ROUTING SHUFFLES DO NOT IDENTIFY ADAPTIVE BENEFIT: THEORY AND A CONTROLLED ACM-GCN AUDIT

Anonymous authors
Paper under double-blind review

ABSTRACT

When a trained network loses accuracy after its routing gates are shuffled, it is natural to attribute the loss to the value of adaptive routing. We show why this inference requires care. Shuffling changes the assignment of gates while preserving their collection; replacing them by a constant changes both. For separable gate-to-output computation and additive node scores, the expected shuffled score equals an average of constant-gate scores. The shuffle drop therefore upper-bounds the frozen network's advantage over the best constant gate. Once message passing couples the gates, either ordering is possible, as two explicit constructions show. We examine the distinction in ACM-GCN on five previously studied graphs, using 2,100 tuning fits and 750 final models. On Roman-empire, the joint shuffle drop is 11.59 accuracy points, the gap against a searched frozen constant is 4.27 points, and the advantage over a separately trained constant-routing model is 1.42 points. These comparisons concern different properties of the network. Our analysis explains when a shuffle test admits a bound and why a practical assessment of routing should also include searched constant gates and separately trained static controls.

1 INTRODUCTION

A graph neural network may assign different filters or experts to different nodes (Luan et al., 2022; Han et al., 2024; Shi et al., 2024; Chen et al., 2025). To examine whether these assignments matter, one can hold the trained weights fixed, randomly permute the gate vectors among nodes, and measure the resulting loss of performance. A large loss shows that the trained model is sensitive to this reassignment. Does it also show that the model benefits from adaptive routing?

There are two distinct comparisons behind this question. We may keep the trained weights and replace every gate by a common value, asking how much the particular network gains from using different gates. Alternatively, we may train a new network whose gates are constrained to be constant, asking whether adaptive routing improves what can be learned under the chosen training procedure. A shuffle test makes neither comparison directly. It changes which node receives each gate, but preserves the collection of gates itself.

This difference is decisive when information is passed between nodes after routing. A node may benefit from receiving differently processed messages from its neighbours, even if the assignment of gates to those neighbours is arbitrary. Constant routing removes that variety; shuffling does not. By contrast, when each node's prediction depends only on its own replacement gate, an averaging argument relates shuffled performance exactly to constant routing. We call this latter condition *separability*. It concerns the computation after the intervention, not the architecture's name: routing after precomputed graph propagation may be separable, whereas routing followed by further message passing may not be (Yue & Zhang, 2026).

We prove that, for separable computation and an additive node score, the expected shuffled score is an average of scores obtained by assigning a common gate to all nodes. The shuffle drop is therefore an upper bound on the advantage over the best constant gate. For coupled computation, neither ordering holds in general. Two constructions establish this distinction, and a degree-dependent

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.

calculation describes one way in which coupling changes the comparison. We then consider what can be inferred from a finite collection of permutations and a numerical search for constant gates.

Our experiments examine these comparisons in ACM-GCN. On Roman-empire, shuffling both layers reduces accuracy by 11.59 points, whereas replacing the gates by a searched constant reduces it by 4.27 points. Separately trained adaptive and constant-routing models differ by only 1.42 points. These numbers answer different questions, despite their common use as measures of routing benefit. The study covers five graphs previously used in this project; it is a controlled examination of one routing architecture, rather than a claim about all graph neural networks.

The question is related to the distinction between attention weights and explanations (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019), and to the dependence of removal-based explanations on the replacement operation (Covert et al., 2021; Hooker et al., 2021). Here the replacement operations are explicit enough to admit a precise comparison. Table 1 summarizes them; the remainder of the paper develops the mathematical distinction and then examines its practical consequences.

Table 1: The three comparisons studied in this paper. In the first two, the trained expert weights remain fixed. In the third, the weights are learned again under the stated training budget.

Comparison	What changes?	What does it measure?
Shuffle sensitivity	Permute the learned gate tuples among nodes.	Sensitivity to the learned assignment.
Frozen constant gap	Give every node the same validation-selected gate tuple.	An upper bound on the advantage over the best constant.
Training comparison	Train a model with a shared constant gate.	The adaptive advantage under the chosen training procedure.

2 Notation and the three comparisons

We first consider a single trained network. Fix its graph, features, labels and weights, and let T be a nonempty set of m evaluation nodes. There are $N \geq 2$ nodes in all. At node v, a gate vector $q_v \in \Delta_K$ gives nonnegative weights summing to one over K experts; the collection of these vectors is the gate field Q. If several layers are intervened on, q_v denotes the complete tuple of gates at that node, and its domain is a product of simplexes. Write $z_v(Q)$ for the resulting logits and

$$F(Q) = \frac{1}{m} \sum_{v \in T} s_v(z_v(Q)) \tag{1}$$

for the score of the network. We take larger scores to be better and assume that all scores and suprema below are finite. The functions s_v are fixed. Accuracy and negative cross-entropy have this additive form; ordinary dataset-level AUC does not.

For a gate tuple c, let $Q_c = \mathbf{1}c^\top$: every node now receives the same tuple. We also call this operation *broadcasting* c. Let $\mathcal{C}$ be the allowed class of constants, and suppose it contains every learned row q_v. In our experiments it is the full simplex for a single layer, or the product of two simplexes when both layers are changed. A uniform random permutation P instead gives $(PQ)_v = q_{P(v)}$. We shall compare

$$S^* = \sup_{c \in \mathcal{C}} F(Q_c), \qquad \Delta_{\mathrm{sh}} = F(Q) - \mathbb{E}_P F(PQ),$$
$$\mathrm{ben} = F(Q) - S^*, \qquad \Gamma = \mathbb{E}_P F(PQ) - S^*. \tag{2}$$

Here S^* is the best constant score, Δ_{sh} is the expected shuffle drop, and ben is the advantage of the learned gate field over the best constant. They satisfy

$$\boxed{\mathrm{ben} = \Delta_{\mathrm{sh}} + \Gamma.} \tag{3}$$

The term Γ compares shuffled routing with the best constant. It is the part of the frozen advantage that the shuffle drop alone does not determine. In particular, ben may be negative, and all three quantities refer to the fixed network weights. The comparison with a separately trained constant-routing network is a different question, taken up in Section 5.

There is a further distinction between heterogeneity and adaptation. On the fixed graph, each PQ is simply one reassignment of the observed gate vectors. If $\Gamma > 0$, some such heterogeneous assignment must outperform every allowed constant. This does not establish that its gates must be learned from the input. Conversely, because the observed gate vectors may themselves depend on the input, these reassignments do not define an input-independent rule on new inputs. A fixed hash-routing rule (Roller et al., 2021) is a separate comparator.

In practice, one usually finds a constant $\hat{c}$ by numerical search rather than computing S^* exactly. The observed gap $\widehat{\mathrm{ben}} = F(Q) - F(Q_{\hat{c}})$ satisfies $\mathrm{ben} \le \widehat{\mathrm{ben}}$. Thus a good constant can rule out a large advantage, but failure to find one cannot prove that an advantage exists. To distinguish a finite experiment from an expectation, let $P_1, \dots, P_M$ be the recorded permutations and write

$$\mu_M = M^{-1} \sum_{i=1}^{M} F(P_i Q), \qquad \Delta_M = F(Q) - \mu_M,$$
$$\Gamma_M = \mu_M - S^*, \qquad \widehat{\Gamma}_M = \mu_M - F(Q_{\hat{c}}). \tag{4}$$

The corresponding identities are exact: $\mathrm{ben} = \Delta_M + \Gamma_M$ and $\Gamma_M \le \widehat{\Gamma}_M$. In particular, a negative $\widehat{\Gamma}_M$ gives a lower bound on how much the observed shuffle drop overstates the frozen advantage. A statement about the expectation over permutations requires an additional sampling bound. We keep these two forms of inference separate in the experiments.

3 Separable and Coupled Routing

The relation between shuffled and constant routing depends on where the gates enter the computation. We call the intervened computation *separable* if, for every evaluated node v, $z_v(Q)$ depends on Q only through q_v; otherwise it is *coupled*. For example, a final mixture of fixed expert logits is separable even when the experts themselves use the graph. Aggregating those mixed outputs over neighbours can destroy separability.

