	0.186075	0.186075	68.6	52.4	0.954	0.9683
Resource Allocation	0.062267	0.062267	0.72	0.41	0.965	0.9752
Preferential Attach.	0.039262	0.039262	1,328,448	589,500	0.878	0.9012

4 P1: Tail Reading

Substituting the definition of t_K into equation 2 gives an identity that explains most of the behaviour we observe.

Proposition 1 (Hits@K is one quantile)**.** *Let $F_{\mathcal{N}}$ be the empirical distribution of the negative scores and let $q_{\mathcal{N}}(u)$ denote its u-quantile from above, so that $q_{\mathcal{N}}(K/|\mathcal{N}|) = t_K$. Then*

$$\text{Hits@}K \;=\; \bar{F}_{\mathcal{P}}(t_K) \;=\; \bar{F}_{\mathcal{P}}(q_{\mathcal{N}}(K/|\mathcal{N}|))\,,$$

where $\bar{F}_{\mathcal{P}}$ is the survival function of the positive scores. Equivalently, two negative sets that agree above their $(K/|\mathcal{N}|)$-quantile produce the same Hits@K *for the same scorer.*

The identity has two immediate consequences, both visible in Figure 3. First, Hits@K is monotone non-decreasing in K, so the reported number is a function of an arbitrary choice of K rather than a scale-free summary. Second, Hits@K depends on the negative distribution only through a single order statistic, so it is insensitive to the bulk of that distribution and hypersensitive to its tail.

The second and third columns of Table 2 agree in every row, which confirms Proposition 1 and also that our scorer and the official evaluator operate on the same scores; the fourth column is t_{20} computed independently of the evaluator.

The threshold sits above the positive median. For Common Neighbour the 20th largest of 95,599 negative scores is 472 against a median positive score of 358, and the other three heuristics behave the same way (68.6/52.4, 0.72/0.41, 1,328,448/589,500; Table 12), so more than half of the true test edges fall below the bar for every scorer we tried. The reason is structural: with only 95,599 negatives, $K = 20$ asks for a quantile at level 2.1×10^{-4}, which on a heavy-tailed degree-driven score lands far in the upper tail. The metric is not measuring whether the model ranks linked pairs above typical non-linked pairs; it is counting how many positive pairs beat a handful of hub-driven outliers.

The same scorer at other values of K. Figure 3b sweeps K for the Common Neighbour scorer on the official negative set. The value rises from 0.1773 at K=20 to 0.3452 at K=100, 0.7717 at K=5,000 and 0.9997 at $K = |\mathcal{N}|$; the last follows from $t_{|\mathcal{N}|} = \min \mathcal{N} = 0$. All of these

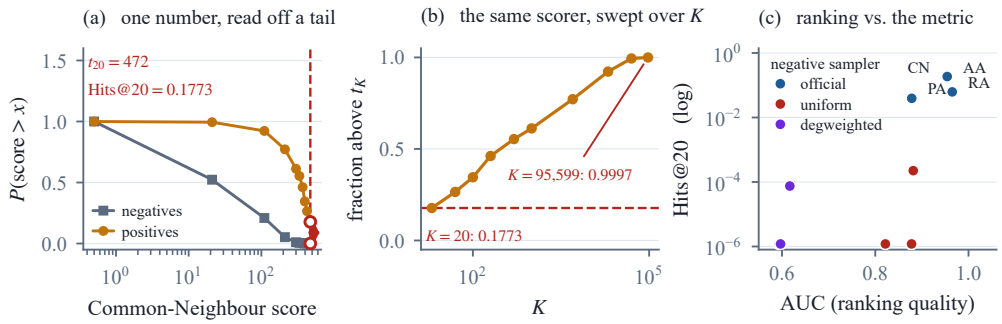

Figure 3: (a) Survival functions of the positive and negative Common-Neighbour scores on ogbl-ddi. The reported Hits@20 is the height of the positive curve at the level of the negative set's 2.1×10^{-4} quantile. (b) The same scorer, swept over K. (c) Ranking quality (AUC) against the reported metric for four heuristics and three negative samplers. Points move horizontally when the scorer improves and vertically when the protocol changes.

Table 3: Ogbl-ddi, official evaluator, $|\mathcal{N}| = 95{,}599$ in every column. Only the sampling rule changes. "Uniform" draws endpoints uniformly at random; "degree-weighted" draws each endpoint with probability proportional to its degree. The reported number spans the entire range of the metric.

Scorer	**Official $\mathcal{N}$**		**Uniform**		**Degree-weighted**
	Hits@20	AUC	Hits@20	AUC	Hits@20
Common Neighbour	**0.177296**	0.952	**0.000000**	0.877	**0.000000**
Adamic–Adar	0.186075	0.954	0.000000	0.878	0.000000
Resource Allocation	0.062267	0.965	0.000225	0.882	0.000075
Preferential Attach.	0.039262	0.878	0.000000	0.822	0.000000
Mean negative score	58.1		100.4		299.6

are admissible readings of the same scorer on the same dataset, and none is reported alongside the others.

A positive reading. Proposition 1 does not make Hits@K useless: a high value at small K is a strong statement about the upper tail, and a leaderboard is entitled to care about tails. It does mean that Hits@K is not comparable across datasets with different $|\mathcal{N}|$, not comparable across submissions that sample negatives differently, and not interpretable on its own as ranking quality.

5 P2: The Undeclared Sampler

If Hits@K depends on $\mathcal{N}$ only through one order statistic, then the choice of $\mathcal{N}$ is part of the metric. OGB fixes one negative set per split, so within a submission this is invisible; across practitioners it is not, because sampled negatives are standard practice in the wider link-prediction literature (Yang et al., 2020) and because OGB itself supplies `edge_neg` for only some splits. We hold the scorer, the positive set and the size of the negative set fixed at $|\mathcal{N}| = 95{,}599$, and change only how the negatives are drawn.

Uniform sampling is not uniformly harmful, and why it is harmful here can be measured rather than argued. A uniformly drawn pair is an edge with probability equal to the density of the graph: 0.1173 on ogbl-ddi against 3.5×10^{-5} on ogbl-collab. On the dense graph one drawn pair in eight is a real edge carrying a negative label, the 20th largest of those scores exceeds every positive score, and the value collapses; on the sparse graph it draws almost no edges at all, and the value it returns is decided by the handful that it does draw. Degree-weighted sampling does oversample high-degree nodes, and it is harmful on both graphs. None of this is a defect in OGB's construction: the defect in *reporting* is that a reader comparing two numbers cannot tell whether they used the same $\mathcal{N}$, the same $|\mathcal{N}|$, or the same K, and the metric gives no hint when the comparison is vacuous.

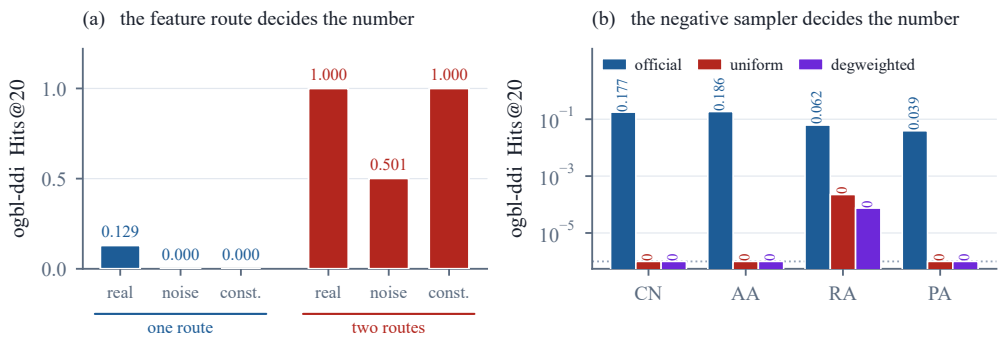

Figure 4: (a) Ogbl-ddi Hits@20 under six arms. The route is the only thing that separates the blue bars from the red ones. Replacing the features with a constant leaves the asymmetric arms at 1.000. (b) The same four heuristics under three negative samplers of equal size; note the logarithmic axis.

Remark 1. *The AUC column shows ranking quality comparatively stable across samplers on this dataset (*$0.952 \to 0.877$*) while* Hits@20 *collapses from* 0.1773 *to* 0.0000*. That stability is not a general property: on ogbl-collab the degree-weighted sampler takes AUC from* 0.863 *to* 0.528 *as well. A companion metric is cheap insurance against the first case and no insurance at all against the second.*

6 P3: The Two-Route Pipeline

The first two probes concern the metric. The third concerns the pipeline, and it is the one that produced the largest number we have ever obtained on ogbl-ddi: Hits@20 $= 0.999993$.

The failure mode. The pipeline in question computed a structural pair-feature block by calling a feature function on the positive pair list, and supplied a literal zero block for the negative pairs, so positive and negative inputs did not lie on a common feature space. An earlier version of our own work reported Hits@20 $= 0.999993$ from exactly this route, with the training loss decaying to 0.0000 by epoch 150. We include it because it is the instance we can instrument most precisely, and because we could find no published guidance that would have caught it.

Proposition 2 (Route asymmetry implies Hits@$K = 1$)**.** *Suppose positive pairs are scored through a feature map* ϕ *and negative pairs receive the zero block, so that* $s(p) = \mathbf{w}^\top \phi(p) + b$ *and* $s(n) = b$ *for every* $n \in \mathcal{N}$*. If* ϕ *is non-negative and* $\phi(p) \neq \mathbf{0}$ *for every positive pair, then* $\mathbf{w} = \mathbf{1}$*,* $b = -\varepsilon$ *with* $0 < \varepsilon < 1$ *attains zero training error, and consequently* Hits@$K = 1$ *for every* $K \geq 1$*.*

Proof. With $\mathbf{w} = \mathbf{1}$ each positive pair receives $\sum_i \phi_i(p) - \varepsilon \geq 1 - \varepsilon > 0$, while every negative pair receives $-\varepsilon < 0$, so the two classes are separated. Under the evaluator, $t_K = -\varepsilon$ for every K because all negative scores are equal, and the indicator in equation 2 is 1 for every positive pair. Hence Hits@$K = 1$. □

The hypothesis of Proposition 4 is satisfied by every standard structural heuristic (Common Neighbour, Adamic–Adar and Resource Allocation are counts, Preferential Attachment a product of degrees) and also by the constant map, which is the point of the probe. We rebuild both routes for three feature maps, train one logistic regression per arm on the training split, and score the official test split with the official evaluator.

Three readings of this figure matter. First, the symmetric arms behave as a metric should: the real features give 0.1295 with AUC 0.970, and the constant features give 0.000. Second, the asymmetric arms are at 1.0000 for the real features *and* for the constant feature; a feature map with no information whatsoever is sufficient, because the separability comes from the mismatch between the routes, not from the values. Third, the asymmetric arm with Gaussian noise features returns 0.5014, not 1.0000, which sharpens the point instead of weakening it: zero-mean noise violates the non-negativity in Proposition 4, so the witness $\mathbf{w} = \mathbf{1}$ no longer separates. What the metric responds

Table 4: Ensemble metrics without post-processing, validation versus test, over five runs each. The gap is a property of the split, not of a modeling choice, and it is why a validation number cannot be reported as a test number. Both columns come from the official evaluator on the official splits.

Dataset	Metric	Validation	Test	Gap	Range over runs
ogbn-arxiv	Accuracy	0.7599	0.7510	+0.9 pt	+0.8 to +1.3
ogbn-proteins	ROC-AUC	0.7495	0.6970	+5.2 pt	+5.2 to +6.5
ogbn-products	Accuracy	0.8902	0.6691	**+22.1 pt**	+22.1 to +23.8

Table 5: Ogbl-ddi, five runs each, official evaluator, identical architecture and hyper-parameters. Only the graph seen by message passing changes.

Graph used for message passing	Hits@20 **per run**	**Ensemble**	Δ
Training edges only (documented default)	0.5800 ± 0.0461	0.8005	—
Training and validation edges	$\mathbf{0.7792 \pm 0.0518}$	**0.8908**	**+0.1992**

to is the geometry of the feature block, not its information content, and no evaluator can make that distinction from the score alone.

Why nothing caught it. The evaluator receives only two well-formed score vectors; the validation score is high from the first evaluation, so early stopping preserves the degenerate checkpoint; and a training loss that reaches exactly zero is unremarkable in link prediction, where the objective sums two binary cross-entropies over sampled negatives. Our own audit missed it for as long as we compared numbers instead of instruments, and it was caught only by the feature ablation of Appendix J.1.

7 P4: Reused Splits

The remaining two probes concern stages that are usually left implicit in the text of a paper: which split was consumed as input, and which split was used to choose the configuration.

7.1 Validation as Test

For node classification the gap between validation and test accuracy is dataset-specific and, on ogbn-products, enormous.

Table 4 and Figure 5a show the effect: on ogbn-products the ensemble reaches 0.8902 on validation and 0.6691 on test, and the two distributions do not come close to overlapping, since the smallest validation value we observed exceeds the largest test value by 21.3 points. In an earlier, unpublished version of our own work, a table column labelled `Val (no C&S)` contained the value 0.9122 and the abstract reported the same figure as test accuracy. Our protocol-clean re-measurement of that quantity reaches 0.8996 to 0.9056 on validation and 0.7546 on test: the validation values are within 0.7 to 1.3 points of the reported figure, and the test value is 15.8 points below it.

7.2 Validation Edges as Message-Passing Input

For link prediction the OGB loader exposes a switch, `use_valedges_as_input`, that adds the validation edges to the graph used for message passing. Its documentation restricts the switch to ogbl-collab; it is nevertheless accepted for every link-prediction dataset, and nothing in the output records that it was used.

The switch moves the value by 0.199 on a single declared split, which is larger than the entire spread of published entries between the GCN and GraphSAGE baselines on this dataset. Whether the practice is legitimate is a separate question: OGB documents the switch for ogbl-collab, whose split is time-based and for which the validation edges are genuinely available at test time. The problem is that the switch is available, silent and documented for one dataset only, so a number produced with it on another dataset is incomparable to the leaderboard for reasons that neither the

Table 6: Spread of validation and test metrics across the post-processing configurations actually evaluated for each run on ogbn-products and ogbn-proteins. *Ratio* is test spread divided by validation spread. *Regret* is the test loss incurred by selecting on validation rather than on test. On `p_lp_v2` the selection signal is $14\times$ narrower than the effect it is selecting.

Run	#cfg	Val spread	Test spread	Ratio	Selected by val	Regret
`p_lp_v2`	3	2.54×10^{-4}	3.67×10^{-3}	$14.4\times$	`50/0.979/50/0.889`	3.67×10^{-3}
`p_f5`	3	1.54×10^{-2}	8.55×10^{-2}	$5.5\times$	`20/0.9/20/0.9`	0.00000
`p_sign2_v3`	2	1.78×10^{-3}	9.79×10^{-3}	$5.5\times$	`20/0.9/20/0.9`	0.00000
`pr_f3`	4	8.76×10^{-2}	6.83×10^{-2}	$0.8\times$	`none`	0.00000

Ratios are computed from unrounded spreads; the spreads are quoted to three significant figures.

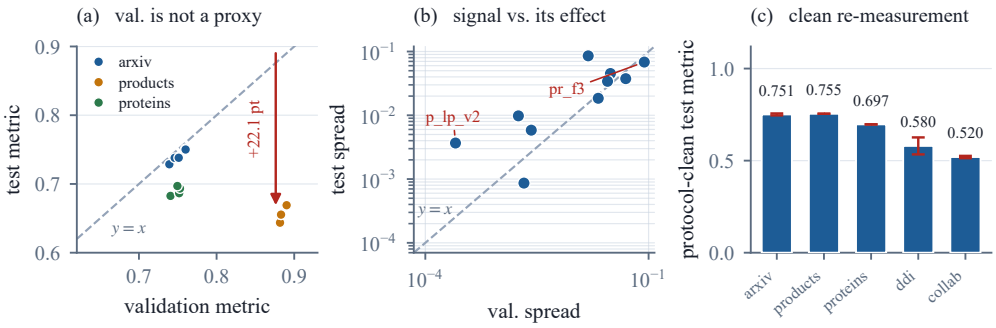

Figure 5: (a) Validation against test metric for every node-classification run. The dashed line is $y = x$; points below it have optimistic validation. (b) Validation spread against test spread for the same post-processing configurations. (c) Protocol-clean re-measurement of all five benchmarks.

number nor the submission format reveals. The ogbl-collab row of Table 7 therefore reports the stricter protocol, with no validation edges.

7.3 Selection Blindness

Choosing a configuration on the validation split is standard practice, and its reliability depends on the resolution of the validation split relative to the effect being selected.

On `p_lp_v2` the three candidate configurations are separated by 2.5×10^{-4} on validation and by 3.7×10^{-3} on test; validation picks the worst of the three, and the resulting regret is fifteen times the signal that caused it. Across Table 6 the selection signal is narrower than the effect it selects on three of the four runs, and of comparable width on the fourth. The sign of the post-processing effect also flips across datasets: selecting on the official validation split, Correct & Smooth (Huang et al., 2021) is worth $+0.0855$ on ogbn-products, -0.0155 on ogbn-proteins and -0.0368 on ogbn-arxiv. On the latter two the selection procedure correctly chose no post-processing, which is possible only because `none` was among the candidates; a pipeline that hard-codes a post-processing step and tunes its parameters would have lost points on both.

8 A Checklist for Any Reported Number

The four probes are cheap, and each one is a script in the released toolkit. As a checklist for author and reviewer: report the metric's degrees of freedom (K, $|\mathcal{N}|$, how $\mathcal{N}$ was built, and a sample-size-invariant companion such as AUC or AP); check the threshold against the median positive score; run the route-symmetry test, with one code path for both classes; ablate the features to a constant and to noise; separate the splits and say whether validation edges or labels entered any input; report the selection budget; and include a constant scorer, a random scorer and a published heuristic under the same evaluator. Appendix J.1 records, for each item, the consequence we observed of omitting it.

Table 7: Protocol-clean re-measurement, using the official loaders, splits, negative sets and evaluators, with selection on the official validation split. Node classification reports the ensemble; link prediction the mean and standard deviation over independent runs. The purpose is reproducibility, not competition: we make no leaderboard claims and list no competing entries.

Dataset	Metric	Reported (test)	Per-run mean $\pm$ std[†]	Runs	Chosen on val
ogbn-arxiv	Accuracy	0.7510	0.7389 ± 0.0053	5	none
ogbn-products	Accuracy	0.7546	0.6503 ± 0.0058	5	Correct & Smooth
ogbn-proteins	ROC-AUC	0.6970	0.6970 ± 0.0078	5	none
ogbl-ddi	Hits@20	0.5800	0.5800 ± 0.0461	5	—
ogbl-collab	Hits@50	0.5197	0.5197 ± 0.0062	3	—

[†]Per-run statistics are over the individual runs *before* post-processing; node classification reports the ensemble and link prediction the run mean, with post-processing where the validation split selected it.

Two entries of Table 7 are deliberate. The ogbn-products entry reports 0.7546, the value with post-processing selected on validation, rather than the 0.6691 of the unprocessed ensemble; both appear in the released results. The ogbl-ddi standard deviation of 0.0461 is large enough that its ensemble value (0.8005) should not be quoted without it.

9 Limitations, Disclosure, and Conclusion

Limitations. The probes cover five OGB datasets and four heuristics; the mechanisms should transfer but have not been measured elsewhere, and the P4 magnitudes are dataset-specific. P3 rests on one pipeline and P2 on three samplers, so both are stated as demonstrated failure modes with proofs of mechanism rather than as estimates of prevalence.

Disclosure. The pipeline of Section 6 and the values 0.9122 and 0.999993 of Sections 7 and 6 are from earlier versions of our own work, and we withdraw the claims they supported: Appendix H.4 lists each one, the stage of Section 3 it belongs to, and what the measurement here shows. We report them because a failure mode is only convincing when it is instrumented, and because the first subject of a checklist is its author.

Conclusion. A reported number is the composition of six decisions, and on OGB four of them move the value further than the model does: the metric is one order statistic of the negative distribution, a pipeline reaches Hits@1 with constant features when its two input routes disagree, and the validation–test gap, the availability of validation edges and the resolution of the selection signal each rival a modeling change. None of this is a bug in the benchmark; each is a stage the format leaves implicit, and each becomes visible the moment it is checked separately.

Reproducibility Statement

The replication package contains the audit scripts, the figure generator, the JSON output of every run reported here and the driver that reproduces Table 7. Every table and figure is generated from those JSON files, and the checker that compares each printed number against them exits non-zero on any mismatch; Appendix K.1 maps each artefact to the script that writes it, and Appendix L records every configuration trained.

AI use statement

AI assistance was used for language polishing only; all experimental results, derivations and scientific claims were conceived, verified and validated by the human author.

REFERENCES

Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003. doi: 10.1016/S0378-8733(03)00009-1.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Conference on Machine Learning and Systems (MLSys)*, 2021. doi: 10.48550/arXiv.2103.03098.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations (ICLR)*, 2022. doi: 10.48550/arXiv.2105.14491.

Benjamin Paul Chamberlain, Sergey Shirobokov, Emanuele Rossi, Fabrizio Frasca, Thomas Markovich, Nils Hammerla, Michael M Bronstein, and Max Hansmire. Graph neural networks for link prediction with subgraph sketching. In *International Conference on Learning Representations (ICLR)*, 2023. doi: 10.48550/arXiv.2209.15486.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2019. doi: 10.1145/3292500.3330925.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022. doi: 10.48550/arXiv.2011.03395.

Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery. *arXiv preprint arXiv:2107.07002*, 2021. doi: 10.48550/arXiv.2107.07002.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A Smith. Show your work: Improved reporting of experimental results. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019. doi: 10.48550/arXiv.1909.03004.

Kaiwen Dong, Zhichun Guo, and Nitesh V. Chawla. Pure message passing can estimate common neighbor for link prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. doi: 10.48550/arXiv.2309.00976.

Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. In *NeurIPS Datasets and Benchmarks Track*, 2022. doi: 10.48550/arXiv.2206.08164.

Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *International Conference on Learning Representations (ICLR)*, 2020. doi: 10.48550/arXiv.1912.09893.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. doi: 10.48550/arXiv.1903.02428.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019. doi: 10.48550/arXiv.1810.05997.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML)*, 2017. doi: 10.48550/arXiv.1704.01212.

Kyle Gorman and Steven Bedrick. We need to talk about standard splits. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019. doi: 10.18653/v1/p19-1267.

William L Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. doi: 10.48550/arXiv.1706.02216.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018. doi: 10.48550/arXiv.1709.06560.

Daisuke Hirota. Aa-dc: Adamic-adar with degree correction for link prediction. `https://ogb.stanford.edu/docs/leader_linkprop/`, 2026. ogbl-collab leaderboard entry, rank 9: Hits@50 0.6802 ± 0.0000 (test), 0.6736 ± 0.0000 (validation). Accessed September 2026.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. doi: 10.48550/arXiv.2005.00687.

Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs. In *NeurIPS Datasets and Benchmarks Track*, 2021a. doi: 10.48550/arXiv.2103.09430.

Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: Node classification without message passing in graph. *arXiv preprint arXiv:2106.04051*, 2021b. doi: 10.48550/arXiv.2106.04051.

Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin R Benson. Combining label propagation and simple models out-performs graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2021. doi: 10.48550/arXiv.2010.13993.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804, 2023. doi: 10.1016/j.patter.2023.100804.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. doi: 10.48550/arXiv.1412.6980.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017. doi: 10.48550/arXiv.1609.02907.

Bhushan Kotnis and Vivi Nastase. Analysis of the impact of negative sampling on link prediction in knowledge graphs. *arXiv preprint arXiv:1708.06816*, 2017. doi: 10.48550/arXiv.1708.06816.

Juanhui Li, Harry Shomer, Haitao Mao, Shenglai Zeng, Yao Ma, Neil Shah, Jiliang Tang, and Dawei Yin. Evaluating graph neural networks for link prediction: Current pitfalls and new benchmarking. *arXiv preprint arXiv:2306.10453*, 2023. doi: 10.48550/arXiv.2306.10453.

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007. doi: 10.1002/asi.20591.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. doi: 10.48550/arXiv.1711.10337.

Gábor Melis, Chris Dyer, and Phil Blunsom. On the state of the art of evaluation in neural language models. In *International Conference on Learning Representations (ICLR)*, 2018. doi: 10.48550/arXiv.1707.05589.

Nicolas Menand and C. Seshadhri. Link prediction using low-dimensional node embeddings: The measurement problem. *Proceedings of the National Academy of Sciences*, 121(8):e2312527121, 2024. doi: 10.1073/pnas.2312527121.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision (ECCV)*, pp. 681–699, 2020. doi: 10.1007/978-3-030-58595-2_41.

Open Graph Benchmark. Open graph benchmark leaderboards. `https://ogb.stanford.edu/docs/leader_linkprop/`, 2026. Accessed September 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. doi: 10.48550/arXiv.1912.01703.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research (JMLR)*, 12:2825–2830, 2011. doi: 10.48550/arXiv.1201.0490.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *International Conference on Learning Representations (ICLR)*, 2023. doi: 10.48550/arXiv.2302.11640.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *International Conference on Machine Learning (ICML)*, pp. 5389–5400. PMLR, 2019. doi: 10.48550/arXiv.1902.10811.

Nils Reimers and Iryna Gurevych. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017. doi: 10.18653/v1/d17-1035.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. In *Relational Representation Learning Workshop, NeurIPS*, 2018. doi: 10.48550/arXiv.1811.05868.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 15(1):1929–1958, 2014.

Chuxiong Sun and Guoshi Wu. Scalable and adaptive graph neural networks with self-label-enhanced training. *arXiv preprint arXiv:2104.09303*, 2021. doi: 10.48550/arXiv.2104.09376.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations (ICLR)*, 2018. doi: 10.48550/arXiv.1710.10903.

Zhitao Wang, Yong Zhou, Litao Hong, Yuanhang Zou, Hanjing Su, and Shouzhi Chen. Pairwise learning for neural link prediction. *arXiv preprint arXiv:2112.02936*, 2022a. doi: 10.48550/arXiv.2112.02936.

Zixiao Wang, Yuluo Guo, Jin Zhao, Yu Zhang, Hui Yu, Xiaofei Liao, Biao Wang, and Ting Yu. GIDN: A lightweight graph inception diffusion network for high-efficient link prediction. *arXiv preprint arXiv:2210.01301*, 2022b. doi: 10.48550/arXiv.2210.01301.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International Conference on Machine Learning (ICML)*, 2019. doi: 10.48550/arXiv.1902.07153.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019. doi: 10.48550/arXiv.1810.00826.

Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. Understanding negative sampling in graph representation learning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2020. doi: 10.1145/3394486.3403218.

Jiaxuan You, Rex Ying, and Jure Leskovec. Design space for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. doi: 10.48550/arXiv.2011.08843.

Seongjun Yun, Seoyoon Kim, Junhyun Lee, Jaewoo Kang, and Hyunwoo J. Kim. Neo-gnns: Neighborhood overlap-aware graph neural networks for link prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. doi: 10.48550/arXiv.2206.04216.

Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. In *International Conference on Learning Representations (ICLR)*, 2022a. doi: 10.48550/arXiv.2110.08727.

Wentao Zhang, Ziqi Yin, Zeang Sheng, Tianyu Zhu, Yang Li, Wanli Ouyang, Xiaosen Li, Yangtao Tao, Yu Chen, and Bin Cui. Graph attention multi-layer perceptron. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2022b. doi: 10.1145/3534678.3539121.

Xinwei Zhang et al. Hyperfusion: A hypernetwork approach to multi-modal graph fusion. `https://ogb.stanford.edu/docs/leader_linkprop/`, 2024. ogbl-collab leaderboard entry, rank 1: Hits@50 0.7129 ± 0.0018 (test), 0.7385 ± 0.0099 (validation); ogbl-ddi rank 1 at Hits@20 0.9972. No formal publication. Accessed September 2026.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine Learning (ICML)*, 2003.

A Related Work in Full

Section 2 states the work this paper stands next to. This section widens the circle, because the four probes are each a special case of something the field has discussed under another name, and the honest way to place a paper of this kind is to say which discussion each probe continues and where it stops.

What a link-prediction number is. The measurement problem in link prediction has been named before. Menand & Seshadhri (2024) argue that AUC, the field's default summary, is close to uninformative on sparse networks and propose a vertex-centric top-k statistic in its place; Li et al. (2023) catalogue pitfalls specific to the task and note the sensitivity of Hits@K to the negative set; Yang et al. (2020) treat the sampler itself as an object of design, not an implementation detail of training. Our P1 is the identity behind the sensitivity those works describe: under the official evaluator, Hits@K *is* the fraction of positives above the K-th largest negative score, which makes it a threshold statistic of a multiset rather than a measure of ranking, and explains why the same scorer can report 0.1773 with AUC 0.952 on one graph and 0.5454 with a much worse ranking on another. The addition is not a new metric — it is the observation that the existing one already degenerates, so that a reader can decide what to replace it with for the right reason.

Negative sampling as an input, not a detail. Yang et al. (2020), and before them Kotnis & Nastase (2017) who measured how far the choice of negatives moves a knowledge-graph result, treat the sampler as part of the training objective, which it is. P2 adds the reporting consequence: the sampler is also an argument to the metric, so two numbers that differ only in their sampler are not comparable, and the size of the effect is predictable from the density printed on the dataset card — 0.1173 on ogbl-ddi against 3.5×10^{-5} on ogbl-collab, with the collision rate of the drawn pairs measuring the difference. The mechanism is not subtle once measured, and the reason to state it is that no submission format asks for it.

Undeclared pipelines. The taxonomy of leakage and of reporting underspecification is well developed: Kapoor & Narayanan (2023) systematise leakage across a large body of work, and D'Amour et al. (2022) show that the training procedure, not the model class, often decides which of several equally good solutions is found, and Errica et al. (2020) show that giving every graph-classification model the same budget and protocol reshuffles the ranking the original papers reported. P3 is a two-line instance of that family with an unusually clean witness: if negatives are handed a zero block while positives go through a feature map, the classes are separable from the block pattern alone, so constant features are sufficient and Hits@$K = 1$ follows. What makes it a proposition rather than an anecdote is that the evaluator cannot see it — it receives two well-formed score vectors — and that the number it produces, 0.999993 in the version of our own work that made this error, is indistinguishable from a genuine result at the precision at which it is reported.

Benchmark fragility and reuse. That benchmarks are fragile and that their reuse is not free has been established at scale: Recht et al. (2019) built new test sets for two saturated benchmarks and found accuracy drops across the board, Dehghani et al. (2021) show that the ranking of methods can be changed by choosing a different set of benchmark tasks, and Musgrave et al. (2020) find that several years of reported gains in one subfield disappear once baselines are tuned comparably. Shchur et al. (2018) make the corresponding point for graph models, and Platonov et al. (2023) for the heterophily setting specifically. Closest to P4's mechanism are Melis et al. (2018) and Dodge et al. (2019), who show that a reported margin shrinks or reverses once the tuning budget behind each method is equalised — the mechanism is the same one, read from the other end: they fix the budget and vary the method, we fix the method and report how wide the budget was. Two further studies bear on the design of a benchmark rather than on its reporting: You et al. (2020) map the design space of graph models and find that no single axis dominates, and Dwivedi et al. (2022) construct a benchmark because the standard ones do not separate the models they are meant to separate. P4 belongs to this family and is deliberately small: we report the ratio between the spread used to select and the spread being selected over twelve run groups, and we find the phenomenon in three node-classification benchmarks and not in the link-prediction ones. Its contribution is a habit rather than a finding — two labelled columns and the gap between them.