This criterion applies to particular interfaces in published models. Node-MoE's final mixture (Han et al., 2024, Eq. (1)) is separable when its expert outputs are fixed and its classifier acts row by row, even though its gate generator is a GNN. The same reasoning applies to FilterMoE's cached experts and node–channel mixture with a row-wise predictor (Yue & Zhang, 2026, Eqs. (4), (12)); the channel gates at a node form a product of simplexes. These are deductions from the stated computations, not experimental replications. They would no longer follow if subsequent operations aggregated across nodes or recomputed cross-node normalization.

For separable routing the comparison is particularly simple. At any one node, a uniform shuffle selects each learned gate with equal probability. Its expected contribution to the score is therefore the average contribution under the corresponding broadcast constants.

Theorem 1 (Separable gates)**.** *For separable gates and an additive score,*

$$\mathbb{E}_P F(PQ) = \frac{1}{N} \sum_{j=1}^{N} F(Q_{q_j}) \le S^*. \tag{5}$$

Consequently $\Gamma \le 0$ *and* $\mathrm{ben} \le \Delta_{\mathrm{sh}}$. *The slack is the average suboptimality of the broadcast learned gates, vanishing exactly when all attain* S^*.

Proof. Each $P(v)$ is uniform over nodes. Separability gives

$$\mathbb{E}_P s_v(z_v(PQ)) = \frac{1}{N} \sum_{j} s_v(z_v(Q_{q_j})).$$

Exchange the finite sums over v and j; every $q_j \in \mathcal{C}$. □

The proof uses only separability and additivity; neither convexity nor linearity is needed. The gap in the inequality measures the average suboptimality of the broadcast gates, rather than their dispersion. Even identical gates can have positive suboptimality if their common value is not optimal. Additivity,

on the other hand, is essential. Consider a positive node with score $2g_+$ and a negative node with score $1 + 2g_-$. At gates $(1, 0)$ the AUC is one, its expected value after shuffling is $1/2$, and every constant gate gives AUC zero. The logits are separable, but the AUC advantage exceeds the shuffle drop.

Separability has a second useful consequence: two fixed replacement fields determine all their node-wise recombinations if the individual node scores are retained (Appendix Corollary 5). We now turn to what fails when gates at different nodes interact.

Theorem 2 (Coupled counterexamples)**.** *For accuracy and the full constant simplex, coupled one-step message-passing networks can have (i)* $\Delta_{\mathrm{sh}} = 0$ *and* $\mathrm{ben} = \Gamma = 1$*, or (ii)* $\Delta_{\mathrm{sh}} = 1 - 2/N$ *and* $\mathrm{ben} = 0$*. Thus neither ordering between shuffle drop and constant-class advantage holds throughout the coupled class.*

For (i), use an even complete graph with $N \geq 4$, all positive labels, and fixed expert outputs $e_1 = (1, -1)$ and $e_2 = (-1, 1)$. Mix them with gate $g_u \in [0, 1]$ *before* a coordinatewise ReLU, so $h_u = \mathrm{ReLU}(g_u e_1 + (1 - g_u)e_2)$:

$$h_u = \big((2g_u - 1)_+, (1 - 2g_u)_+\big), \qquad \bar{h}_v = \frac{1}{N-1} \sum_{u \neq v} h_u. \tag{6}$$

Predict the positive class exactly when both coordinates of $\bar{h}_v$ are at least $1/4$. A constant gate makes one emission coordinate zero at every node, so all predictions are wrong. A balanced assignment, with half the gates zero and half one, instead gives both neighbour means at least $(N/2 - 1)/(N - 1) \geq 1/4$. It is consequently perfect, as is every permutation of it. The network needs a mixture of the two types of messages, but not a particular assignment of them. Indeed, a fixed balanced assignment suffices without feature-dependent gate training. The minimum used by the readout can be implemented by a ReLU MLP.

The reverse discrepancy in (ii) starts from a cycle-indexed array of expert margins. The learned gates and the uniform constant give identical logits and perfect accuracy, while only two of the N experts are correct at each node. For $N = K = 20$, shuffling therefore loses 90 accuracy points despite the exact constant replacement. Adding a sufficiently small, strictly positive contribution from neighbours makes the computation genuinely coupled without changing any hard-permutation prediction or the correctness of the uniform constant. Appendix B gives the construction and the perturbation bound. Its expert count grows with N: the purpose of the examples is to establish the possible orderings, not their frequency in trained networks.

4 Measuring the effect of coupling

The preceding examples suggest separating two reasons why shuffled and optimal constant scores may differ. Let $A = N^{-1} \sum_j F(Q_{q_j})$ be the average broadcast score, $S = S^* - A \geq 0$ its shortfall from the best constant, and $U = \mathbb{E}_P F(PQ) - A$ the change from broadcasting to heterogeneous reassignment. Then

$$\Gamma = U - S. \tag{7}$$

Under separability, $U = 0$ and only the nonnegative shortfall S remains. With coupling, U may outweigh that shortfall. An influence bound for U is given in Appendix E. For accuracy, a more direct diagnostic compares predictions node by node: keep a node's assigned gate fixed, but replace all the other gates by that same value. The following identity distinguishes improvements from deteriorations that an average score may conceal.

Proposition 3 (Paired correctness changes)**.** *For each recorded permutation* P_i *and evaluated node* v*, let* a_{iv} *indicate correctness under* $P_i Q$*, and* b_{iv} *indicate correctness under the broadcast field* $Q_{q_{P_i(v)}}$*. The paired fields agree on the complete gate tuple at* v*. Averaging over the* Mm *pairs, define*

$$G_M = \overline{a(1-b)}, \quad L_M = \overline{(1-a)b}, \quad R_M = G_M + L_M, \quad \eta_M = \bar{b} - A. \tag{8}$$

Then, for $U_M = \mu_M - A$*,*

$$U_M = G_M - L_M + \eta_M, \qquad |U_M - \eta_M| \leq R_M \leq D_M, \tag{9}$$

where D_M *is the paired prediction-disagreement rate. Marginally balanced permutation designs make* $\eta_M = 0$*; uniform random permutations give only* $\mathbb{E}\eta_M = 0$*.*

Proof. Use $a - b = a(1-b) - (1-a)b$ and $|a-b| = a(1-b) + (1-a)b$. A correctness change requires a prediction change. Under marginal balance, each evaluated node receives every gate index equally often, so $\bar{b} = A$. □

The quantity R_M counts changes in correctness without allowing gains and losses to cancel. If it is positive, the tested predictions cannot be separable. The converse need not hold, since coupled logits may leave the predicted classes unchanged. Cancellation can also be complete: take two adjacent positive-label nodes with gates $(0, 1)$, each predicting its neighbour's gate. Then $G_M = L_M = 1/2$ and $R_M = D_M = 1$, yet $U_M = \eta_M = 0$ under either permutation. Appendix E gives the details.

We can also ask how the graph itself enters the first construction of Theorem 2. A node is correct precisely when its neighbourhood contains sufficiently many gates of both types. Counting those gates gives an exact answer.

Proposition 4 (Exact degree characterization)**.** *Use the emission h_u from Eq. (6), now with neighbourhood mean $\bar{h}_v = d_v^{-1} \sum_{u \in \mathcal{N}(v)} h_u$, and the same all-positive-label readout on any simple unweighted loopless graph without isolated evaluated nodes. Let N be even with exactly $N/2$ hard gates of each type. Then*

$$\Gamma = \frac{1}{m} \sum_{v \in T} \sum_{k=\lceil d_v/4 \rceil}^{\lfloor 3d_v/4 \rfloor} \frac{\binom{N/2}{k}\binom{N/2}{d_v - k}}{\binom{N}{d_v}}. \tag{10}$$

If every evaluated node has $d_v \geq 2N/3$, every balanced assignment is perfect, so $\Delta_{\mathrm{sh}} = 0$ and $\mathrm{ben} = \Gamma = 1$.

To see this, the number of first-type neighbours under a uniform permutation has a hypergeometric distribution. Correctness requires this number to lie between $d_v/4$ and $3d_v/4$, while every constant still scores zero. At the stated threshold, each neighbourhood contains at least $d_v - N/2 \geq d_v/4$ nodes of either type. Below it, the integer cutoffs can make the success probability non-monotone in degree. This describes the particular emitter and readout of the construction, rather than a general density criterion for GNNs.