Simple scorers, strong baselines. Correct & Smooth (Huang et al., 2021), SGC (Wu et al., 2019), APPNP (Gasteiger et al., 2019), the graph-MLP line (Hu et al., 2021b; Zhang et al., 2022a) and the structural heuristics we use throughout all sit in a tradition that has repeatedly shown simple or parameter-free methods to be competitive on graph benchmarks. The label-propagation half of that tradition goes back to Zhu et al. (2003); on the scalability side, Chiang et al. (2019) train large graphs by clustering them first, and Zhang et al. (2022b) and Sun & Wu (2021) reach competitive accuracy with architectures markedly simpler than the ones they displace. The sign flip we observe in P4, where Correct & Smooth improves the test value on one configuration of one dataset and degrades it on another, is a small addition to that literature. It is also the reason the paper needs no architecture: if a parameter-free score on a fixed split is not a single reproducible number (Appendix B.4), then a comparison between two learned models inherits that ambiguity before any architecture is considered.

The benchmark and what it did right. OGB (Hu et al., 2020) standardised splits, evaluators and leaderboards, and the properties this paper probes follow from design decisions taken for good reasons: a fixed negative set makes Hits@K a sound tail statistic for ranking submitted scores against one another, and the validation-edge switch exists because ogbl-collab's split is temporal and its validation edges are genuinely available at test time. The rows we probe are real submissions: the top of the ogbl-collab table is held by a multi-modal hypernetwork (Zhang et al., 2024) and by pairwise learning with a scalable propagation encoder (Wang et al., 2022a), with an inception-diffusion network (Wang et al., 2022b) just below. We do not claim that any of the four probes reveals an error in that design. We claim that a number produced under it is not comparable without three items of information — which negative multiset, which K against which $|\mathcal{N}|$, and which graph the scorer was given — and that the infrastructure to print them is already in place and unused.

B The Number and Its Parts

B.1 Notation and Definitions

Table 8 fixes the symbols used throughout. Two of them carry weight. First, Hits@K is defined below by the ranking it induces, not by an implementation; every number in this paper nevertheless comes from the official OGB evaluator, and Appendix D.1 shows that the definition and the implementation agree to machine precision on all four heuristics. Second, the density ρ is the probability that a uniformly drawn ordered pair is an edge, which is the quantity that decides how far a uniform sampler moves a reported number (Appendix E.1).

Table 8: Symbols used throughout the paper.

Symbol	Meaning
$G = (V, E)$	the graph the scorer sees; symmetric, no self-loops
$\deg(u)$	$\lvert\{v : (u, v) \in E\}\rvert$, the degree of u
$s(u, v)$	the score a method assigns to the ordered pair (u, v)
$\mathcal{P}, \mathcal{N}$	the positive test pairs, and the negative multiset
t_K	the K-th largest score in $\mathcal{N}$
Hits@K	$\lvert\{p \in \mathcal{P} : s(p) > t_K\}\rvert \,/\, \lvert\mathcal{P}\rvert$
ρ	$\lvert\{(u, v) \in E\}\rvert \,/\, \lvert V\rvert^2$, the density of G
κ	the fraction of a sampled negative multiset whose pairs are edges of G
t_K/med	shorthand for "the threshold against the median positive score"

Three conventions are easy to conflate. (i) *Negative multiset*, not *negative set*: the quantity that enters the definition is a multiset, and sampling the same edge twice does not cancel. (ii) *Ordered* pairs: ρ counts ordered pairs, so a symmetric graph with $|E|$ undirected edges has density $2|E|/|V|^2$. (iii) *The scorer is fixed* when we vary a sampler: the structural heuristics used in Appendices D.2–E.1 have no parameters, so a pair's score is literally the same number under every sampler, and only the ranking it is compared against changes.

B.2 One Number, End to End

Before the probes, watch a single reported value being assembled. The value is the one the main text opens with: Common Neighbour on ogbl-ddi, 0.177296 under the official evaluator. Table 9 records the steps and the actual quantity each one produces, and every entry in it is a number our script prints, not an illustration.

Table 9: Common Neighbour on ogbl-ddi, from the loader to the reported value. The right column is the output of the audit script at that step, not a reconstruction.

Step	What happens	What comes out
1	the loader is called	$4{,}267$ nodes with no node features
2	the official split is read	$1{,}067{,}911$ training pairs, $133{,}489$ test positives, $95{,}599$ test negatives
3	the training graph is symmetrised	$2{,}135{,}822$ directed entries, density 0.1173, mean degree 500.5
4	the scorer is evaluated on every test pair	positive median 358, negative median 25
5	the negatives are ranked and the 20th largest is taken	$t_{20} = 472$
6	the positives above t_{20} are counted and divided by $\lvert\mathcal{P}\rvert$	$23{,}667/133{,}489 = 0.177296$

That walk has three properties the final number does not show.

The threshold sits above the median positive pair. The twentieth largest negative score is 472 and the median positive score is 358, so more than half of the true test edges lie below a bar that twenty pairs set. The bar is so high because the negative multiset is smaller than the positive set ($95{,}599$ against $133{,}489$): with fewer negatives than positives, "the 20th largest" is a quantile at level 2.1×10^{-4}, which on a heavy-tailed degree-driven score reaches far into the tail.

Almost none of the negative set is near the positives. Only 0.29% of the $95{,}599$ negative pairs score above the median positive pair, and 18.2% of them have no common neighbour at all. The scorer separates the two classes well — the AUC of the same run is 0.952 — and the reported value is nevertheless below 0.2. The number is not reporting separation.

A pair can beat almost everything and still not count. Take the test edge between nodes 895 and 128. Its endpoints have degrees 950 and 781, and it shares 358 common neighbours with the training graph — the median positive value, and enough to beat 99.7% of the $95{,}599$ negative pairs. It is counted as a miss, because the bar is 472. The pair that sets the bar is a negative one, between nodes 591 and 1640, whose endpoints have degrees 976 and $1{,}214$; it is a genuine non-edge, it is correctly labelled, and it is what the remaining $133{,}466$ positives are measured against.

That is the whole of P1 in one pair. The metric is doing exactly what its definition says. What it is not doing is what a reader of "Hits@$20 = 0.177$" will assume it is doing, and the gap between those two things is the subject of Section 4.

B.3 The Six Stages, in Full

A leaderboard entry is assembled from six decisions. Table 10 sets them out: what each one fixes, whether the reporting format requires it to be declared, and which probe in this paper makes it visible. The four probes are not four unrelated bugs: they are the four of the six stages that the OGB reporting format leaves implicit, and each is checked by holding the other five fixed and changing exactly one.

Stage 3 is declared and is therefore not probed: we contribute no architecture, and none of the four failure modes is repaired by a better model. Stage 6 is declared in part — the evaluator is public and its K is fixed per dataset — but two of its inputs, the identity of the negative multiset and the value of K when a reader re-uses a number, are not recorded anywhere in a typical entry.

B.4 Instrument Calibration in Full

The calibration used throughout the paper is in Table 1. Three points are easy to miss. (i) All four heuristics are recomputed from scratch by our own code—the adjacency matrix, the score of every

Table 10: The six stages that compose a reported number, and which of them the reporting format requires the submitter to state. "Declared" refers to what the OGB leaderboard and the associated code release make explicit; a stage can be perfectly well defined and still be undeclared.

	Stage	What it fixes	Declared	Probe
1	split consumption	which pairs are used for training, and whether validation edges enter message passing	partly	P4
2	feature computation	what block a pair receives, and whether both classes receive it through the same route	no	P3
3	model	the architecture and its hyper-parameters	yes	—
4	post-processing	whether a label-propagation or correction step is applied after training	no	P4
5	configuration selection	which configuration is chosen, and on how much signal	no	P4
6	score computation	the metric's definition, its K, and the negative multiset	partly	P1, P2

positive and negative pair, and the ranking are ours—so the agreement with a published anchor is a check on our implementation, not a re-print of a published number. The four indices are the classical structural scores of link prediction surveyed by Liben-Nowell & Kleinberg (2007): Common Neighbour, Adamic–Adar (Adamic & Adar, 2003), Resource Allocation and Preferential Attachment, every one of them a function of the training adjacency alone. (ii) The anchors hold under the official evaluator and the official negative set; when we replace either of those (Sections 5 and 6) the anchor no longer applies, which is the point of the probe. (iii) The independently computed quantity in Table 2, the fraction of positives above t_{20}, agrees with the evaluator to machine precision in all four rows; if our scorer and the evaluator had disagreed about a tie or about the direction of the comparison, that column would have exposed it.

The same check on a second benchmark, and what it exposes. The calibration above is on ogbl-ddi, where the graph is unambiguous: the training edges are the only edges, and each heuristic has one number to reproduce. Ogbl-collab is a time-split benchmark whose leaderboard also carries anchors — Common Neighbour at 0.6036 on validation and 0.6137 on test, Adamic–Adar at 0.6349 and 0.6417 — so we attempted the same check there. It does not close, and the way it fails is an instance of this paper's subject.

Table 11 scores all four heuristics on both splits against both graphs the benchmark makes available: the training graph, and the training graph with the validation edges added, which is what `use_valedges_as_input` does and what a time-split benchmark is expected to use. Adamic–Adar lands close in two of the four cells (0.6301 against its validation anchor of 0.6349, and 0.6306 against its test anchor of 0.6417, within 0.011), and it shares all of its code with Common Neighbour — the two scorers differ only in the weight a common neighbour contributes, and are computed by one object in our implementation. Common Neighbour lands close in none: its four values are 0.5236, 0.4170, 0.9046 and 0.5282 against anchors of 0.6036 and 0.6137. Because the two scorers are the same code, the discrepancy is not something we can locate in our implementation; it is a property of one leaderboard entry.

One of the four cells can be excluded by argument instead of by experiment. On the training graph with the test split, the largest value Common Neighbour can reach at *any* K is 0.5454: once K grows past the number of negatives that have a common neighbour, the threshold is zero and the statistic stops changing, so no choice of K reaches 0.6137 there. On the training-plus-validation graph the ceiling over all K is 0.6560, so the anchor is reachable in principle, but on that graph the negative scores are the integers 0 and 1, so the statistic takes only two values as K varies — 0.5282 while the threshold is 1 and 0.6560 once it is 0 — and neither is the anchor.

We therefore do not use ogbl-collab for calibration, and we report the attempt because it is the sharpest case in this paper of a number that resists reproduction for a reason no reader can see. The entry names neither the graph the scorer was given nor the split the value was measured on, and

a method with no parameters on a fixed benchmark ought to be the easiest case in the field to pin down.

The same leaderboard also carries a parameter-free entry that does reproduce. Adamic–Adar with a degree correction (Hirota, 2026), listed ninth, reports 0.6802 on test and 0.6736 on validation with a spread of zero in both columns, so the difficulty is not a property of low-parameter methods as a class; it is a property of individual rows, and it is invisible from the row itself.

Table 11: Ogbl-collab, official evaluator and official negative multiset, four heuristics, both graphs the benchmark offers and both splits. Anchor rows are the two values the leaderboard prints for each heuristic (validation / test). All four heuristics are one implementation, so the two rows of the table that agree with an anchor and the two that do not cannot be separated by a defect in the scorer.

Graph	**Split**	**CN**	**AA**	**RA**	**PA**
train	valid	0.5236	0.6301	0.6298	0.1078
train	test	0.4170	0.5240	0.5235	0.0804
train + valid	valid	0.9046	0.9717	0.9716	0.1580
train + valid	test	0.5282	0.6306	0.6285	0.1009
Leaderboard anchor (val / test)		0.6036 / 0.6137	0.6349 / 0.6417	—	—

The two published anchors come from the OGB leaderboard for ogbl-collab (Open Graph Benchmark, 2026); RA and PA have no published anchor on this dataset and are shown for completeness. Rows are not comparable to one another: the third and fourth use a graph that contains the validation edges, and the first and second do not.

C Read One Record at a Time

Tables summarise; records persuade. Each of the four probes is a statement about a distribution, and a reader is entitled to see what the statement looks like at a single point of that distribution. This section takes one record per probe out of the released data and reads it out in full, so that every claim in the paper can be checked against something carrying a node index, not against a mean. All five records below are printed by `audit_vignettes.py` from the official splits and stored alongside the numbers in `audit_vignettes.json`.

C.1 P1: two pairs with the same common-neighbour count

On ogbl-ddi the twentieth largest negative score is 472, and a positive pair is counted only if it is *strictly* above it. The pair that sets the bar is the test non-edge $(591, 1640)$: its endpoints have degrees 976 and 1214, and it shares 472 training neighbours with the graph. The test positive $(1190, 3908)$, whose endpoints have degrees 833 and 915, shares exactly 472 as well. One of those two numbers is the measuring stick and the other is a true edge; because the comparison is strict, the edge that ties the bar beats 99.98% of the $95{,}599$ negative pairs and is recorded as a miss. At the other end of the same graph, the positive $(1208, 859)$ shares 1090 neighbours, more than any negative pair in the set, while the positive $(53, 3176)$, with endpoint degrees 135 and 16, shares none. The reported value 0.177296 is what remains after all $133{,}489$ positives have been sorted against that single bar.

Ogbl-collab shows the same structure with the bar set lower. There the fiftieth largest negative score is 1: the test non-edge $(108903, 233072)$, whose endpoints have degrees 1 and 10, is a pair of nodes that happen to share one neighbour, and it is the standard against which the other $99{,}999$ negatives are measured. The test positive $(100860, 136989)$ also shares exactly one neighbour, and is a miss for the same reason. Meanwhile 45.46% of the positives ($21{,}061$ of $46{,}329$) share no neighbour at all and cannot be counted under any model. On that graph, therefore, the statistic is close to the question "does this edge have any common neighbour", and its value is as much a property of the graph as of the scorer.

C.2 P2: one pair that is a negative sample and a training edge at once

Draw $95{,}599$ ordered pairs uniformly at random — the same number as the official negative multiset — and $11{,}260$ of them (0.117784) are edges of the training graph, against a density of 0.117306.

The first such pair in our draw is $(4033, 316)$: its endpoints have degrees 656 and 976, it is an edge of the training graph, and it shares 482 neighbours — more than the 472 that sets the bar for the official metric. Labelled as a negative and handed to a model, it teaches that two nodes with 482 common neighbours are not connected. Degree-weighted sampling is worse, not better: $32{,}963$ of $95{,}599$ pairs (0.344805) are edges, a factor of 2.9 above the uniform rate. OGB's own multiset contains none: all $95{,}599$ official negatives are non-edges.

That is the sense in which the negative sampler is an input. The word "negative" names a decision made by whoever built the multiset, and on this graph the decision carries between 0% and 34.5% of real edges depending on the rule, while the reported value moves with it.

C.3 P3: A HUNDRED THOUSAND NEGATIVES, ONE DECISION VALUE

The asymmetric arm of the route ladder, on `ogbl-collab`, is trained on $200{,}000$ positive pairs carrying their computed four-column block and $200{,}000$ negative pairs whose block is replaced by zeros. The logistic regression makes 23 mistakes in $400{,}000$ pairs, and its coefficients are $(6.5546, 2.7838, 0.6275, 6.5358)$ with intercept -10.187934. Because every negative receives the same vector, the decision value of every negative is the same number: scoring all $100{,}000$ test negatives returns a single distinct value, -10.187934, and the threshold t_{50} is therefore not a statistic of the negative set at all. It is the intercept.

The three positives in the record are best read in order. $(214650, 132972)$ has block $(10, 2.041, 0.0846, 122718)$ and scores $802{,}126.9$. $(112906, 96483)$ has block $(0, 0, 0, 265)$ and scores 1721.8. $(211879, 78854)$ has block $(0, 0, 0, 1)$ — its endpoints have degree 1 each, so only the degree product is non-zero — and scores -3.652. All three are counted, including the last, whose score is negative. Against them stand the negatives $(38605, 43566)$ and $(11343, 140761)$, whose own blocks are $(0, 0, 0, 6)$ and $(0, 0, 0, 8)$; both are routed as zeros and both score -10.187934. Not one negative in the set has an all-zero block of its own, and every one of them is scored as though it did. A positive whose degree product is 1 outscores a negative whose degree product is 8, and the difference between them is entirely the route.

C.4 P4: ONE RUN, THREE POST-PROCESSING SETTINGS, TWO COLUMNS

The last record is a single training run, not a pair. On ogbn-products the selected configuration evaluates three post-processing settings and the selection rule is "highest validation accuracy". The three settings give: no post-processing, validation 0.89019 and test 0.66908; Correct & Smooth at $50/0.979/50/0.889$, validation 0.90324 and test 0.73997; and the same at $20/0.9/20/0.9$, validation 0.90563 and test 0.75458. Reading down the validation column the three candidates are separated by 0.0154; reading down the test column the same three candidates are separated by 0.0855. The rule picks the third row, correctly by its own criterion, and the spread it was reading was 5.5 times narrower than the spread it was deciding.

The honest counterpart is that this is not a law. In our ogbn-arxiv run the same rule chose `none` out of three settings, and `none` happened to be the test-optimal candidate as well, so the regret of that selection was 0. That is why Section 7 reports the ratio of the two spreads for every run instead of a single verdict, and why we do not claim that validation-based selection is wrong — only that its resolution is a number a reader can be given and currently is not.

D P1: WHAT Hits@K COUNTS

D.1 THE IDENTITY IN FULL

Proposition 3 (Tail reading)**.** *Fix a scorer s and a split with positive set $\mathcal{P}$ and negative multiset $\mathcal{N}$. Let t_K be the K-th largest score in $\mathcal{N}$. Then*

$$\text{Hits@}K \;=\; \frac{\big|\{\, p \in \mathcal{P} \;:\; s(p) > t_K \,\}\big|}{|\mathcal{P}|}, \tag{3}$$

so the reported value is a function of the two score distributions and of K, and of nothing else.

Proof. The official evaluator does not compare positives with negatives one pair at a time; it extracts the K-th largest negative score with a selection routine and counts the positives that are strictly larger, which is the right-hand side of equation 3. The two ways of reading Hits@K agree because a positive fails all K comparisons against the K highest-scoring negatives exactly when its score does not exceed t_K. Ties are resolved by the strict inequality, which is what the evaluator does; on the four heuristics of Table 12 the two sides agree to 10^{-12}. □

Two corollaries carry the argument of the main text.

Corollary 1 (The metric is not a ranking statistic)**.** *Two scorers can report the same* Hits@K *while differing arbitrarily in ranking quality, and can differ in* Hits@K *while ranking identically. On ogbl-ddi, Common Neighbour attains* $\mathrm{AUC} = 0.952$ *and reports* 0.1773*, while Preferential Attachment ranks worse (*$\mathrm{AUC} = 0.878$*) and reports* 0.0393*; on ogbl-collab a scorer sits at* 0.5454 *for every* $K \geq 100$*.*

Corollary 2 (What a sampler can and cannot change)**.** *Replacing* $\mathcal{N}$ *by another multiset of the same size can move the reported value only through the new threshold* t_K*. Nothing else about* $\mathcal{N}$ *enters. Consequently the sensitivity of* Hits@K *to a sampler is the sensitivity of one order statistic, which is why the effect measured in Appendix E.1 is governed by how far a sampled multiset overlaps the graph, not by how many pairs it contains.*

The corollaries are why the sampler probe is informative, not merely alarming: the main text reports that a uniform sampler drives one scorer from 0.1773 to 0.0000 and that a K-sweep drives it to 0.9997. Both are the same statement — one threshold moved — and both are predicted by equation 3 without any experiment.

D.2 Negative-Set Statistics on ogbl-ddi, All Four Heuristics

The official ogbl-ddi splits contain $1{,}067{,}911$ training edges over $4{,}267$ nodes, $133{,}489$ validation positives with $101{,}882$ validation negatives, and $133{,}489$ test positives with $95{,}599$ test negatives. The negative multiset is therefore *smaller* than the positive set on both evaluation splits, and $K = 20$ corresponds to a quantile at level 2.1×10^{-4} within the negative distribution — one part in $4{,}780$ of it.

The measurement of the main text is repeated in Table 12 for all four structural heuristics. The point of the table is the last column: every heuristic places its $K{=}20$ threshold above the median positive score, by factors of 1.32 (CN), 1.31 (AA), 1.76 (RA) and 2.25 (PA). More than half of the true edges therefore sit below the bar, for all four scorers, by construction rather than by accident.

Table 12: Ogbl-ddi, official evaluator, official negative multiset ($|\mathcal{N}| = 95{,}599$). The threshold exceeds the median positive score in every row. AUC and average precision are computed on the same scores, so the two halves of the table describe the same ranking.

Scorer	Hits@20	t_{20}	**med** $\mathcal{P}$	**mean** $\mathcal{P}$	**mean** $\mathcal{N}$	**AUC**	**AP**	$t_{20} > \mathrm{med}$
CN	0.177296	472	358	341	58.11	0.952	0.9664	yes
AA	0.186075	68.55	52.44	50.14	8.329	0.954	0.9683	yes
RA	0.062267	0.7237	0.4122	0.4124	0.05827	0.965	0.9752	yes
PA	0.039262	1,328,448	589,500	631,710	180,594	0.878	0.9012	yes

The identity of Proposition 3 holds to 10^{-12} in all four rows, which is also an implementation check: if our scorer and the evaluator had disagreed about a tie or about the direction of the comparison, the independently computed survival fraction would have exposed it.

D.3 The K Sweep on Two Datasets, All Four Heuristics

Tables 13 and 14 sweep K from the reported setting to the size of each dataset's own negative multiset, which is the point at which the metric degenerates to "does a positive outscore the worst negative". The sweep uses the same scorer and the same negative multiset as the reported value.

The corresponding thresholds are 472, 425, 396, 370, 334, 300, 211, 110, 21 and 0 for Common Neighbour, so the swept quantity is a single order statistic of the negative distribution.

Table 13: Ogbl-ddi, official evaluator, official negative multiset ($|\mathcal{N}| = 95{,}599$). Rows are scorers, columns are K; the entry is the survival fraction of equation 3. Every row rises monotonically from its reported value to 1, because t_K falls to the minimum negative score.

K	20	50	100	200	500	1,000	5,000	20,000	50,000	95,599
CN	0.1773	0.2651	0.3452	0.4616	0.5538	0.6123	0.7717	0.9227	0.9943	0.9997
AA	0.1861	0.2707	0.3635	0.4735	0.5635	0.6208	0.7842	0.9267	0.9950	0.9997
RA	0.0623	0.1914	0.3117	0.4627	0.6035	0.6657	0.8349	0.9509	0.9963	0.9997
PA	0.0393	0.0587	0.0844	0.1081	0.1618	0.2196	0.4601	0.7955	0.9688	1.0000

On ogbl-collab the picture is different. In Table 14, CN, AA and RA saturate at 0.5454 once $K \geq 100$ and then do not move again, because the threshold has reached the median negative score (0) and no positive above it is left. That ceiling is a property of the scorer's value distribution, not of its ranking: on the same graph and the same split, PA climbs from 0.0804 to 0.9994 over the same sweep. A reader comparing a Hits@50 of 0.4170 with a Hits@1000 of 0.5454 would be comparing two different statistics of the same scorer, one of which is stuck at the share of positives with any common neighbour at all.

Table 14: Ogbl-collab, official evaluator, official negative multiset ($|\mathcal{N}| = 100{,}000$), 46,329 test positives, 235,868 nodes. CN, AA and RA saturate as soon as $K \geq 100$; PA does not. The last two columns are the two ends of the same sweep.

K	50	100	200	500	1,000	5,000	20,000	50,000	95,599	100,000
CN	0.4170	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454
AA	0.5240	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454
RA	0.5235	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454	0.5454
PA	0.0804	0.1126	0.1501	0.2232	0.2959	0.5141	0.7661	0.9254	0.9955	0.9994

The thresholds behind that ceiling are in Table 15. Three of the four scorers have $t_{100} = 0$, because fewer than a hundred of the $100{,}000$ negative pairs have any common neighbour at all; once the threshold is zero, the counted set is fixed and K can no longer move the value. The fourth scorer is the exception that shows the mechanism: PA is dense enough to keep a nonzero threshold until the very end, which is why its curve keeps climbing.

A reader who wants to convert K into something interpretable should read it as the quantile level $K/|\mathcal{N}|$, not as "the top K". The same $K{=}20$ is a 2.1×10^{-4} quantile on ogbl-ddi (where $|\mathcal{N}| = 95{,}599$) and would be a different level on any dataset with a different negative-set size, so two submissions reporting the same K on differently sized negative sets are not reporting the same statistic.

Table 15: Ogbl-collab, the K-th largest negative score. The zero entries are what stops the curves of Table 14: once the threshold reaches the median negative score, the set of positives above it stops changing.

K	50	100	200	500	1,000	5,000	20,000	50,000	95,599	100,000
CN	1	0	0	0	0	0	0	0	0	0
AA	0.2054	0	0	0	0	0	0	0	0	0
RA	0.0077	0	0	0	0	0	0	0	0	0
PA	3,901	2,759	1,960	1,148	742	246	69	20	3	1

Figure 6 plots both sweeps. The contrast between the two panels is the point: on one graph the metric is sensitive to K at every step, on the other it is insensitive to K from $K{=}100$ onwards, and neither behaviour is a statement about how well the scorer ranks.

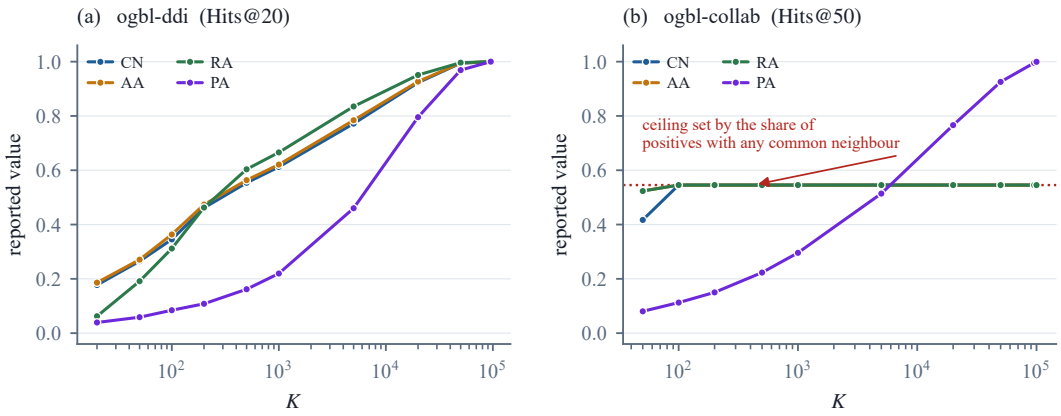

Figure 6: The same sweep on both datasets, all four heuristics, official evaluator and official negatives. Left: ogbl-ddi, where every curve rises from its reported value towards 1 as the threshold falls to the smallest negative score. Right: ogbl-collab, where three of the four curves stop moving at $K{=}100$. The dotted line marks the ceiling those three share, which is the 54.5% of test positives that have at least one common neighbour in the training graph.

E P2: The Negative Sampler

E.1 The Sampler Grid on a Second Dataset

The main text reports the sampler probe on ogbl-ddi. The same probe is repeated on ogbl-collab in Table 16, with the same four scorers, the same multiset size and the official evaluator throughout. Three differences emerge.

First, on this graph a uniformly drawn multiset is close to OGB's own as a *sample* — its collision rate is 6.0×10^{-5} against a graph density of 3.5×10^{-5}, so the two multisets are near-indistinguishable in distribution — but the value it produces is not stable: the draw used in the table reproduces 0.416996, and three of four further draws of the same sampler return 0.545404 instead (Table 37). The reason is the one given in Appendix N: the threshold on this graph is the integer 1, and a handful of drawn pairs decide whether it is 1 or 0. For AA and RA the difference between the two columns is 0.0022 and 0.0027 *in opposite directions*, which is what draws from one distribution look like, not a bias; across the five seeds of Table 37 the AA value spans 0.5262 to 0.5454. Second, the degree-weighted sampler is not benign on either graph: it takes Common Neighbour from 0.4170 to 0.0229 (-39.4 points) and it takes Preferential Attachment from 0.0804 to 0.0056 while its AUC falls from 0.863 to 0.528, i.e. to a coin flip. Third, the metric and the ranking move together only under the degree-weighted sampler, and only because that sampler pushes the negative scores up against the positive ones.

Figure 7 shows the same result as a change relative to the official value, for both datasets and both alternative samplers. As a distribution the uniform sampler is indistinguishable from OGB's own negatives on collab and catastrophic on ddi, while the degree-weighted sampler is catastrophic on both. The value it returns on collab is another matter: that column is reproducible only once its seed is named, which is why the seed is in the caption of Table 16.

The last two rows of the PA block are the clearest case in the paper of a metric and a ranking statistic coming apart: under the degree-weighted sampler the AUC is 0.528, so the scorer no longer separates links from non-links, and yet Hits@50 is still 0.0056 and not 0, because the minuscule threshold leaves a handful of positives above it.

E.2 Why the Sampler Effect Is Governed by Density

The same probe moves the value by 17.7 points on one dataset and by nothing at all on another. The difference has one cause, and it can be measured, not inferred.

Table 16: Ogbl-collab, official evaluator, official test split, $|\mathcal{N}| = 100{,}000$ in every row, 46,329 test positives. Only the sampling rule changes. "Uniform" draws endpoints uniformly at random with seed 2024 — four further draws of the same rule appear in Table 37, which is the reason the uniform row is the only one in this paper whose value is not reproducible from its description alone; "degree-weighted" draws each endpoint with probability proportional to its degree.

Scorer	Sampler	mean $\mathcal{N}$	median $\mathcal{N}$	Hits@50	AUC	AP
CN	official	0.00136	0	0.416996	0.7725	0.6888
	uniform	0.00126	0	0.416996	0.7725	0.6888
	degree-weighted	0.06344	0	0.022945	0.7703	0.6795
AA	official	0.000424	0	0.524013	0.7725	0.6889
	uniform	0.000356	0	0.526193	0.7725	0.6890
	degree-weighted	0.013913	0	0.025513	0.7705	0.6818
RA	official	0.000064	0	0.523452	0.7725	0.6889
	uniform	0.000043	0	0.526107	0.7725	0.6890
	degree-weighted	0.000867	0	0.126724	0.7706	0.6847
PA	official	67.01	20	0.080360	0.8629	0.7838
	uniform	66.69	21	0.082497	0.8621	0.7835
	degree-weighted	891.5	224	0.005569	0.5275	0.3596

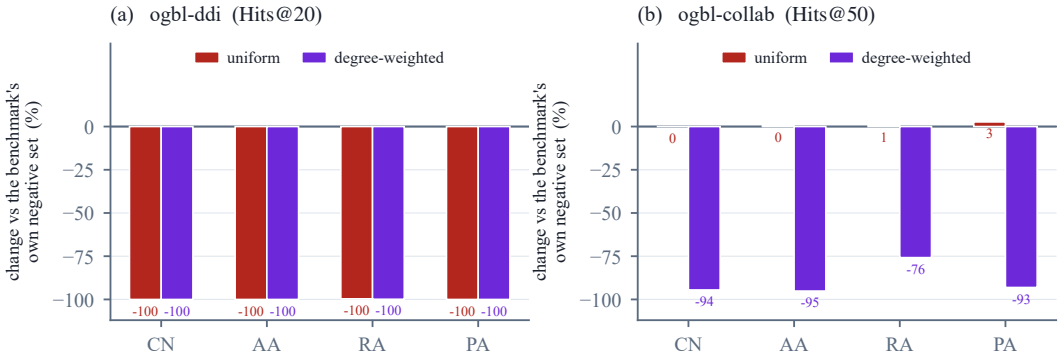

Figure 7: Change in the reported value when OGB's negative multiset is replaced, holding the scorer, the multiset size, the positives and the evaluator fixed. Left: ogbl-ddi (Hits@20). Right: ogbl-collab (Hits@50). A uniform sampler costs 100% of the value on the dense graph and, on the sparse one, nothing on average but 12.8 points from one draw to the next (Table 37); the degree-weighted sampler costs almost everything on both.