We checked the formula against exhaustive balanced-assignment enumeration in 50 (N, d) cases and six seeded graphs, and separately checked the dense case and the AUC, margin and cross-entropy examples. The script retains graph seeds and edge lists. Appendix Figure 4 compares balanced reassignment with fixed parity and node-hash assignments, illustrating why the choice of heterogeneous control matters even in this simple example.

5 Experiments with ACM-GCN

The theory concerns a fixed network. To connect it with the usual question of whether adaptive routing helps in practice, we also train the corresponding static models. This lets us compare the shuffle loss, the gap against a constant at fixed weights, and the difference between separately trained models within the same study.

We use the two-layer ACM-GCN computation of Luan et al. (2022) on Roman-empire, Amazon-ratings, Minesweeper, Chameleon-filtered and Actor (Platonov et al., 2023; Rozemberczki et al., 2021; Pei et al., 2020). All five graphs had been used previously in this project. We retain their ten native splits and use three final seeds for each split. Features are taken from the original benchmarks, without the signed row-sum transformation used in an earlier study reported in the appendix.

Training and comparisons. We train three versions of the same expert architecture: node-adaptive routing, a learned gate shared by all nodes, and fixed balanced one-hot routing based on public node indices. Each receives twelve configurations spanning widths $\{128, 512\}$, learning rates $\{.001, .003, .01\}$ and dropout probabilities $\{0, .5\}$, with 1,000 Adam updates per fit. Expert initializations agree at the same configuration and seed. Since each version is tuned independently, the selected widths, parameter counts and actual computation may differ. Residual GCN and ResNet supply additional context rather than architecture-matched routing controls; they use width 512, LayerNorm, depths $\{1, 3, 5\}$ and the benchmark's AdamW procedure. Altogether, 2,100 tuning fits select 250 configurations, followed by 750 final fits. Checkpoints and configurations are chosen by

Table 2: Test performance of the five trained models, averaged over ten overlapping splits and three final seeds. Entries give the equal-cell mean ± SD. Accuracy is in percent; Minesweeper's primary AUC is on the 0–1 scale. Residual GCN and ResNet provide context, not architecture-matched routing comparisons.

Graph / score	Adaptive	Constant	Hash	Res. GCN	ResNet
Chameleon-f. (acc.)	39.20 ± 3.38	39.30 ± 4.10	33.53 ± 3.59	43.23 ± 3.02	37.64 ± 3.23
Actor (acc.)	33.97 ± 0.74	34.14 ± 1.14	25.15 ± 1.15	30.20 ± 1.16	35.46 ± 0.71
Roman-empire (acc.)	80.57 ± 0.57	79.15 ± 0.60	49.31 ± 0.56	72.98 ± 0.86	65.73 ± 0.40
Amazon-ratings (acc.)	51.13 ± 0.58	50.54 ± 0.49	37.47 ± 0.62	48.08 ± 0.81	46.19 ± 0.56
Minesweeper (AUC)	0.9056 ± 0.0060	0.9057 ± 0.0061	0.7607 ± 0.0113	0.8956 ± 0.0071	0.5086 ± 0.0124
Minesweeper (acc.)	85.95 ± 0.55	85.93 ± 0.57	80.91 ± 0.53	85.03 ± 0.59	80.00 ± 0.00

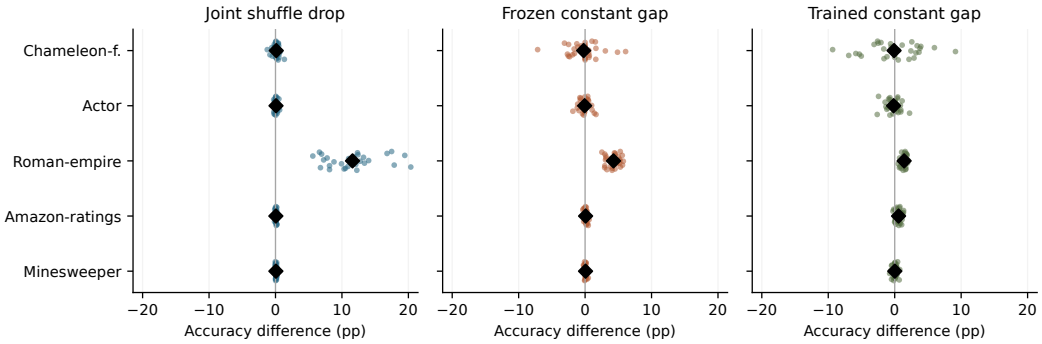

Figure 1: The three accuracy comparisons on the same study: the joint shuffle drop, the gap against a validation-accuracy-selected frozen constant, and the gap against a separately trained constant. Dots show all 30 split-seed cells per graph; diamonds show their equal-cell means. A positive gap against a searched frozen constant is only an upper bound on the advantage over the best constant.

validation accuracy, except on binary Minesweeper, where we use margin-score ROC-AUC. Full settings and tie rules appear in Appendix A.

For each of the 150 trained adaptive networks, we intervene at three places. At the *first layer*, we replace its gates and recompute the downstream router. At the *terminal layer*, we hold the hidden states fixed and replace only the final mixing weights. The *joint* intervention replaces both fields. Terminal mixing is separable; the other two computations can couple gate replacements across nodes. We compare the learned gates with their mean, uniform gates, constants selected by validation cross-entropy or accuracy, fixed hash routing, and 100 full-node permutations. Joint shuffles move a node's two-layer gate tuple together. The first-layer and joint searches use 1,753 and 3,169 candidates, respectively. Terminal search also permits a numerical FP64 convex validation-CE certificate. These frozen comparisons use accuracy and negative CE, not AUC.

What changes when the static model is trained? Table 2 reports all five models on every graph. Adaptive routing improves accuracy over the trained constant on Roman-empire by 1.42 points and on Amazon-ratings by 0.58 points, with descriptive 95% intervals $[1.28, 1.57]$ and $[0.41, 0.76]$. This advantage is metric-dependent: on Amazon, negative cross-entropy instead favors the constant (Table 6). The remaining primary-score differences are -0.10 accuracy points on Chameleon, -0.17 on Actor, and approximately -0.0001 AUC on Minesweeper. Their intervals include zero, leaving the direction unresolved under this resampling summary; they do not establish equivalence.

Absolute performance is also important. The residual GCN reaches 72.98% accuracy on Roman and 48.08% on Amazon, while adaptive ACM exceeds the training-class-frequency baseline in mean primary score on all five graphs. These checks address the quality of the models being compared. They are not a claim of state-of-the-art performance or exact reproduction of published benchmark means.

What does shuffling add to this picture? Figure 1 places the three comparisons on a common accuracy scale, including on Minesweeper. Roman-empire provides the clearest discrepancy. Shuffling both layers loses 11.59 points, but the gap against a validation-selected frozen constant is 4.27 points, and the separately trained models differ by 1.42 points. The shuffle loss therefore cannot be read as the benefit of adaptive training. The discrepancy can go the other way as well: Amazon's mean joint shuffle drop is 0.06 points, below its 0.58-point trained advantage (Tables 3 and 6). This latter comparison does not certify a positive optimal frozen-class gap.

The finite permutation experiment also permits a conditional statement about expected shuffle performance. All three Roman interfaces have negative simultaneous upper bounds on the optimal-constant composition gap. For the joint intervention the bound is -6.57 points, rounded upward. Thus, under the stated independent-permutation model, the expected shuffle drop exceeds the mean frozen advantage by at least 6.57 points. The other twelve dataset/interface bounds do not certify a strictly negative gap. Positive gaps against searched constants cannot establish a positive optimal gap, because the best constant may be better than the one found.

These two types of uncertainty should be distinguished. For training contrasts, we average the three seeds within each split and bootstrap the ten split means 10,000 times with a fixed seed. The resulting intervals are descriptive: overlapping splits do not constitute independent graphs. For the expected-shuffle calculation, we use one declared family of fifteen accuracy comparisons and condition on the graphs, checkpoints and selected constants, as in Appendix Proposition 7. The appendix reports every cell, the inconclusive comparisons, class-frequency baselines, selected widths and parameter counts.