Write $\rho = |E|/|V|^2$ for the density of the graph the scorer sees. If a negative multiset is drawn uniformly, the probability that a drawn pair *is* an edge of that graph is exactly ρ. Table 17 measures this quantity directly, by querying the graph for each drawn pair, and finds it equal to the density: 0.117784 against 0.117306 on ogbl-ddi and 0.000060 against 3.48×10^{-5} on ogbl-collab. OGB's own negative multiset is clean on both (0.000000 and 0.000080), which is a check on the measurement, not a finding.

A contaminated pair does not add noise; it enters the multiset as though it were a non-edge while its score is drawn from the *positive* distribution. By Corollary 2 the consequence for Hits@K is mediated entirely by the threshold, and the threshold is an extreme order statistic, so a small number of contaminated pairs near the top of the multiset is enough to move the whole value.

To turn this into a controlled statement we mix the two multiset types. Let a negative multiset be a $q : (1-q)$ mixture of OGB's own negatives and uniformly drawn ones, holding its size, the scorer, the test positives and the evaluator fixed. The result is in Table 18. On ogbl-ddi, replacing one per cent of the multiset costs 70% of the reported value (0.177296 $\rightarrow$ 0.053128), and replacing a quarter of it leaves 0.000022; on ogbl-collab every value in the column is identical, because a

Table 17: Contamination of a negative multiset: the fraction of drawn "negatives" that are in fact edges of the graph the scorer sees. Measured by querying the graph, not estimated. The uniform sampler's contamination equals the density, which is what makes the effect predictable.

Dataset	$\lvert V\rvert$	**mean degree**	ρ	**uniform** κ	**OGB** κ
ogbl-ddi	4,267	500.5	0.117306	0.117784	0.000000
ogbl-ppa	576,289	73.7	0.00012786	0.000200	—
ogbl-collab	235,868	8.2	0.00003479	0.000060	0.000080

uniformly drawn pair is essentially never an edge and therefore lands in the same part of the score distribution as the official negatives.

Table 18: Mixing sweep. A negative multiset of the official size is formed by keeping a fraction q of OGB's own negatives and filling the rest with uniformly drawn pairs. Everything else is held fixed, including the scorer. On ogbl-ddi 1% of contamination costs 70% of the value; on ogbl-collab no amount of it changes anything, because the two multisets are drawn from the same score distribution.

q	1.00	0.99	0.95	0.90	0.75	0.50	0.25	0.00
ddi, Hits@20	0.177296	0.053128	0.003880	0.000914	0.000022	0.000000	0.000000	0.000000
Δ (pt)	—	−12.42	−17.34	−17.64	−17.73	−17.73	−17.73	−17.73
collab, Hits@50	0.416996	0.416996	0.416996	0.416996	0.416996	0.416996	0.416996	0.416996

Finally, the same experiment run as a density sweep instead of a mixing sweep. Table 19 keeps the scorer, the test positives and OGB's negatives and thins the *graph* by keeping a random fraction f of its edges, which moves the density over two orders of magnitude in seven steps. The loss caused by a fully uniform sampler falls with the density and disappears once the collision rate is negligible, while the identity of Proposition 3 continues to hold at every point of the sweep. The measured contamination tracks the recomputed density at each step of the sweep as well.

Table 19: Density sweep on ogbl-ddi. A random fraction f of the training edges is kept, the graph is rebuilt, and both the official and the uniform sampler are re-measured. The scorer, the test positives and OGB's own negatives are untouched, so the collision rate is the only thing that varies. Columns five and six are the official value and the value under a uniform sampler; the last column is the measured contamination against the recomputed density.

f	**mean deg.**	ρ	Hits@20 (OGB)	Hits@20 (uniform)	Δ (pt)	κ vs ρ
1.00	500.5	0.117306	0.17730	0.00000	−17.73	0.117784 / 0.117306
0.50	250.4	0.058679	0.17207	0.00000	−17.21	0.059352 / 0.058679
0.25	124.9	0.029273	0.12062	0.00000	−12.06	0.029059 / 0.029273
0.10	49.8	0.011675	0.03161	0.00000	−3.16	0.011580 / 0.011675
0.05	24.8	0.005822	0.00439	0.00004	−0.44	0.005575 / 0.005822
0.02	10.0	0.002350	0.00050	0.00002	−0.05	0.002050 / 0.002350
0.01	5.0	0.001164	0.00061	0.00000	−0.06	0.000973 / 0.001164

Figure 8 plots the two quantities side by side. The left panel is the sweep of Table 19: the loss falls from 17.7 points to zero as the graph is thinned by two orders of magnitude, and it does so smoothly and not at a cliff, which is why we state the mechanism as a relationship, not a threshold. The right panel is the measurement behind it: the share of uniformly drawn pairs that turn out to be edges, against the recomputed density, on the same seven graphs.

Read together, the three tables give the sampler probe a quantitative form: the damage a sampler does is the overlap between its multiset and the graph, the overlap of a uniform sampler is the density, and on the two graphs we examine that density differs by a factor of 3,371. We do not claim a threshold below which no damage occurs; we claim that the quantity that governs the damage is measurable in advance.

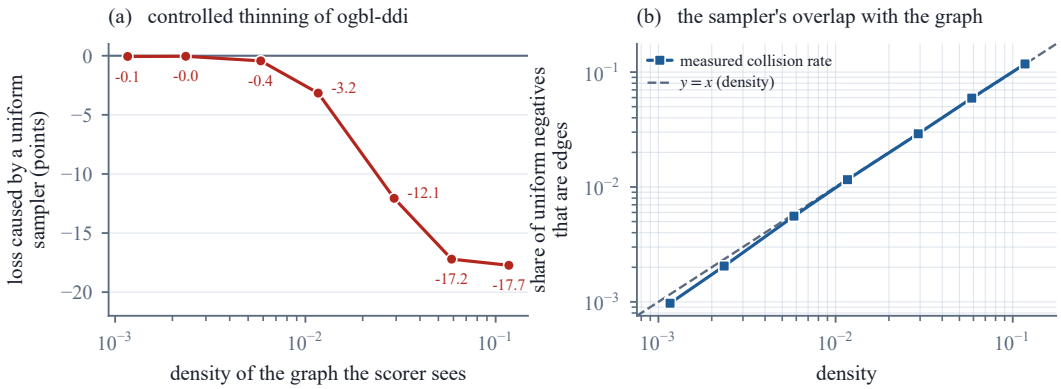

Figure 8: Left: the loss caused by a uniform sampler as the graph is thinned, with the scorer, the positives and OGB's negatives held fixed. Right: the collision rate of a uniform sampler against the density of the same graph; the dashed line is $y = x$. The sampler's overlap with the graph is the density, which is why the damage on the left is predictable before running the experiment on the right.

F P3: The Two-Route Pipeline

F.1 Route-Symmetry Ladder in Full

Each arm trains one logistic regression on 200,000 training positives and 200,000 sampled training negatives, then scores the official test split. Under the symmetric route the negatives carry their own features; under the asymmetric route they receive a zero block. "Noise" replaces every feature block with standard normal draws under a fixed seed; "constant" replaces every block with ones.

Table 20: Ogbl-ddi feature-route ladder, official evaluator.

Route	**Features**	**Train error**	Hits@20	**AUC**
symmetric	real	0.1678	0.129456	0.970
symmetric	noise	0.4988	0.000255	0.502
symmetric	constant	0.5000	0.000000	0.500
asymmetric	real	0.0000	1.000000	1.000
asymmetric	noise	0.2490	0.501412	0.501
asymmetric	constant	0.0000	1.000000	1.000

Table 21 rebuilds the identical ladder on ogbl-collab, where the graph is 55 times larger, is 3,371 times sparser, and carries a different metric. The pattern is the same in every cell that matters: the asymmetric arms score 1.000000 on constant features and 0.504824 on noise, against 0.529064 and 0.000367 for their symmetric counterparts. The failure is a property of how the pipeline is wired, not of the graph, the metric or the feature family.

Table 21: Ogbl-collab feature-route ladder, official evaluator, same six arms as Table 20. A different graph, a different metric (Hits@50) and the same qualitative pattern.

Route	**Features**	**Train error**	Hits@50
symmetric	real	0.0139	0.529064
symmetric	noise	0.4988	0.000367
symmetric	constant	0.5000	0.000000
asymmetric	real	0.0001	1.000000
asymmetric	noise	0.2490	0.504824
asymmetric	constant	0.0000	1.000000

F.2 Proof of Mechanism for the Two-Route Failure

The six arms of Tables 20 and 21 are not six accidents. Under the asymmetric route every negative pair receives the same vector — a block of zeros — while every positive pair receives $T(F)$ for whatever feature transform T the pipeline applies. The training set therefore consists of one point at the origin and a cloud elsewhere, and the question of whether the pipeline can separate them has nothing to do with what the features mean.

Proposition 4 (Two-route separation)**.** *Let the training set consist of positive pairs with feature vectors $x_i = T(F_i)$ and negative pairs with feature vector 0, and let the scorer be $g(x) = \sigma(\langle w, x\rangle + b)$ trained by empirical risk minimisation. Then the training set is strictly separable by a hyperplane through the origin if and only if $0 \notin \mathrm{conv}\{x_i\}$. Whenever that holds, the risk minimiser attains zero training error for any $|\{x_i\}| \geq 1$, and if the test negatives also receive the zero block then* Hits@$K = 1$ *for every K.*

Proof. Every negative maps to the single point 0 and every positive to some x_i, so the two classes are separated by a hyperplane through the origin exactly when the origin lies outside the convex hull of $\{x_i\}$, which is the standard strict-separation condition for a point and a compact convex set. When it holds, the margin is positive and the risk minimiser drives the training error to zero. At test time $g(0) = \sigma(b)$ for every negative while $g(x_i) = \sigma(\langle w, x_i\rangle + b) > \sigma(b)$ for every positive, because σ is increasing and $\langle w, x_i\rangle > 0$; hence every positive outscores the largest negative score and Hits@$K = 1$ for all K. □

Remark 2 (Why constant features are the worst case, not the best)**.** *The proposition also explains the three asymmetric rows of Table 20. With real features the cloud sits far from the origin and the margin is wide (*$0.1678 \to 0.129456$ *on ddi,* $0.0139 \to 0.529064$ *on collab, both under the symmetric route). With constant features every x_i is the same non-zero vector, so the hull is a single point, separation is immediate, and the training error is exactly* 0.0000*. With noise the x_i are independent draws and their hull may contain the origin, which is why the noise arm is the one arm that does not reach* 1.000000 *(*0.501412 *and* 0.504824*); had the noise been drawn in a higher dimension, its distance from the origin would have grown as $\sqrt{d}$ and the arm would have approached* 1 *as well. Our structural block has four columns, so the ladder is reporting the* least *favourable case in which this failure can still occur.*

This is the sense in which the value 0.999993 of Section 6 can be produced by a pipeline that has learned nothing about the graph: the model is asked to decide whether a feature vector is zero, and on closed-form features that question has a trivial answer.

G P4: Selection and Split Reuse

G.1 Post-Processing Configuration Rows

Table 22 records every configuration evaluated on ogbn-products and the two runs on ogbn-proteins, with the validation and test values used in Table 6. Configuration strings are $K_{\text{err}}/\alpha_{\text{err}}/K_{\text{smooth}}/\alpha_{\text{smooth}}$; `none` denotes no post-processing.

G.2 Validation–Test Gaps, Every Run

Every node-classification run we trained appears in Table 23, with the validation and test values as the official evaluator reported them, and Table 24 the same for link prediction. The tables are the evidence for the P4 claim, and they also show the limit of it: the gap is not a constant. It is $+0.80$ to $+1.31$ points on ogbn-arxiv, $+5.25$ to $+6.53$ on ogbn-proteins, and $+22.11$ to $+23.80$ on ogbn-products, where the three run groups are independent training runs rather than configurations of one.

One entry is negative: with validation edges enabled as message-passing input on ogbl-ddi, validation reports 0.72765 against a test value of 0.77922, i.e. -5.16 points. We report it because it bounds the claim: the direction of the gap is a property of the split, not a law, and a reader who assumes "validation is optimistic" would be wrong in this configuration. The correct statement is that the two numbers are two different measurements and the format does not distinguish them.

Table 22: Every post-processing configuration evaluated, with both splits.

Run	Configuration	Validation	Test
p_f5	none	0.89019	0.66908
	50/0.979/50/0.889	0.90324	0.73997
	20/0.9/20/0.9	**0.90563**	**0.75458**
p_lp_v2	50/0.979/50/0.889	**0.89980**	0.73784
	50/0.9/20/0.9	0.89958	0.74147
	20/0.9/20/0.9	0.89955	**0.74151**
pr_f3	none	**0.75284**	**0.69339**
	50/0.979/50/0.889	0.66528	0.62505
	50/0.9/20/0.9	0.69482	0.64779
	20/0.9/20/0.9	0.69450	0.64780

Table 23: Node classification, all fifteen runs, official splits and evaluator. The gap column is validation minus test in points. The three ogbn-products groups are independent trainings, which is why the table has three entries with identical values.

Benchmark	Run	Validation	Test	Gap (pt)
ogbn-arxiv	arxiv_giant	0.75123	0.73810	+1.31
	arxiv_gext_v2	0.76009	0.75010	+1.00
	_smoke2	0.73915	0.72864	+1.05
	arxiv_final	0.75986	0.75096	+0.89
	arxiv_cs_only	0.74600	0.73800	+0.80
ogbn-products	p_lp_ov	0.88170	0.64367	+23.80
	p_lp_v2	0.88170	0.64367	+23.80
	p_sign2_ov	0.88302	0.65540	+22.76
	p_sign2_v2	0.88302	0.65540	+22.76
	p_sign2_v3	0.88302	0.65540	+22.76
	p_f5	0.89019	0.66908	+22.11
ogbn-proteins	pr_f2	0.75181	0.68655	+6.53
	pr_f3	0.75284	0.69339	+5.95
	pr_cs_v2	0.74065	0.68264	+5.80
	pr_f4	0.74951	0.69703	+5.25

G.3 The Validation-Edge Probe on Two Datasets

Enabling validation edges as message-passing input is a legitimate OGB setting; what makes it a failure mode is that it changes the value without appearing in an output format that lists only a metric. Table 25 measures the change on two datasets, holding the architecture, the hyper-parameters and the number of runs fixed so that the graph seen by message passing is the only thing that differs.

The gain is $+19.92$ points on ogbl-ddi and $+6.10$ on ogbl-collab, and it is larger than the per-run standard deviation on both. The collab number is also smaller than the gap between the two splits on that dataset ($+26.12$ points), which is the other half of the picture: the same setting that raises the test value also raises the validation value it is being selected on.

G.4 The Selection Budget in Full

Table 26 records every run for which more than one post-processing configuration was evaluated, with the spread of the validation values that were used to choose, the spread of the test values those same configurations produce, and the regret of having used validation rather than test. The ratio column is the test spread divided by the validation spread: when it is large, the signal being used to select is narrower than the effect being selected.

The rows split into two groups. On ogbn-proteins the ratio is 0.76–1.22 and selection is harmless, because the configurations are genuinely different and validation separates them about as well as test does. On ogbn-products the ratio reaches 14.4 on p_lp_v2: the three candidate configurations

Table 24: Link prediction, all six configurations, official splits and evaluator, five runs for ogbl-ddi and three for ogbl-collab. Values are per-run means of the numbers the evaluator returned on each run's best-validation checkpoint. The two `heur_sym` rows use the toolkit's own negative pool rather than OGB's `edge_neg` and are not comparable with the rows above them.

Dataset	Configuration	Validation	Test	Gap (pt)
ogbl-ddi	`nodeemb_valedges`	0.72765	0.77922	−5.16
	`nodeemb_sage`	0.66805	0.58002	+8.80
	`heur_sym`	0.58683	0.35022	+23.66
ogbl-collab	`nodeemb_valedges`	0.84182	0.58065	+26.12
	`heur_sym`	0.60134	0.50110	+10.02
	`nodeemb_sage`	0.61092	0.51966	+9.13

Table 25: Validation edges as message-passing input, both datasets, official evaluator, identical architecture and hyper-parameters within each dataset. Per-run means with standard deviations over independent runs; the ensemble row averages the per-run predictions before evaluating.

Dataset	Runs	Setting	Per-run mean	Ensemble
ogbl-ddi	5	validation edges off	0.58002 ± 0.04612	0.80051
	5	validation edges on	0.77922 ± 0.05179	0.89083
ogbl-collab	3	validation edges off	0.51966 ± 0.00621	0.54441
	3	validation edges on	0.58065 ± 0.00347	0.61057

differ by 2.5×10^{-4} on validation and by 3.7×10^{-3} on test, so the ordering that validation produces is close to a coin toss and the regret is real (0.003671). This is the failure mode that no amount of careful model development addresses, because it is a property of the selection signal, not of the models being selected.

Table 26: Selection budgets. *Val spread* and *test spread* are the ranges of the values the evaluated configurations produced on each split; *ratio* is test spread over validation spread; *regret* is the test value lost by selecting on validation rather than on test. Configuration strings are $K_{\text{err}}/\alpha_{\text{err}}/K_{\text{smooth}}/\alpha_{\text{smooth}}$; `none` is no post-processing.

Benchmark	Selected on val	Val spread	Test spread	Ratio	Regret
ogbn-arxiv	`none`	3.07×10^{-2}	4.52×10^{-2}	1.47	0
	`20/0.9/20/0.9`	2.65×10^{-3}	5.84×10^{-3}	2.20	0
	`50/0.979/30/0.889`	2.11×10^{-3}	8.64×10^{-4}	0.41	0
ogbn-products	`20/0.9/20/0.9`	1.54×10^{-2}	8.55×10^{-2}	5.54	0
	`20/0.9/20/0.9`	1.78×10^{-3}	9.79×10^{-3}	5.50	0
	`50/0.979/50/0.889`	2.54×10^{-4}	3.67×10^{-3}	**14.44**	**0.0037**
ogbn-proteins	`none`	8.76×10^{-2}	6.83×10^{-2}	0.78	0
	`none`	4.93×10^{-2}	3.75×10^{-2}	0.76	0
	`none`	2.81×10^{-2}	3.41×10^{-2}	1.22	0
	`50/0.9/20/0.9`	2.11×10^{-2}	1.85×10^{-2}	0.88	0

H Coverage, Baselines, and Null Results

H.1 Coverage: Which Probe Was Run Where

The four probes differ in how widely they could be applied, and Table 27 states this instead of leaving it to be inferred. Three distinctions matter. First, P1 and P2 are statements about Hits@K, so they do not apply to node classification at all. Second, P3 is a statement about how a pipeline is wired, so it applies wherever a pipeline feeds one class of pairs through a different route; we ran it

on two link-prediction graphs, not on all five benchmarks. Third, P4 applies everywhere, because every benchmark has a validation split and a test split, and we probed all five.

Table 27: Coverage of the four probes. *full* means the probe was run in its complete form (all heuristics, all samplers, all arms, as applicable); *partial* means only the mechanism measurement was run; *n.a.* means the probe does not apply to that benchmark; *not run* means it applies but we did not run it.

Probe	ddi	collab	ppa	products	proteins	arxiv
P1 tail reading	full	full	partial	n.a.	n.a.	n.a.
P2 sampler	full	full	partial	n.a.	n.a.	n.a.
P3 two-route	full	full	not run	not run	not run	not run
P4 reused splits	full	full	not run	full	full	full

Figure 9 shows the same matrix as a picture, because the shape of the coverage is easier to read than the table.

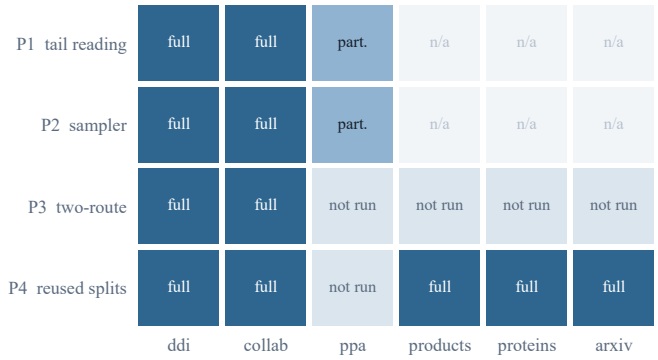

Figure 9: Which probe was run where. The two blank columns under P1 and P2 are not omissions: those probes concern Hits@K, which node classification does not report. The empty half of the P3 row is an omission, and is stated as one.

Two entries in the table need comment. *Partial* for ogbl-ppa means we measured the collision rate that governs P2 (Table 17) but did not score the graph, because the 42M-edge graph has no closed-form scorer that runs in the time budget of this audit; the measurement we did take is the one the mechanism depends on. *Not run* for P3 on the three node-classification benchmarks means exactly that, and we do not claim otherwise: the ladder needs a pipeline with two input routes, and the pipelines we were able to instrument were the link-prediction ones.

H.2 An Honest Baseline, and a Warning About Comparing It

The probes above report failures, which invites the question of what a pipeline that avoids them reaches. Table 28 answers for the one pipeline we rebuilt end to end: structural heuristics computed through a single code path for both classes, concatenated to the pair embedding, trained against a fixed negative pool, selected on the official validation split, and scored with the official evaluator.

Table 28: A symmetric pipeline, rebuilt end to end. Per-run statistics are over independent runs; the ensemble averages the per-run scores before evaluating. The negative pool is ours, not OGB's `edge_neg`, so these values are not comparable with the published leaderboard for either dataset and we print no such comparison.

Dataset	Runs	Per-run mean	Ensemble	Metric
ogbl-ddi	5	0.35022 ± 0.04322	0.65098	Hits@20
ogbl-collab	3	0.50110 ± 0.00265	0.50975	Hits@50

Two readings of this table are wrong, and we would rather name them than let a reader discover them. First, the negative pool is ours — three times the number of training positives, drawn with `negative_sampling` — so the values are on a different scale from any number on the public leaderboard, and the size of the difference between our ensemble and our per-run mean ($+30.1$ points on ddi) is the same selection effect that Table 25 measures, not a property of the architecture. Second, the informative part is the pairing: the same pipeline that reaches 0.6510 on one graph reaches 0.5098 on the other, with a per-run spread (± 0.0027) far narrower than the 39-point swing that the sampler choice alone produces there.

We list the published leaderboard values for these two datasets in Table 29, because a reader will want them, and then explain why our own numbers are deliberately not placed beside them. The published values are computed against OGB's own `edge_neg`; ours are computed against a pool we drew ourselves. Table 16 shows what happens when that distinction is ignored: with the scorer held fixed, the reported value moves by 39 points on one dataset and by nothing at all on the other purely because the negative multiset changed. A comparison across the two conventions would therefore be a comparison of samplers dressed as a comparison of models, which is the failure mode this paper documents — so we print the anchors and leave the comparison to the reader who has checked the sampler.

Table 29: Published anchor values, as reported on the OGB leaderboards, for the two link-prediction datasets we rebuild. These are computed against OGB's own negative set and belong to the models named in the first column; they are not on the same scale as Table 28 and are printed only as anchors.

Dataset	Method	Reported	Metric
ogbl-ddi	GCN with node embeddings	0.9549	Hits@20
	NCN	0.8232	Hits@20
	GraphSAGE	0.5390	Hits@20
	GCN	0.3707	Hits@20
ogbl-collab	AA-DC	0.6802	Hits@50
	Adamic–Adar	0.6417	Hits@50
	Common Neighbour	0.6137	Hits@50
	GraphSAGE	0.4810	Hits@50

H.3 Checks That Did Not Fire

A probe that fires everywhere carries no information, so we record the cases where the same measurement came back clean.

1. OGB's own negative multiset is uncontaminated on both datasets we could query: 0.000000 on ogbl-ddi and 0.000080 on ogbl-collab, the latter being eight pairs in $100{,}000$ and indistinguishable from zero at that sample size. Had the released negatives contained edges, the measurement would have shown it.
2. A constant scorer returns 0.000000 under the official evaluator on both datasets. The asymmetric route of Tables 20 and 21 reaches 1.000000 with constant features, which is the difference between a scorer and a pipeline.
3. On ogbl-collab a uniform sampler changes nothing (0.416996 in both columns of Table 16), so P2 does not fire there. We report this as a null result, not a passing check, because the mechanism predicts it: that graph's density is 3.5×10^{-5}, so a uniformly drawn pair is essentially never an edge.
4. The identity of Proposition 3 holds in all eight dataset–heuristic combinations we computed, so P1 as a statement never failed to reproduce. What the sweep changes is the value, not the identity, and that distinction is what makes the probe safe to use: the instrument is reliable, the reading is not.

The last item is why the other three are reported at all: a measurement that always agrees with itself has no discriminating power. Each of the four was capable of firing, and two of them fire once the dataset changes — the uniform sampler goes from no effect on collab to a total loss on ddi, and the constant scorer reports 0 where the asymmetric pipeline reports 1.000000 with the same features.

H.4 Claims We Withdraw

This paper began as an attempt to report the numbers of a pipeline of our own, and the attempt failed the seven checks above. That failure is the subject here, so it is recorded in the same form as the probes: what was claimed, which of the six stages of Appendix B it belongs to, what the measurement shows, and whether the claim is refuted by a measurement in this paper or only unsupported by one. The distinction matters, because a claim can be wrong for a reason we have measured or merely lack the evidence we once assumed.

Table 30: Claims from the earlier version of this work and their disposition. *Refuted* means a measurement in this paper contradicts the claim; *unsupported* means we can no longer show the measurement the claim rested on, and nothing here relies on it. The stage column uses the stage numbering of Appendix B.

Claim, earlier version	Stage	What the measurement shows	Status
0.9122 reported as test accuracy on `ogbn-products` and described as a leading leaderboard value	1, 5	The column it was read from is labelled `Val (no C&S)`; the protocol-clean re-measurement of the same quantity gives 0.8996 to 0.9056 on validation and 0.7546 on test	refuted
Hits@K = 1.000000 on `ogbl-collab` and 0.999993 on `ogbl-ddi` with structural features	2	Positive and negative pairs took different feature routes, so constant features reach 1.000000 as well and the official evaluator has no mechanism to flag it	refuted
"No message passing in training"	3	Held for the node-classification pipeline only: the link-prediction and graph-classification variants substituted convolutional or sampling encoders, so the scope was overstated	unsupported
Post-processing is effective only above a graph-density threshold	4	The effect is positive on `ogbn-products` and negative on `ogbn-proteins`, and both are dense, so a density ordering does not separate them	unsupported
A leaderboard position on `ogbn-mag`	1	The graph had been reduced to a paper–paper subgraph, so the value is not comparable with entries scored on the full heterogeneous graph	unsupported
Monotonic gate convergence follows from fixed cluster assignments	3	The training code routes by the gate against task labels and never against cluster targets, so the premise of the argument was not the objective being optimised	refuted

Three of the six are consequences of stages this paper instruments directly: the read column (stage 1), the two feature routes (stage 2), and the objective the gate was actually trained on (stage 3). The other three were scope overstatements, and we mark them as unsupported rather than merely withdrawn, because the difference between "we measured something else" and "we no longer have the measurement" is the difference this paper is about.

I What Would Have to Change for Us to Be Wrong

An audit paper has a characteristic failure mode of its own: it can be written so that no observation could contradict it, describing a way in which a number may mislead while committing to nothing that could turn out false. Each of the four claims above can be stated in a form that one measurement would refute, and this section states them that way. Our measurements are consistent with all four; what is written here is what would have disproved them.

P1 is falsified by a threshold below the positive median. The claim is that Hits@K under the official evaluator reports the fraction of positives clearing a bar set by K negative scores, and that on the benchmarks we examine that bar sits above the median positive. We have computed t_K against the median positive score for four heuristics on ogbl-ddi and four on ogbl-collab, eight comparisons in total, and the bar is above the median in all eight (Table 12). One counterexample would localise the finding: a dataset whose negative multiset is large enough relative to K that the K-th largest negative score falls below the typical positive. On ogbl-ddi, where $|\mathcal{N}|$ is smaller than

$|\mathcal{P}|$, this cannot happen for the reasons in Appendix B.2; on a dataset with many more negatives than positives it could, and we would want to be the ones to report it.

P2 is falsified by a graph whose uniform sampler is harmless at high density, or harmful at low. The claim is that a uniformly drawn pair collides with the graph at a rate equal to the density, so the damage a uniform sampler does is predictable from the dataset card. We measured the collision rate against the density on seven thinned versions of ogbl-ddi (Table 19) and on ogbl-ppa, and the two agree everywhere within the sampling error of the estimate. A single graph where the measured collision rate departs from its density by more than that error would break the mechanism; none of the eight we measured does.

P3 is falsified by a pipeline that survives the constant-feature rerun. Here the claim is not statistical and can be settled by construction: Proposition 4 shows that if negatives are given a zero block while positives go through a feature function, the two classes are linearly separable from the block pattern alone, and a constant feature therefore suffices. The prediction is exact and unconditional, so a single counterexample — a pipeline with that structure in which constant features score 1.000000 on some split and not on others, or fail to separate at all — would refute it. We have not found one, and we do not expect to: the claim is a statement about linear separability, not about learning.

P4 is falsified if the selection spread is always comparable to the test spread. The claim is that a validation-selected number can be uninformative when the configurations being selected among are separated by less on validation than on test. The ratios in Table 26 run from 0.41 to 14.4, so the phenomenon is not universal; if every benchmark we had looked at produced a ratio near 1, the finding would be a footnote rather than a failure mode. The honest summary is that we have three node-classification benchmarks and one of them shows the extreme case; whether the extreme case is common is a question we cannot answer from our own measurements, and we do not claim to.

What we did not measure. Five benchmarks, four structural heuristics, three negative samplers, one link-prediction pipeline, one evaluator family. Everything above is stated for that scope. We have not measured knowledge-graph link prediction, recommendation, or time-split evaluation, and there is no reason to expect the numbers to transfer unchanged; the mechanisms should, which is exactly the kind of claim that the four paragraphs above are designed to let a reader test.

J GUIDANCE

J.1 THE CHECKLIST, WITH THE OBSERVED CONSEQUENCE OF OMITTING EACH ITEM

1. **Report the metric's degrees of freedom.** *Observed consequence of omitting this: a range from* 0.0000 *to* 0.9997 *for one fixed scorer on one fixed dataset. Applied to ourselves: no link-prediction value of ours is printed without* K*, without* $|\mathcal{N}|$ *and without the identity of the negative set, in either the main text or Table 7.*
2. **Check the threshold.** *Observed:* $t_{20} = 472$ *against a median positive score of* 358*, so more than half of the true edges sit below the bar. Applied to ourselves: threshold and median are reported side by side, for four heuristics on two datasets, in Appendix D.3.*
3. **Run the route-symmetry test.** *Observed consequence of omitting this: a pipeline scoring* 1.000000 *with constant features and a training error of exactly* 0.0000*. Applied to ourselves: our own pair features are built by one code path for both classes, which is why the symmetric arms reach* 0.129 *where the asymmetric arm reaches* 1.000*.*
4. **Ablate the features.** *Observed:* $1.000 \to 1.000$ *under the asymmetric route, against* $0.129 \to 0.000$ *under the symmetric one. Applied to ourselves: we ran both arms and report the one that does not reach* 1.000*; the noise arm gives* 0.5014*.*
5. **Separate the splits in the text.** *Observed: a* 22.1*-point gap on ogbn-products, and a* 0.199 *gain from a validation-edge switch that the output format does not record. Applied to ourselves: validation and test appear in separate columns with the gap throughout; which of our inputs came from outside the benchmark, and what the label block does with a training node's own target, are stated in Appendices L and M.*

6. **Report the selection budget.** *Observed: a selection signal* $14\times$ *narrower than its own effect, and a regret fifteen times the signal that caused it. Applied to ourselves: the number of configurations and both spreads are given for every run group in Appendix G.4.*
7. **Include sanity anchors.** *Observed: three heuristics reproduce published values to the precision at which those values are published, which is how the instrument was established before it was used to make a negative point. Applied to ourselves: the anchors are in Appendix B.4 — three heuristics against published values, plus a constant scorer and a random scorer.*

Nothing above is a standard this paper sets for others and not for itself; where our own numbers would fail one of the seven, Appendix H.4 says so.