6 Related work

Routing serves several purposes in mixture-of-experts models. It can allocate computation (Shazeer et al., 2017; Fedus et al., 2022), make use of diverse graph experts (Wang et al., 2023), or adapt an operation to a particular input. Our question concerns the last of these at a fixed intervention point. A comparison of prediction scores alone does not settle questions of capacity, load balancing or computational efficiency.

The debate over attention as an explanation makes a related distinction between inspecting weights and intervening on a computation (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). Those works offer different interpretations of attention experiments. More generally, removal-based explanations depend on what replaces the removed component (Covert et al., 2021), and unrestricted permutation may force extrapolation (Hooker et al., 2021). We study a particular instance of this problem in which the replacement is either a permutation or a common gate, and the computation admits an exact comparison under separability.

Learned and fixed routing have been compared in several MoE settings (Roller et al., 2021; Dikkala et al., 2023; Fan et al., 2024). UniPool reports model-dependent losses after randomizing one deep-half MoE layer at a time (Huang et al., 2026). Routing Absorption compares learned and frozen-random sparse-attention training, attributing part of their similar performance to co-adaptation (Aquino-Michaels, 2026). Neither comparison includes the best constant at the same checkpoint, which is the object needed for our frozen-advantage question. FilterMoE compares node-only, node-shared channel-only and joint graph-filter routing under matched training (Yue & Zhang, 2026). Such comparisons complement, rather than duplicate, a permutation of a trained gate field. Other studies of routing and aggregation similarly emphasize the importance of strong fixed alternatives (Rubio-Madrigal & Burkholz, 2026; Weiss, 2026).

The auxiliary cross-entropy identities in the appendix use standard Jensen–Bregman and ensemble-diversity arguments (Banerjee et al., 2005; Wood et al., 2023). The layerwise and fixed-replacement calculations are instances of multilinear interpolation (Sundararajan et al., 2020). The contribution here is the comparison these elementary facts allow: a precise condition for ordering shuffle and constant scores, counterexamples when it fails, and experiments that distinguish those scores from a training advantage.

7 DISCUSSION

The mathematical distinction is broader than the experiments. Separability is a property of the computation after the replaced gates, and so can be checked in other architectures. Our empirical evidence, however, concerns only two-layer ACM-GCN on five previously exposed graphs with overlapping splits. Independently tuned models may have different widths, and equal search and step budgets need not mean equal parameter counts or computation. The finite budgets do not establish convergence; GCN and ResNet are contextual baselines, not architecture-matched controls. The earlier interpolation and sensitivity studies are retained separately as exploratory evidence.

The counterexamples also have a specific role. The positive-Γ construction uses all-positive labels and shows that heterogeneous messages can suffice even when their assignment does not matter. It establishes a possibility, not its prevalence in trained networks. Likewise, the influence bound may be vacuous. In the experiments, the first-layer and joint searches give only upper bounds on the advantage over the best constant. The terminal affine calculation certifies validation cross-entropy, not test accuracy, while the expected-shuffle bounds depend on the stated conditional sampling assumptions. None of the additive-score results extends automatically to AUC.

A shuffle test nevertheless answers a useful question: how much does this trained network rely on its particular assignment of gates? Under separability and additive scoring, its expected performance drop also bounds the frozen advantage over constant routing. Beyond that setting, the comparison must be made directly. A convincing assessment should therefore state which gates are replaced and which score is measured, search for constants without test feedback, and train static alternatives when the claim concerns learning. Where average changes are small, paired gains and losses help distinguish genuine stability from cancellation. With these distinctions in place, shuffle experiments become easier to interpret without asking them to answer a question they were not designed to test.

REPRODUCIBILITY STATEMENT

The accompanying material supports three levels of reproduction: reconstructing the tables from saved scores, replaying the selected checkpoints, and repeating training. It records the training configurations, source and input hashes, validation selections, checkpoints, candidate constants and intervention predictions. Model replay and checks on the complete policy and prediction files preceded held-out scoring. The full model bank is separate from the smaller source/results archive; the earlier studies and their provenance are kept distinct from the present experiment.

AI-USE STATEMENT

Generative AI tools assisted literature web searches, coding, experimental data verification and reruns, final prose rewriting, and figure preparation. The human co-authors alone formulated the theorems, determined counterexample directions and proof strategies, drafted the proofs, designed the experimental protocols, and interpreted the results. AI assistance was limited to the supporting tasks listed above. The human co-authors finalized the manuscript and take responsibility for all its content, including AI-assisted text, code, figures, and numerical work.

REFERENCES

Keston Aquino-Michaels. Routing absorption in sparse attention: Why random gates are hard to beat. *arXiv preprint arXiv:2603.02227*, 2026. URL https://arxiv.org/abs/2603.02227.

Arindam Banerjee, Srujana Merugu, Inderjit S. Dhillon, and Joydeep Ghosh. Clustering with bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005. URL https://jmlr.org/papers/v6/banerjee05b.html.

Xuanze Chen, Jiajun Zhou, Yadong Li, Jinsong Chen, Shanqing Yu, and Qi Xuan. Mixture of message passing experts with routing entropy regularization for node classification. *arXiv preprint arXiv:2502.08083*, 2025. URL https://arxiv.org/abs/2502.08083.

Ian Covert, Scott Lundberg, and Su-In Lee. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021. URL https://jmlr.org/papers/v22/20-1316.html.

Nishanth Dikkala, Nikhil Ghosh, Raghu Meka, Rina Panigrahy, Nikhil Vyas, and Xin Wang. On the benefits of learning to route in mixture-of-experts models. In *EMNLP*, 2023. URL https://aclanthology.org/2023.emnlp-main.583/.

Dongyang Fan, Bettina Messmer, and Martin Jaggi. Towards an empirical understanding of MoE design choices. *arXiv preprint arXiv:2402.13089*, 2024. URL https://arxiv.org/abs/2402.13089.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://www.jmlr.org/papers/v23/21-0998.html.

Haoyu Han, Juanhui Li, Wei Huang, Xianfeng Tang, Hanqing Lu, Chen Luo, Hui Liu, and Jiliang Tang. Node-wise filtering in graph neural networks: A mixture of experts approach. *arXiv preprint arXiv:2406.03464*, 2024. URL https://arxiv.org/abs/2406.03464.

Giles Hooker, Lucas Mentch, and Siyu Zhou. Unrestricted permutation forces extrapolation: Variable importance requires at least one more model, or there is no free variable importance. *arXiv preprint arXiv:1905.03151*, 2021. URL https://arxiv.org/abs/1905.03151v2.

Minbin Huang, Han Shi, Chuanyang Zheng, Yimeng Wu, Guoxuan Chen, Xintong Yu, Yichun Yin, and Hong Cheng. Unipool: A globally shared expert pool for mixture-of-experts. *arXiv preprint arXiv:2605.06665*, 2026. URL https://arxiv.org/abs/2605.06665.

Martin Jaggi. Revisiting Frank–Wolfe: Projection-free sparse convex optimization. In *ICML*, volume 28, pp. 427–435, 2013. URL https://proceedings.mlr.press/v28/jaggi13.html.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *NAACL-HLT*, pp. 3543–3556, 2019. URL https://aclanthology.org/N19-1357/.

Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Revisiting heterophily for graph neural networks. In *NeurIPS*, 2022. URL https://arxiv.org/abs/2210.07606.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009. URL https://arxiv.org/abs/0907.3740.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph convolutional networks. In *ICLR*, 2020. URL https://arxiv.org/abs/2002.05287.

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 *ICLR*, 2023. URL https://arxiv.org/abs/2302.11640.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. Hash layers for large sparse models. *arXiv preprint arXiv:2106.04426*, 2021. URL https://arxiv.org/abs/2106.04426.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 2021. URL https://arxiv.org/abs/1909.13021.

Celia Rubio-Madrigal and Rebekka Burkholz. Fixed aggregation features can rival GNNs. *arXiv preprint arXiv:2601.19449*, 2026. URL https://arxiv.org/abs/2601.19449.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017. URL https://arxiv.org/abs/1701.06538.