J.2 Implications for Four Audiences

Section 8 gives a checklist, which is a list of actions. This section gives the same material by role, because a probe matters only to a reader positioned to act on it, and different readers are positioned to act on different ones. Nothing here is a new finding; everything is one of the measurements above, restated as something a particular reader can do.

J.3 For maintainers of a benchmark or leaderboard

A leaderboard fixes a number's meaning before any submitter touches it, so three of the four failure modes are either created or prevented at that level.

1. **Publish the identity of the negative set, not only its size.** The evaluator already fixes the sampler for a given dataset, which is the right call, but nothing on the entry says so. A reader cannot tell a submission that used the shipped negatives from one that drew its own, and Section 5 shows the two differ by 39 points on ogbl-collab with the scorer held fixed. Naming the sampler in the entry costs a column.
2. **Report K as the quantile it is.** $K{=}20$ against $95{,}599$ negatives is a level-2.1×10^{-4} quantile, not "the top 20". The same K against a differently sized negative multiset is a different statistic, so a K without $|\mathcal{N}|$ cannot be compared across datasets.
3. **Say when a metric has a reachable ceiling.** On ogbl-collab three of four heuristics stop moving at $K{\geq}100$ because 54.5% of the test positives have no common neighbour at all (Appendix D.3). A value of 0.55 on that graph is not a weak model; it is the end of the range. A one-line note in the metric's definition prevents the misreading.
4. **Add a companion column that does not depend on the negative multiset.** AUC or AP next to Hits@K is cheap and separates "the scorer ranks well" from "the scorer clears a bar". It is insurance, not a guarantee: Section 5 shows a sampler change that damages both.
5. **Consider requiring one ablation from link-prediction submissions**: rerun with the input features replaced by a constant. It is one line of code, and Appendix F.2 shows why it is the difference between a measured 1.000000 and an undetected pipeline artefact.

J.4 For authors preparing a submission

1. **Treat the number as a triple** — metric definition, negative multiset, K — rather than a scalar. A reader comparing two triples that differ in one component is comparing that component; say which one it is.
2. **Give validation and test in separate labelled columns with the gap between them.** On ogbn-products the configurations we evaluated differ by 2.5×10^{-4} on validation and by 3.7×10^{-3} on test, a ratio of 14.4 (Appendix G.4). Reporting only the selected value hides the fact that the selection signal was narrower than the effect being selected.
3. **State the selection budget.** How many configurations were evaluated, what the validation spread was, and what the test spread of those same configurations turned out to be. Three numbers, all of which are already computed by anyone who tunes anything.
4. **Do not quote an ensemble without the per-run spread.** Our ogbl-ddi ensemble is 0.8005 and the per-run standard deviation is 0.0461; the first number without the second invites a comparison that the data does not support.

5. **Declare inputs that came from outside the benchmark.** Appendix L records that one of our configurations uses a pre-trained embedding computed elsewhere. The paper's own argument is that such differences should be visible, not assumed, and that applies to our submission first.

J.5 For reviewers and area chairs

1. **Ask what K and $|\mathcal{N}|$ were**, and whether the reported value would survive a different pair. This is one question, and it is answerable from the paper without any re-execution.
2. **Ask how the negative set was constructed.** If the answer is "the official one", the entry is comparable; if it is anything else, the value needs its own column rather than a place in the ranking.
3. **Ask how many configurations were evaluated on validation**, and whether any of them was the one that would have been chosen on test. The gap is a property of the benchmark, not of the authors' care.
4. **Ask for the rerun with constant features.** If it scores 1.0 on a link-prediction task, the reported value is not measuring what the metric's name suggests, and no amount of training fixes that.
5. **Read "reproduces a published number to four decimals" as evidence about the protocol, not about the model.** Our three heuristics agree with their published anchors to printing precision (Appendix B.4), which tells a reader that the pipeline is complete and nothing at all about quality.

J.6 For readers re-using a published value

1. A published value is a triple. Comparing two values that differ in the negative multiset compares samplers, and Section 5 gives the size of that comparison: 39 points on one graph, nothing on another.
2. The sampler's effect scales with graph density, and the density is printed on the dataset card. On ogbl-ddi it is 0.1173; on ogbl-collab, 3.5×10^{-5}.
3. A value selected on validation is the centre of a range, not a point. Where the selection spread is narrower than the test spread, the range is wide, and the number should be quoted with it.
4. A rise in a leaderboard position across submissions may be a change of convention rather than a change of model. The check is cheap: ask for the triple before believing the movement.

The four roles are links in one chain. A maintainer fixes what the number means, an author reports it, a reviewer checks it, and a reader compares it. Each of the four probes above is executable by at least one of them without cooperation from the others, which is why we state them as checks rather than as recommendations: the point is not that reporting should be more careful, but that the information needed to be careful is cheap to produce and is not currently produced.

K Reproducibility

K.1 Reproducibility: Inventory and Run Map

The package is organised so that a reader can get from any number in this paper to the code that produced it without reading the code first. Table 31 is that map. Every artefact is a JSON file written by exactly one script, and no script reads another script's output, so a disagreement between two tables is a disagreement between two runs, not a transcription error.

The entry point is `run_all.py`. The file `verify_paper.py` re-computes every value printed in the manuscript from the JSON artefacts and exits non-zero if any of them disagrees, which turns "the paper agrees with the artefacts" into a command rather than a claim. The structural probes (P1, P2, and the mechanism measurements in Appendices E.1–E.2) need no GPU, no training and no random seed beyond the sampler's; they are pure functions of the published splits and can be re-run in seconds.

Table 31: Every artefact, the script that writes it, and where it is used. P1–P4 refer to the four failure modes of the main text.

Script	Artefact	Used for
`audit_ddi.py`	`audit_ddi.json`	P1 four-heuristic statistics; P3 ladder on ddi
`audit_neg.py`	`audit_neg.json`	P2 sampler rule on ddi; K sweep
`audit_neg_ext.py`	`audit_neg_ext.json`	P1/P2 on collab; all four heuristics
`audit_contam.py`	`audit_contam.json`	contamination and mixing sweeps
`audit_density.py`	`audit_density.json`	density sweep; ppa collision rate
`audit_route_collab.py`	`audit_route_collab.json`	P3 ladder on collab
`audit_select.py`	`audit_select.json`	P4 gaps per run; selection spreads
`make_clean.py`	`audit_clean.json`	protocol-clean re-measurement
`make_figs.py`	`figures/*.pdf`	all figures, including the appendix
`verify_paper.py`	—	exit status of the whole audit

K.2 Minimal Reproductions

Each probe can be run on its own, in minutes, without a GPU, from the official splits. Table 32 records the invocation used for the numbers in this paper and what the reader should see. Three of the four need no training at all, and the fourth is a logistic regression over four columns; this is the reason the probes are cheap enough for a reviewer to re-run rather than take on trust.

Table 32: Command-level recipes. All commands run from the extracted package root; the last one is the one that matters, because it re-checks every number printed in this paper against the JSON artefacts and exits non-zero on any mismatch.

Probe	Command	What it produces
P1, four heuristics	`python audit/audit_ddi.py`	Hits@20, t_{20}, AUC and AP per heuristic
P2, sampler rule	`python audit/audit_neg.py`	sampler grid and K sweep on ddi
P1–P2, 2nd dataset	`python audit/audit_neg_ext.py`	the same two probes on collab
P2, mechanism	`python audit/audit_contam.py`	contamination and mixing sweeps
P2, density	`python audit/audit_density.py`	seven-point thinning; ppa collision
P3, 2nd dataset	`python audit/audit_route_collab.py`	the six-arm ladder on collab
P4	`python audit/audit_select.py`	per-run gaps and selection budgets
P4, clean table	`python audit/make_clean.py`	Table 7
figures	`python audit/make_figs.py`	Figures 1–5 and A1–A4
all of the above	`python audit/verify_paper.py`	exit status of the whole audit

L What We Trained, and What the Search Looked Like

The paper makes a claim about the *reporting* of a number, not about model quality, and the models we train are chosen to keep that distinction clean. Every re-measurement uses either a three-layer MLP over concatenated feature blocks or a two-layer GraphSAGE encoder with a two-layer predictor head, both deliberately small; none of the four failure modes of Section 4–7 is repaired by a better architecture, and a simpler model makes it harder to attribute a probe's result to model capacity. Table 33 holds the configuration behind each reported value, and Table 35 defines the feature blocks its *Features* column names.

These five configurations are what survived a search. Table 34 accounts for all nineteen configurations we trained, eighty-eight independent runs in total, with the device each one ran on and the test value it reached before any post-processing. Two things in the table matter beyond bookkeeping.

The first is that the search itself is an instance of the failure mode of Section 7. On ogbn-arxiv the configuration we selected and the one that came second differ in the fourth decimal place — 0.7510 against 0.7501 — which is a difference of the same order as the run-to-run spread of either of them. On ogbn-products three configurations trained from different feature sets land on exactly the same test value, 0.6554. A selection procedure that is asked to choose among options separated by 10^{-4}

is being asked to read a signal narrower than its own noise, which is the quantitative form of the claim that the selection budget is a property of the benchmark and not of the submitter.

The second is that the table records which device each run used, which is information a reader may reasonably want and which we would rather state than leave to inference. The pipeline places a run on the GPU when CUDA is available and on the CPU otherwise; eight configurations ran on the CPU, seven on the GPU, and four are link-prediction runs that do not record the field. The structural probes that carry the paper's quantitative claims — P1, P2, the contamination and density measurements, and the symmetric arms of P3 — need no GPU at all, because their scorers have no parameters.

Two further conventions are stated here because the tables do not show them. First, the training runs share a fixed seed for everything that determines the data — the split, the negative sampling, the feature graph — and vary only the training seed, which is set to the base value plus the run index; that is why five runs on the same configuration produce five slightly different values, and why we report those values as a mean and a standard deviation rather than a single number. Second, the ogbn-arxiv entry uses a pre-trained node embedding computed outside this paper (the `ext` block of Table 35); we declare it here because a reader comparing the value with a submission that does not use it is comparing two different input sets, and the audit's own argument is that such differences should be visible rather than assumed.

Table 33: The configuration reported for each benchmark, exactly as the training script recorded it. *Features* lists the input blocks concatenated before the first linear layer; the block names are defined in Table 35. *Dev.* is the device the run executed on: the pipeline selects CUDA when it is available and CPU otherwise, and the record shows which happened.

Benchmark	Architecture	H/L	Features	lr	Epochs	Batch	Dev.
ogbn-arxiv	MLP×3	256/3	`raw128+deg+ext768+sign1-2`	0.01	300	8192	cpu
ogbn-products	MLP×3	256/3	`raw100+deg+sign1-3`	0.01	300	32768	cuda
ogbn-proteins	MLP×3	256/3	`deg+ef8+lp112+sign1-6`	0.003	300	32768	cpu
ogbl-ddi	SAGE×2 + MLP×2	256/2	`none`	0.005	200	65536	auto
ogbl-collab	SAGE×2 + MLP×2	128/2	`raw`	0.005	150	65536	auto

Table 34: Every training configuration recorded by the audit, with the number of independent runs it contains and the device it ran on. The selection column marks the configuration whose value is reported in Table 7; the others are the search that produced it, and the last column gives that search's test value *without* post-processing, so that it is comparable across rows and with itself rather than with the selected value. Two tags are probes rather than candidates: `nodeemb_valedges` is the declared validation-edge-as-input variant of Appendix G.3, and `arxiv_cs_only` isolates the effect of post-processing.

Benchmark	Tag	Runs	Dev.	Selected	Test (no post-proc.)
ogbn-arxiv	`_smoke2`	2	cpu		0.7286
	`arxiv_cs_only`	5	cpu		0.7380
	`arxiv_final`	5	cpu	✓	0.7510
	`arxiv_gext_v2`	5	cpu		0.7501
	`arxiv_giant`	5	cuda		0.7381
ogbn-products	`p_f5`	5	cuda	✓	0.6691
	`p_lp_ov`	5	cuda		0.6437
	`p_lp_v2`	5	cuda		0.6437
	`p_sign2_ov`	5	cuda		0.6554
	`p_sign2_v2`	5	cuda		0.6554
	`p_sign2_v3`	5	cuda		0.6554
ogbn-proteins	`pr_cs_v2`	5	cpu		0.6826
	`pr_f2`	5	cpu		0.6865
	`pr_f3`	5	cpu		0.6934
	`pr_f4`	5	cpu	✓	0.6970
ogbl-collab	`nodeemb_sage`	3	auto	✓	0.5197
	`nodeemb_valedges`	3	auto		0.5807
ogbl-ddi	`nodeemb_sage`	5	auto	✓	0.5800
	`nodeemb_valedges`	5	auto		0.7792

Table 35: The input blocks that the `feats` strings in Table 33 name, and what each one is computed from. Every block that depends on labels or on the graph is computed from the training split only, which is what makes the validation-edge probe of Appendix G.3 a probe rather than a routine choice.

Block	Name	Definition
`raw`	the node features shipped with the dataset	read from `data.x`; standardisation is configurable and is `none` in every run reported here
`deg`	node degree	`log1p` of the degree, standardised on the training split
`ef`	incident edge features	the dataset's edge attributes summed onto their source node; used for ogbn-proteins, which has no node features of its own
`lp`	propagated training labels	the one-hot targets of the training nodes, propagated over the training graph; the width equals the number of target columns
`ext`	external embedding	a pre-trained node embedding computed outside this paper (GIANT-XRT, 768 columns); used only for ogbn-arxiv
`sign`	diffused features	the concatenated block, diffused over the training adjacency in the style of SIGN; the width is the number of blocks times the number of hops

L.1 What the Probes Print

Appendix K.2 lists the commands. This section shows what they produce, because a reader who intends to re-run a probe should be able to tell in a few seconds whether their copy is behaving. The block below is the complete output of the worked-example probe, verbatim, from the run that produced Table 9.

```
audit_worked: one number, end to end (ogbl-ddi, CN, hits@20)
  ddi: |V|=4267  train pairs=2135822  sym pairs=2135822  mean deg=500.5
  ddi: |test pos|=133489  |test neg|=95599
  ddi: hits@20=0.177296  thr=472.0  positives above=23667/133489
  ddi: pos q=222.0/358.0/433.0  neg q=2.0/25.0/93.0
  ddi: zeros  pos=0.0003 neg=0.1816   neg>pos_median=0.0029
  ddi: example positive (895,128) cn=358.0 deg=950,781 beats 99.7% of neg
  ddi: example negative (591,1640) cn=472.0 deg=976,1214
wrote audit_worked.json in 8.8s
```

The third line is the reported value and the two quantities behind it. The fourth gives the quartiles of each class, and the fifth the two facts Section 4 leans on: almost a fifth of the negative pairs have no common neighbour, and 0.29% of them score above the median positive. The last two lines name the pairs the appendix uses as examples, so a reader can look up either one in the released split.

The wall-clock time of each structural probe is in Table 36. None of these scripts trains anything, loads a GPU, or reads another script's output; the figure at the bottom is the whole cost of every non-training number in this paper.