Yu Shi, Yiqi Wang, WeiXuan Lang, Jiaxin Zhang, Pan Dong, and Aiping Li. Mixture of experts for node classification. *arXiv preprint arXiv:2412.00418*, 2024. URL https://arxiv.org/abs/2412.00418.

Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The shapley taylor interaction index. In *ICML*, volume 119, pp. 9259–9268, 2020. URL https://proceedings.mlr.press/v119/sundararajan20a.html.

Haotao Wang, Ziyu Jiang, Yuning You, Yan Han, Gaowen Liu, Jayanth Srinivasa, Ramana Rao Kompella, and Zhangyang Wang. Graph mixture of experts: Learning on large-scale graphs with explicit diversity modeling. In *NeurIPS*, 2023. URL https://arxiv.org/abs/2304.02806.

Eyal Weiss. Cost-sensitive neighborhood aggregation for heterophilous graphs: When does per-edge routing help? *arXiv preprint arXiv:2603.24291*, 2026. URL https://arxiv.org/abs/2603.24291.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *EMNLP-IJCNLP*, 2019. URL https://aclanthology.org/D19-1002/.

Danny Wood, Tingting Mu, Andrew M. Webb, Henry W. J. Reeve, Mikel Luján, and Gavin Brown. A unified theory of diversity in ensemble learning. *Journal of Machine Learning Research*, 24 (359):1–49, 2023. URL https://jmlr.org/papers/v24/23-0041.html.

Zichao Yue and Zhiru Zhang. Gate the filter, not the message: Node-channel mixtures for pre-propagation GNNs. *arXiv preprint arXiv:2606.01660*, 2026. URL https://arxiv.org/abs/2606.01660.

A EXPERIMENTAL DETAILS

A.1 ARCHITECTURE AND TRAINING

The ACM models use the active computation of upstream commit `75b4853f29b2`. We retain its low-pass, high-pass and identity experts, its ReLU operations, and its threefold gate scaling, while omitting unused upstream parameters. Forward and gradient comparisons were made against the upstream implementation. The three routing variants share the same order of random draws for initialization; we do not assert identity with the upstream initialization, which also draws unused tensors. ACM uses a bidirected graph with self-loops and row normalization.

The contextual baselines follow commit `a431395582e9`. They use symmetric graph normalization, width 512, residual feed-forward blocks, LayerNorm, dropout .2, and AdamW with learning rate $3 \cdot 10^{-5}$ and zero weight decay. Their search covers depths 1, 3 and 5, giving three configurations per split, compared with twelve for each routing variant. Full commit hashes and source provenance are included in the accompanying material.

Every fit runs for 1,000 steps. The routed models use Adam with weight decay .0005. We use seed 2601 for tuning and seeds 2701–2703 for final training. The checkpoint with the highest validation primary score is retained, with the earliest step breaking exact ties. Configuration selection uses the same score, with the earliest grid index breaking ties. For binary classification, ACM has two logits and uses cross-entropy: a single ReLU output would restrict its logit. The binary baseline models instead have one logit and use binary cross-entropy, as in the upstream implementation. The primary AUC is computed from the unrounded logit difference or scalar logit, avoiding ties introduced by softmax rounding. Accuracy and negative CE are also retained for every model.

The fixed-hash comparison assigns a balanced collection of one-hot gates using a local seeded CPU random-number generator and public node indices, independently at each layer. This assignment uses neither features nor labels. The three routing variants share expert initializations at a given width and seed, but independent validation selection may choose different widths. For this reason we report parameter counts and actual computation rather than treating them as matched after selection.

A.2 CONSTANT-GATE SEARCHES AND PERMUTATIONS

At the first layer, the constant search consists of a 15-point quarter-simplex grid, the uniform and mean gates, 128 sampled learned rows broadcast to all nodes, and eight Adam restarts with 201 retained points each. For the joint intervention, it consists of 225 grid pairs, the uniform pair and paired means, 128 paired broadcasts, and fourteen restarts with 201 retained points each. Adam acts on constant-gate logits with learning rate .05 and moments $(.9, .999)$. All expert weights remain fixed. The mean and broadcast gates may use unlabeled nodes of the transductive graph, but the search objectives use validation labels only. CE selection breaks ties by candidate order; accuracy selection breaks them first by lower CE and then by that order.

For the terminal intervention, the validation-CE objective is convex. We apply Frank–Wolfe in FP64 for at most 2,000 iterations, with steps $2/(t+2)$ and a dual-gap target of 10^{-6}. We report the attained gap, including when this target is not met. These are numerical optimization certificates; they are not formal proofs obtained by outward-rounded arithmetic.

Each cell uses separate SHA256-derived PCG64 streams for broadcast sampling, restarts, hash assignments and the 100 permutations. The permutation bank is shared by the three intervention interfaces. All validation-selected policies are fixed before prediction export. Shuffled outputs retain the predicted classes and the mean FP64 log-probability vector at each node. These suffice to compute mean additive accuracy and negative CE on a later evaluation mask, but not the variance of per-permutation CE or permutation AUC.

A.3 VERIFICATION AND UNCERTAINTY

We required exact independent replay of all 750 final checkpoints on their producing backend. An additional, fixed set of 25 cells—split zero and the first final seed for every dataset and model—was checked with an independently implemented CPU forward computation and a stated tolerance for comparison with the GPU. Checks covered input and source hashes, finite gradients, complete logs,

earliest-tie selection and independent recomputation of validation scores. Successive records fixed the tuning choices, replay results, constant-policy selections for all 150 adaptive checkpoints, and intervention predictions before the single held-out evaluation. These checks concern computational fidelity; absolute test performance is a separate measure of model quality.

Training used PyTorch 2.8.0 and CUDA 12.8 on RTX 5090 GPUs. The three final seeds ran on separate shards with distinct drivers, whose versions are recorded in the artifact. Thus hardware and seed were not independently crossed. Exact replay on the producing backend should not be read as exact reproducibility across every hardware and software combination. The recorded learning curves, validation-selected settings and budget-boundary counts describe optimization under the fixed training budget.

For each paired contrast, we average the three seeds within each split and then bootstrap the ten split means 10,000 times using PCG64 seed 2901. These intervals describe variation over reused, overlapping splits. For expected shuffle accuracy, Appendix Proposition 7 is applied to a family of $H = 15$ comparisons with $\alpha = .05$ and 3,000 permutation accuracies per comparison, using the validation-accuracy-selected constant. This family was declared separately from the historical $H = 6$ and $H = 4$ families. The bounded-accuracy certificate is not applied to CE or AUC.

Table 3: Accuracy differences (percentage points). All entries are learned minus the stated frozen control. Each mean retains all 30 cells.

Graph / interface	Shuffle	CE const.	Acc. const.	Mean	Uniform	Hash
Chameleon-f. / first	-0.05	+0.25	+0.13	-0.01	+0.16	+3.28
Chameleon-f. / terminal	+0.12	-0.56	-0.41	+0.05	-0.04	+7.77
Chameleon-f. / joint	+0.12	+0.50	-0.22	+0.13	-0.00	+9.74
Actor / first	+0.06	+0.05	-0.14	+0.05	-0.23	+4.17
Actor / terminal	+0.02	+0.56	+0.08	+0.01	+1.14	+6.86
Actor / joint	+0.08	-0.04	-0.07	+0.07	+0.64	+8.91
Roman-empire / first	+3.86	+1.64	+1.65	+1.90	+6.76	+36.49
Roman-empire / terminal	+8.25	+2.39	+2.43	+3.96	+21.87	+42.83
Roman-empire / joint	+11.59	+4.27	+4.27	+5.25	+35.73	+56.93
Amazon-ratings / first	+0.06	+0.07	+0.05	+0.04	+4.39	+10.72
Amazon-ratings / terminal	-0.00	+0.15	+0.01	+0.00	+7.96	+15.83
Amazon-ratings / joint	+0.06	+0.21	+0.08	+0.03	+7.95	+19.83
Minesweeper / first	+0.02	+0.06	+0.08	+0.00	+1.94	+11.71
Minesweeper / terminal	+0.04	+0.17	+0.12	+0.05	+0.34	+12.00
Minesweeper / joint	+0.06	+0.18	+0.07	+0.09	+1.59	+16.17

Table 4: Negative-CE differences (nats/node). All entries are learned minus the stated frozen control. Each mean retains all 30 cells.

Graph / interface	Shuffle	CE const.	Acc. const.	Mean	Uniform	Hash
Chameleon-f. / first	-0.0025	-0.5415	-0.2803	-0.0031	-0.1318	+0.5631
Chameleon-f. / terminal	-0.0069	-0.2460	-0.0263	-0.0081	-0.0490	+1.8166
Chameleon-f. / joint	-0.0091	-0.8558	-0.3112	-0.0112	-0.1718	+2.3358
Actor / first	-0.0025	-0.0174	+0.0076	-0.0035	+0.0702	+0.4172
Actor / terminal	-0.0017	-0.0735	+0.0019	-0.0019	-0.0474	+0.3517
Actor / joint	-0.0035	-0.0981	-0.0143	-0.0048	-0.0406	+0.7302
Roman-empire / first	+0.1413	+0.0603	+0.0630	+0.0725	+0.2443	+1.7540
Roman-empire / terminal	+0.3108	+0.0877	+0.0910	+0.1418	+0.5691	+4.0707
Roman-empire / joint	+0.4639	+0.1662	+0.1727	+0.2135	+0.9991	+5.4645
Amazon-ratings / first	+0.0008	+0.0006	+0.0011	+0.0004	+0.0894	+0.3933
Amazon-ratings / terminal	+0.0001	-0.0059	+0.0000	+0.0001	+0.1995	+0.6149
Amazon-ratings / joint	+0.0009	-0.0055	-0.0020	+0.0005	+0.1261	+0.7793
Minesweeper / first	+0.0005	-0.0035	+0.0009	+0.0004	+0.0312	+0.5622
Minesweeper / terminal	+0.0005	-0.0023	-0.0003	+0.0003	+0.0099	+0.5443
Minesweeper / joint	+0.0011	-0.0039	+0.0014	+0.0007	+0.0280	+1.0483

Table 5: New conditional accuracy family: $H = 15$, $\alpha = .05$. Found gap, sampling margin and optimal-gap upper bound are percentage points. A nonnegative upper bound is inconclusive; upper bounds are rounded upward to two decimals. Unrounded values are retained in the artifact.

Graph / interface	Found gap	Margin	Upper bound
Chameleon-f. / first	+0.18	0.72	+0.90
Chameleon-f. / terminal	-0.53	0.72	+0.20
Chameleon-f. / joint	-0.34	0.72	+0.39
Actor / first	-0.20	0.55	+0.35
Actor / terminal	+0.06	0.55	+0.61
Actor / joint	-0.15	0.55	+0.40
Roman-empire / first	-2.21	0.56	-1.65
Roman-empire / terminal	-5.82	0.78	-5.03
Roman-empire / joint	-7.32	0.74	-6.57
Amazon-ratings / first	-0.01	0.53	+0.53
Amazon-ratings / terminal	+0.01	0.53	+0.55
Amazon-ratings / joint	+0.02	0.53	+0.56
Minesweeper / first	+0.05	0.53	+0.59
Minesweeper / terminal	+0.08	0.53	+0.62
Minesweeper / joint	+0.01	0.53	+0.55

Table 6: Paired training contrasts: mean [descriptive 95% interval]. Accuracy differences use percentage points; AUC and negative CE use their original scales.

Graph / metric	Adaptive minus constant	Adaptive minus hash
Chameleon-f. / primary	-0.10 [-1.78, +1.70]	+5.67 [+3.91, +7.34]
Chameleon-f. / negative cross entropy	-0.1499 [-0.4438, +0.1581]	+1.5117 [+0.8647, +2.1322]
Actor / primary	-0.17 [-0.59, +0.27]	+8.82 [+8.46, +9.21]
Actor / negative cross entropy	-0.0139 [-0.0483, +0.0256]	+0.5999 [+0.4985, +0.7436]
Roman-empire / primary	+1.42 [+1.28, +1.57]	+31.26 [+30.86, +31.69]
Roman-empire / negative cross entropy	+0.0550 [+0.0501, +0.0597]	+1.0462 [+1.0354, +1.0564]
Amazon-ratings / primary	+0.58 [+0.41, +0.76]	+13.65 [+13.29, +14.05]
Amazon-ratings / negative cross entropy	-0.0026 [-0.0043, -0.0011]	+0.2282 [+0.2123, +0.2447]
Minesweeper / AUC	-0.0001 [-0.0003, +0.0001]	+0.1450 [+0.1400, +0.1503]
Minesweeper / accuracy	+0.02 [-0.04, +0.07]	+5.04 [+4.65, +5.49]
Minesweeper / negative cross entropy	-0.0057 [-0.0171, +0.0002]	+0.1368 [+0.1260, +0.1448]

Table 7: Active stored parameters per selected final model: minimum–maximum over all cells. Independently selected widths may differ. Unused upstream tensors are omitted in this port.

Graph	Adaptive	Constant	Hash	Res. GCN	ResNet
Chameleon-f.	895137–3580449	894726–3578886	894720–3578880	1720837–3826181	1720837–3826181
Actor	360225–1440801	359814–1439238	359808–1439232	1007621–3112965	1007621
Roman-empire	490056	488454	122112	690706–1743378	690706–2796050
Amazon-ratings	470049	468486	117120–468480	1736709–2789381	684037
Minesweeper	15384	3462–13830	3456–13824	1584641–2637313	531969–2637313

Table 8: Training-class-frequency control, averaged over all ten native splits. Class probabilities use Laplace smoothing. Primary score is accuracy in percent except Minesweeper AUC (0–1); the accuracy column is always percent. Negative CE is nats/node. No test label selects the majority class or class frequencies.

Graph	Primary	Accuracy	Negative CE
Chameleon-f.	24.99	24.99	-1.6032
Actor	25.24	25.24	-1.5738
Roman-empire	13.96	13.96	-2.6140
Amazon-ratings	36.79	36.79	-1.4111
Minesweeper	0.5000	80.00	-0.5004

Table 9: Compute and checkpoint selection. Worker-hours sum recorded fit times (training, validation and checkpoint checking/saving), excluding initialization, engineering, frozen search and separate post-training replay. They are neither wall time nor accelerator-utilization hours. Final median is seconds. Last column counts final checkpoints selected exactly at the 1,000-step budget boundary; a non-boundary choice does not prove optimization convergence.

Arm	HPO fits	Final fits	Worker-h	Final median (s)	At boundary
adaptive	600	150	5.60	25.5	1/150
learned constant	600	150	4.44	20.1	4/150
fixed hash	600	150	4.19	18.1	0/150
gcn	150	150	2.18	28.0	2/150
resnet	150	150	1.27	13.4	2/150

Table 10: Terminal validation-CE Frank–Wolfe gaps (nats): actual residuals and count meeting 10^{-6}. Unmet tolerance is not described as convergence. These numerical intervals concern validation CE only.

Graph	Min.	Median	Max.	Tolerance met
Chameleon-f.	0.000247	0.00282	0.00595	0/30
Actor	4.09e-07	0.000377	0.0014	2/30
Roman-empire	0.000271	0.00173	0.00449	0/30
Amazon-ratings	5.33e-05	0.000228	0.000443	0/30
Minesweeper	5.43e-08	0.000555	0.00168	4/30

Table 11: Frozen accuracy uncertainty: mean [descriptive split-bootstrap 95% interval], in percentage points. The last column uses a found constant and is an upper estimate of the optimal-class composition gap.