Table 36: Wall-clock time of each structural probe, on the machine of Appendix O.1. The commands are the ones in Table 32. Times are for a single run; the probes are deterministic, so re-running reproduces the printed values exactly.

Script	Produces	Wall clock
`audit_worked.py`	the steps of Table 9	8.8 s
`audit_neg_ext.py`	the sampler grid and K sweep on collab	7.3 s
`audit_contam.py`	collision rate; the mixing sweep	21.4 s
`audit_density.py`	seven-point thinning; ppa collision rate	50.6 s
`audit_route_collab.py`	the six-arm ladder on collab	3.0 s
`audit_ddi.py`	four-heuristic statistics; the ladder on ddi	under a minute
all structural probes	**every non-training number here**	≈ 91 s

M Implementation Details, per Benchmark

This paper contributes no architecture and every number in it leaves a published code path, so this section is short by design. It is written out anyway, because two of the four probes are statements about wiring, and wiring can only be checked if it is visible. Nothing below is novel; that is why it is stated.

Data. Every dataset is read through the official OGB package (Hu et al., 2020), and we apply no preprocessing of our own to any of them. The pipeline is implemented in PyTorch (Paszke et al., 2019), with the loaders and the graph layers taken from PyTorch Geometric (Fey & Lenssen, 2019):

```
from ogb.nodeproppred import PygNodePropPredDataset, Evaluator
ds = PygNodePropPredDataset(name=name, root=root)
data, split, ev = ds[0], ds.get_idx_split(), Evaluator(name=name)

from ogb.linkproppred import PygLinkPropPredDataset, Evaluator
ds = PygLinkPropPredDataset(name=name, root=root)
data, split, ev = ds[0], ds.get_edge_split(), Evaluator(name=name)
ev.K = 20 if name == "ogbl-ddi" else 50
```

Two loading details are easy to get wrong and would have changed numbers silently. First, `ogbn-proteins` carries no node features of its own: its 8-column `edge-feat` matrix is a species encoding and the scalar edge weight every published baseline uses is its row sum, which we compute from the raw `edge.csv.gz` and `edge-feat.csv.gz`, not from the PyG copy, whose `to_undirected` duplication would no longer line up with the feature rows. Second, for the link-prediction datasets the training graph is symmetrised with `to_undirected`; the negative multiset is never symmetrised, since it is a set of ordered pairs.

Node classification. The model is a three-layer MLP with hidden width 256, dropout 0.5 (Srivastava et al., 2014) and no weight decay, trained with Adam (Kingma & Ba, 2015) at $\mathrm{lr} = 0.01$ for at most 300 epochs in batches of $32{,}768$, with the checkpoint chosen on the official validation split and early stopping after 40 evaluations without improvement. The loss is cross-entropy for the single-label datasets and `binary_cross_entropy_with_logits` for `ogbn-proteins`, whose 112 target columns are scored by micro ROC-AUC. The input to the MLP is the concatenation of the feature blocks named in Table 35; every block that depends on labels or on the graph is built from the training split alone, with `lp_alpha` $= 0.9$ and twenty propagation steps for the label block, and three hops for the diffused block. Post-processing is Correct & Smooth with the four parameters of Table 7; the candidate list always contains `none`, and the selected setting is the one with the highest official validation score.

The label block, and a training node's own target. The fifth item of the checklist in Section 8 asks whether labels entered any input. For this pipeline the answer is not yes or no but a coefficient, and we print it, because the label block is the one stage where our own numbers inherit a decision that no reporting format asks anyone to declare. The block is built by propagating the one-hot targets of the training nodes over the training graph for twenty steps at `lp_alpha` $= 0.9$ with the update $\mathbf{Y} \leftarrow \alpha\,\mathbf{M}\mathbf{Y} + (1-\alpha)\,\mathbf{Y}_0$, where $\mathbf{M}$ is the symmetrised adjacency normalised as $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ *without* self-loops and $\mathbf{Y}_0$ is the one-hot matrix that is non-zero on the training rows alone. The residual term is what makes the coefficient explicit: the row of a training node keeps $1-\alpha = 0.1$ of its own target and re-injects it at every step, and no masking, out-of-fold or leave-one-out scheme is applied to it. The earlier pipeline of Section 6 built the same block with self-loops and clamped the training rows back to their own one-hot after every step, so there the coefficient was 1.0: every training node was handed its own label as an input feature.

Both consequences are stated rather than left to be inferred. Of the five configurations whose values appear in Table 7, one uses the block (`ogbn-proteins`); the other two node-classification pipelines contain no label block at all, and neither link-prediction configuration has one. And where the block is used, our number inherits a 0.1 self-term that the benchmark does not ask anyone to declare — the same species of unrecorded stage that P4 measures. We report the coefficient instead of removing it: removing it would be a different pipeline, not a correction of this one, and the value it affects is a validation–test gap rather than a claim of accuracy.

One property of `ogbn-products` matters for how its P4 result should be read. Its official validation split holds 39,323 of 2,449,029 nodes, 1.6% of the graph, so one node of validation accuracy is 2.5×10^{-5}. A selection rule asked to choose among settings separated by 2.5×10^{-4} is resolving about ten nodes.

Link prediction. The encoder is a two-layer GraphSAGE (Hamilton et al., 2017) with hidden width 256 and dropout 0.5 over a learnable node embedding, optionally concatenated with the dataset's node features; the predictor is a two-layer MLP over the pair $[h_u, h_v]$. Both are trained jointly with Adam at $\mathrm{lr} = 5 \times 10^{-3}$ for at most 200 epochs in batches of 65,536 positive pairs against one sampled negative each, evaluated every five epochs on the official validation split, and stopped after 40 evaluations without improvement. Five runs are made per configuration, with the training seed set to $42+r$; the split, the negative sampling and the feature graph are the same across runs. Every reported link-prediction value is the mean of those five runs, and the per-run standard deviation is reported alongside it in Table 7. The only switch we ever change is `use_valedges_as_input`, whose effect on a single declared split is $+0.199$ and which is off in the row we report.

Structural probes. The probes that carry the paper's quantitative claims — the identity of Section 4, the sampler grid, the contamination and density measurements, and the symmetric arms of the route ladder — train nothing. They build the symmetrised adjacency of the training graph once, evaluate a closed-form score for every positive and negative pair, rank, and hand the two score vectors to the official evaluator. The asymmetric arms fit a single logistic regression (Pedregosa et al., 2011) on 400,000 pairs. Total wall-clock cost is given in Table 36.

What we did not do. We did not tune an architecture, did not ensemble across architectures for the reported values, and did not attempt a competitive entry on any leaderboard. The one ensemble we computed is reported as such in Table 7, with its per-run spread, because the difference between it and the mean is itself an instance of Section 4. The scope of the implementation is part of the claim: a failure mode that needs a 10-layer model to appear is not what this paper is about, and none of the four appears less clearly at this scale.

N Sensitivity to Our Own Choices

An audit that reports a difference owes a reader the answer to one question: is the difference in the benchmark, or in the auditor's settings? Every quantity below is a knob this paper turned. Each is varied while the rest of the pipeline is held fixed, and the outcome is reported whichever way it falls. The section ends with the one knob whose variation does move a number, because that case turns out to be the paper's argument rather than an exception to it.

The seed of the sampler. The sampler probe of Section 5 draws its own negative multiset, so the draw has a seed. The same probe under five seeds per sampler is in Table 37. On ogbl-ddi nothing moves: the collision rate stays within 0.0018 of the density and the reported value is 0.000000 for all ten draws, because the 20th largest negative score is more than three times the median positive one under either rule. On ogbl-collab the *value* of the uniform sampler moves, and the reason can be stated exactly. There the official fiftieth largest negative score is 1, the negative multiset is 100,000 pairs of which 99.94% score zero, and a uniform draw puts about 3.5 of its pairs on the graph. Whether the threshold ends up as 0 or 1 therefore depends on whether those few pairs exist, and the statistic takes 0.545404 or 0.416996 accordingly: three of five seeds give the first and two give the second, a spread of 12.8 points between two draws of the same sampler with the scorer, the positives, the multiset size and the evaluator all held fixed.

That is not a defect in the sampler and not an artefact of our implementation. It is Section 4's identity being read out loud: the threshold is an extreme order statistic of a multiset that the metric does not fix, so on a sparse graph it is decided by a handful of pairs, and the reported value inherits their luck. The degree-weighted sampler shows the same thing more loudly — its threshold on ogbl-collab ranges from 16 to 35 across the same five seeds, and its value from 0.0229 to 0.0509.

Ties at the threshold. The evaluator counts a positive when its score is strictly greater than t_K, and on two of the eight scorer–dataset pairs we use there are positives whose score is exactly t_K.

Table 37: The sampler probe under five seeds per sampler. "Collision" is the fraction of the drawn pairs that are edges of the training graph, computed against the density of each graph (0.117306 on ogbl-ddi, 3.5×10^{-5} on ogbl-collab). The value column is Common Neighbour under the official evaluator. The official row uses OGB's own multiset, whose collision rate is 0 on both graphs; the seed column does not apply to it.

		ogbl-ddi (Hits@20)		**ogbl-collab** (Hits@50)	
Sampler	**seed**	**collision**	**value**	**collision**	**value**
uniform	2024	0.117784	0.000000	0.000060	0.416996
uniform	7	0.116779	0.000000	0.000040	0.416996
uniform	11	0.118275	0.000000	0.000020	0.545404
uniform	2025	0.117763	0.000000	0.000030	0.545404
uniform	99	0.116518	0.000000	0.000030	0.545404
degree-weighted	2024	0.344805	0.000000	0.000790	0.022945
degree-weighted	7	0.343257	0.000000	0.000850	0.026225
degree-weighted	11	0.344062	0.000000	0.000590	0.041594
degree-weighted	2025	0.341280	0.000000	0.000610	0.050897
degree-weighted	99	0.344920	0.000000	0.000660	0.032118
official	—	0.000000	0.177296	0.000000	0.416996

The count and the value under either convention are in Table 38. On ogbl-ddi the convention moves the value by 0.0016 and flips 209 positives; on ogbl-collab it moves it by 12.8 points and flips 5,949, because that graph's threshold sits on the integer 1 and an eighth of the positives sit on it too. Both figures are within the paper's own subject: a number that changes by 12.8 points under a convention nobody states is a number whose reader cannot reconstruct it.

Table 38: The tie rule at the threshold. "Tied" counts the positives whose score equals t_K exactly; they are counted under $\geq$ and not under $>$. The evaluator uses the strict form.

Dataset	**Scorer**	t_K	**with** $>$	**with** $\geq$	**tied positives**
ogbl-ddi	CN	472.0	0.177296	0.178861	209
ogbl-ddi	AA	68.551	0.186075	0.186075	0
ogbl-ddi	RA	0.724	0.062267	0.062267	0
ogbl-ddi	PA	1,328,448	0.039262	0.039262	0
ogbl-collab	CN	1.0	0.416996	0.545404	5,949
ogbl-collab	AA	0.205	0.524013	0.524250	11
ogbl-collab	RA	0.008	0.523452	0.523689	11
ogbl-collab	PA	3,901	0.080360	0.080382	1

Thinning, and the regularisation of the route fit. Two further choices were varied. For the density measurement we drop edges at random at three rates under three seeds; at rate 1.0 the three seeds are identical by construction, and at 0.5 and 0.1 the measured collision rate tracks the density to within 0.0019 and 0.0012 across seeds, while the density itself varies by 0.0001 between seeds at either rate. For the asymmetric arm of the route ladder we varied the logistic regression's inverse regularisation over five orders of magnitude, $C \in \{0.01, 0.1, 1, 10, 100\}$. The training error ranges from 0.000058 to 0.000300, and the two properties the section leans on do not move at all: the number of distinct decision values over the 100,000 test negatives is 1 in all five fits, and the fraction of positives above that single value is 1.000000 in all five.

One run's two numbers. The last item is not a knob we can turn, and is listed here because it is the one a reader is most likely to mistake for one. A training run reports a validation score and a test score, and both belong to the same checkpoint. On ogbl-products our selected configuration gives validation 0.89019 and test 0.66908 before post-processing and 0.90563 and 0.75458 after it; the two columns move by 0.0154 and 0.0855 respectively over the same three settings. The validation-edges probe makes the same point on a switch that is declared in the loader: with it on, ogbl-ddi has a mean validation value of 0.7277 and a mean test value of 0.7792, while the configuration we

report with it off has a test value of 0.5800; on ogbl-collab the same switch takes validation from 0.6109 to 0.8418 but test only from 0.5197 to 0.5807, so the two columns disagree by 26.1 points in the direction that would flatter a submission.

What this leaves. Of the knobs above, the sampler seed, the thinning seed and the regularisation strength leave every structural quantity in the paper unchanged. The sampler seed moves one value on one graph, by 12.8 points, in the direction that makes the metric look better, and the mechanism that moves it is the mechanism the paper describes. We report it as a fifth piece of evidence rather than as a caveat, and we have left the corresponding row of Table 16 standing as it was computed, with the draw that produced it named in the caption.

O Environment and Data

O.1 Hardware, Runtime, and Determinism

All experiments run on one workstation: an RTX 5060 with 8 GB of VRAM and a 16-core CPU. As Table 34 records, individual configurations are placed on the GPU or the CPU according to what the pipeline finds available, so the device column there, not this sentence, is the authority on that question. The audit scripts and the re-measurement driver together execute in about six hours of wall-clock time, dominated by the training runs; the structural probes — P1, P2, the contamination and density measurements, and the symmetric arms of P3 — account for under two minutes of that total, because their scorers have no parameters.

Determinism is stated per artefact, not in general. The structural measurements are deterministic given the split: the samplers use a fixed seed (`RandomState(2024)`) and the scores are closed-form, so re-running any of them reproduces the printed value bit for bit. The training runs are not deterministic in that sense — they use different seeds per run, which is why every link-prediction number is reported as a mean and a standard deviation over independent runs. Configuration selection uses the official validation split only; where a number depends on a choice, the choice is named in the corresponding table.

O.2 The Benchmarks, as Loaded

Every number in this paper comes from the official OGB loaders, so the statistics below are the ones a reader obtains by calling `get_edge_split()` or `get_idx_split()` on the released data rather than values we computed in a preprocessing step of our own. We print them because two of the probes depend on them: the ratio of positives to negatives on a split determines the quantile that K corresponds to (Appendix B.1), and the density of the graph determines how far a uniform sampler moves the value (Appendix E.2).

Table 39: The five benchmarks, as loaded from the official OGB packages. Density is $|E|/|V|^2$ on the symmetrised graph and is reported for the link-prediction datasets, where it governs the sampler effect.

Benchmark	$\lvert V\rvert$	$\lvert E\rvert$	feat.	class.	train	valid	test
ogbn-arxiv	169,343	1,166,243	128	40	90,941	29,799	48,603
ogbn-products	2,449,029	123,718,280	100	47	196,615	39,323	2,213,091
ogbn-proteins	132,534	79,122,504	0	2	86,619	21,236	24,679
ogbl-ddi	4,267	2,135,822	0	—	1,067,911	133,489	133,489
ogbl-collab	235,868	2,358,104	128	—	1,179,052	60,084	46,329
ogbl-ppa	576,289	42,463,862	58	—	—	—	—

Three of these numbers matter for how a reported value should be read, and all three are properties of the released split, not of any model. On ogbl-ddi the negative multiset is smaller than the positive set on both evaluation splits ($95{,}599$ against $133{,}489$ test positives), so $K{=}20$ is a quantile at level 2.1×10^{-4} rather than the "top K" a reader might picture. On ogbl-collab the two splits differ in size by a factor of 1.3 and the validation split is larger than the test split, which is the structural reason a value tuned on validation need not transfer. On ogbl-ppa the graph has 42M directed pairs

and a mean degree of 73.7, which is why we measured its collision rate rather than scoring it: at that size a closed-form scorer over all test pairs is outside the audit's budget.