Graph / interface	Shuffle drop	Found constant gap	Found composition gap
Chameleon-f. / first	-0.05 [-0.20, +0.10]	+0.13 [-0.61, +1.20]	+0.18 [-0.55, +1.20]
Chameleon-f. / terminal	+0.12 [-0.02, +0.27]	-0.41 [-0.79, -0.04]	-0.53 [-0.87, -0.21]
Chameleon-f. / joint	+0.12 [-0.05, +0.29]	-0.22 [-1.29, +1.11]	-0.34 [-1.34, +0.93]
Actor / first	+0.06 [+0.01, +0.12]	-0.14 [-0.31, +0.01]	-0.20 [-0.38, -0.05]
Actor / terminal	+0.02 [-0.01, +0.06]	+0.08 [-0.02, +0.19]	+0.06 [-0.03, +0.15]
Actor / joint	+0.08 [+0.01, +0.15]	-0.07 [-0.25, +0.11]	-0.15 [-0.39, +0.09]
Roman-empire / first	+3.86 [+3.60, +4.06]	+1.65 [+1.47, +1.82]	-2.21 [-2.36, -2.05]
Roman-empire / terminal	+8.25 [+7.56, +8.88]	+2.43 [+2.18, +2.65]	-5.82 [-6.32, -5.30]
Roman-empire / joint	+11.59 [+11.10, +12.05]	+4.27 [+4.05, +4.51]	-7.32 [-7.81, -6.80]
Amazon-ratings / first	+0.06 [+0.02, +0.11]	+0.05 [-0.02, +0.14]	-0.01 [-0.05, +0.05]
Amazon-ratings / terminal	-0.00 [-0.02, +0.01]	+0.01 [-0.03, +0.06]	+0.01 [-0.02, +0.05]
Amazon-ratings / joint	+0.06 [+0.01, +0.10]	+0.08 [-0.01, +0.16]	+0.02 [-0.05, +0.08]
Minesweeper / first	+0.02 [+0.00, +0.04]	+0.08 [+0.01, +0.17]	+0.05 [-0.03, +0.16]
Minesweeper / terminal	+0.04 [+0.02, +0.06]	+0.12 [+0.04, +0.19]	+0.08 [-0.00, +0.16]
Minesweeper / joint	+0.06 [+0.04, +0.09]	+0.07 [+0.01, +0.12]	+0.01 [-0.04, +0.06]

Table 12: Validation-selected configuration counts over ten splits per graph/arm, before three final seeds. Width column counts 128/512; depth counts 1/2/3/5; routed learning-rate counts .001/.003/.01; routed dropout counts 0/.5. Contextual learning rate/dropout cells instead state their fixed value. No held-out score selects these settings.

Graph / arm	Widths	Depths	Learning rates	Dropouts
Chameleon-f. / adaptive	5/5	0/10/0/0	3/2/5	0/10
Chameleon-f. / learned constant	5/5	0/10/0/0	2/2/6	0/10
Chameleon-f. / fixed hash	5/5	0/10/0/0	4/3/3	2/8
Chameleon-f. / gcn	0/10	1/0/3/6	$3 \cdot 10^{-5}$	.2
Chameleon-f. / resnet	0/10	2/0/4/4	$3 \cdot 10^{-5}$	.2
Actor / adaptive	7/3	0/10/0/0	3/1/6	0/10
Actor / learned constant	7/3	0/10/0/0	2/3/5	0/10
Actor / fixed hash	9/1	0/10/0/0	0/1/9	1/9
Actor / gcn	0/10	1/0/6/3	$3 \cdot 10^{-5}$	.2
Actor / resnet	0/10	10/0/0/0	$3 \cdot 10^{-5}$	.2
Roman-empire / adaptive	0/10	0/10/0/0	0/0/10	0/10
Roman-empire / learned constant	0/10	0/10/0/0	0/1/9	1/9
Roman-empire / fixed hash	10/0	0/10/0/0	1/3/6	0/10
Roman-empire / gcn	0/10	5/0/5/0	$3 \cdot 10^{-5}$	.2
Roman-empire / resnet	0/10	4/0/4/2	$3 \cdot 10^{-5}$	.2
Amazon-ratings / adaptive	0/10	0/10/0/0	0/10/0	0/10
Amazon-ratings / learned constant	0/10	0/10/0/0	0/10/0	0/10
Amazon-ratings / fixed hash	6/4	0/10/0/0	2/3/5	0/10
Amazon-ratings / gcn	0/10	0/0/6/4	$3 \cdot 10^{-5}$	.2
Amazon-ratings / resnet	0/10	10/0/0/0	$3 \cdot 10^{-5}$	.2
Minesweeper / adaptive	0/10	0/10/0/0	5/3/2	9/1
Minesweeper / learned constant	1/9	0/10/0/0	5/3/2	10/0
Minesweeper / fixed hash	6/4	0/10/0/0	0/5/5	4/6
Minesweeper / gcn	0/10	0/0/5/5	$3 \cdot 10^{-5}$	.2
Minesweeper / resnet	0/10	4/0/4/2	$3 \cdot 10^{-5}$	.2

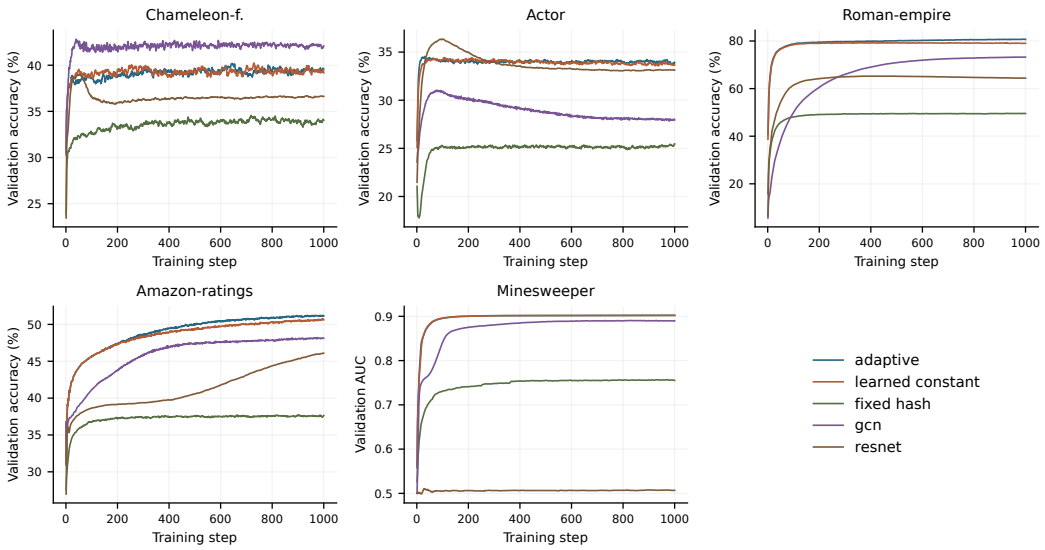

Figure 2: Complete 1,000-step validation curves, averaged over all 30 final cells per graph/arm. The configuration was selected before these final fits; each final checkpoint is separately selected by validation. Curves describe learning under the fixed budget and do not establish convergence or test performance. Minesweeper uses AUC; other panels use accuracy.

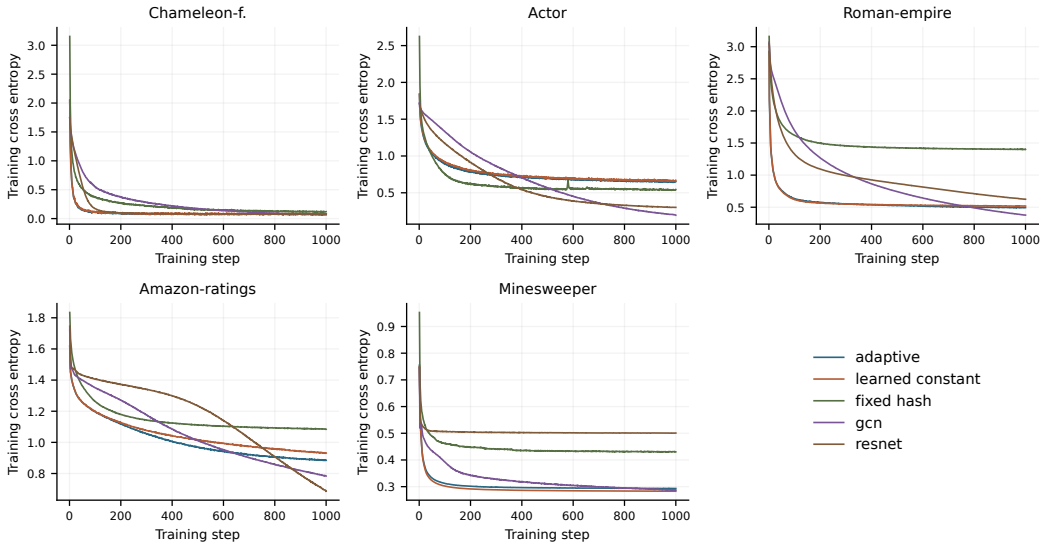

Figure 3: Training cross entropy averaged over all 30 final cells per graph/arm; binary baseline models use equivalent binary cross entropy. These full-budget curves are descriptive diagnostics, not a post-release stopping rule. Training curves and validation spreads are retained numerically in the artifact.

B Counterexample constructions and proofs

We give the details of the two constructions used in Theorem 2. The first starts from expert margins for which uniform routing reproduces the learned logits exactly; a small neighbour contribution then makes the computation coupled without changing its predictions. The second relies on a different mechanism: heterogeneous gates supply two types of messages, whereas a constant gate suppresses one of them. Figure 4 illustrates the latter mechanism on several simple graphs.

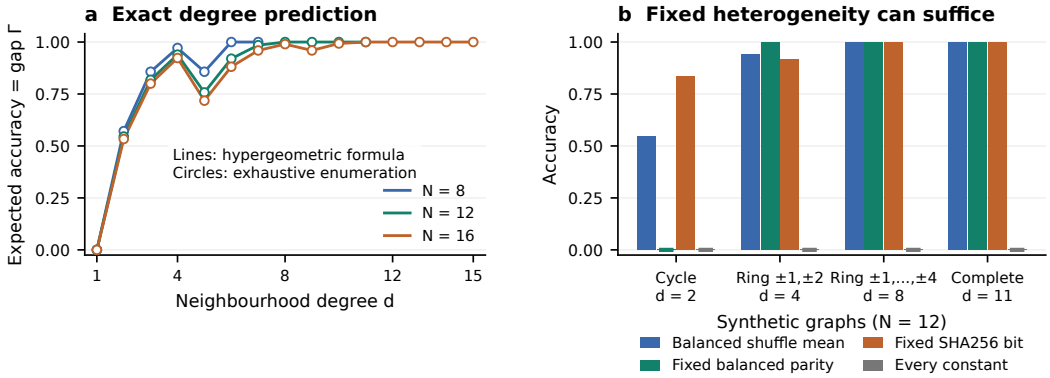

Figure 4: Exact synthetic checks, not trained-network evidence. Lines give Eq. (10); circles give exhaustive balanced-field enumeration. On four $N = 12$ graphs, the balanced-shuffle expectation is compared with fixed parity, a specified unconstrained SHA256-bit assignment, and constants. Degree eight meets the $2N/3$ threshold; every balanced field succeeds there, whereas parity fails on the degree-two cycle.

B.1 Large shuffle drop with exact constant reproduction

Take $N = K \geq 3$, all labels positive, and define binary true-class expert margins

$$a_{vj} = \begin{cases} b, & j = v, \\ b + (N-2)M, & j = v+1 \pmod N, \\ b - M, & \text{otherwise}, \end{cases} \qquad M > b > 0. \tag{11}$$

Realize each margin as canonical binary logits $(a_{vj}, 0)$, with the positive-class coordinate first. One-hot node features and fixed linear expert maps realize this finite matrix. For learned hard gates $q_v = e_v$, each margin is b. Every row averages to b, so the uniform constant reproduces the logits. Each broadcast hard gate is correct at exactly two nodes, hence Theorem 1 gives expected shuffled accuracy $2/N$. The learned and best constant scores are both one, so $\mathrm{ben} = 0$ and $\Delta_{\mathrm{sh}} = 1 - 2/N$. At $N = 20$ this is 0.9, or 90 percentage points. This is not a universal maximal slack result: diagonal margin N with all off-diagonal margins -1 gives shuffled accuracy $1/N$ while a uniform constant remains correct, but then the logits differ.

B.2 A genuinely coupled version

On the complete graph without self-loops, write $a_v(q_v) = \sum_j q_{vj} a_{vj}$ and replace the margin by

$$\widetilde{a}_v(Q) = a_v(q_v) + \frac{\varepsilon}{N-1} \sum_{u \neq v} a_u(q_u), \quad 0 < \varepsilon < \frac{\min(b, M-b)}{b + (N-2)M}. \tag{12}$$

Every gated margin has absolute value at most $H = b + (N-2)M$. Under any hard assignment, every local margin has absolute value at least $\min(b, M-b)$, so the perturbation cannot change its sign. Thus all hard-permutation and learned accuracies are unchanged. The uniform constant gives margin $b(1+\varepsilon) > 0$ everywhere and therefore attains accuracy one, proving case (ii) of Theorem 2. Cross-node dependence is genuine because $\varepsilon > 0$ and neighbours' gates can change their margins. We do not assert sign preservation for arbitrary soft fields; they can approach zero margin.

B.3 Complete-graph construction and degree formula

For Eq. (6), the expert outputs $(1, -1)$ and $(-1, 1)$ are mixed before ReLU. At least one coordinate of a constant emission is therefore zero for every $g \in [0, 1]$. Thus all constant policies fail under the positive-label readout, proving its constant optimum exactly, without a grid-search assumption. Mixing the post-ReLU vectors $(1, 0)$ and $(0, 1)$ instead would be a different construction: its constant

gate $g = 1/2$ would succeed. Balanced hard gates emit the two coordinate basis vectors. Under a uniform permutation, a fixed set of d_v neighbours receives k first-type gates with probability

$$\Pr(X_v = k) = \binom{N/2}{k}\binom{N/2}{d_v - k}/\binom{N}{d_v}. \tag{13}$$

Both neighbour means exceed or equal $1/4$ precisely for the integer interval in Eq. (10). The best constant score is zero, so averaging these probabilities over evaluated nodes gives Γ regardless of the original assignment's score. If the original balanced assignment is perfect, then $\mathrm{ben} = 1$ and $\Delta_{\mathrm{sh}} = 1 - \Gamma$.

For $d_v \geq 2N/3$, there can be at most $N/2$ neighbours of either type, so each type also has at least $d_v - N/2 \geq d_v/4$ representatives. Every balanced assignment is then perfect. In the complete graph $d_v = N-1$, this holds for even $N \geq 4$. The readout is realizable using $\min(a, b) = a - \mathrm{ReLU}(a-b)$ and a binary logit threshold.

C CE IDENTITIES, AFFINE CERTIFICATES, AND LAYERWISE EFFECTS

The ordering in Theorem 1 requires neither linear logits nor a convex loss. When the logits are affine in the replaced gates, cross-entropy permits a more detailed calculation. We first separate the mean-gate comparison from the effect of dispersion, then explain what remains true with nonlinear gates. The final parts of this section concern numerical optimization and combinations of a fixed set of interventions.

C.1 AFFINE CE DISPERSION

For separable affine logits $z_v(q) = Z_v q + b_v$, write $\bar{q} = N^{-1}\sum_j q_j$, $\bar{p}_v = \mathrm{softmax}(Z_v\bar{q} + b_v)$, and $\Pi = I_C - \mathbf{1}\mathbf{1}^\top/C$. For negative CE define $J = F(Q_{\bar{q}}) - \mathbb{E}_P F(PQ)$ and $\Delta_{\mathrm{mean}} = F(Q) - F(Q_{\bar{q}})$. Then

$$\Delta_{\mathrm{sh}} = \Delta_{\mathrm{mean}} + J, \tag{14}$$

$$J = \frac{1}{m}\sum_{v\in T}\mathbb{E}_P \mathrm{KL}\big(\bar{p}_v \,\big\|\, \mathrm{softmax}(Z_v q_{P(v)} + b_v)\big) \geq 0. \tag{15}$$

Let $\Sigma = N^{-1}\sum_j (q_j - \bar{q})(q_j - \bar{q})^\top$. The log-sum-exp Hessian is $\mathrm{diag}(p) - pp^\top$, with Euclidean operator norm at most $1/2$ on centered directions. The integrated Taylor remainder contributes another factor $1/2$, yielding

$$0 \leq J \leq \frac{1}{4m}\sum_{v\in T}\mathrm{tr}(Z_v^\top \Pi Z_v \Sigma). \tag{16}$$

The coefficient $1/4$ is sharp in the small-perturbation limit at balanced binary probabilities, where the Hessian eigenvalues are 0 and $1/2$. Equality $J = 0$ holds exactly when $\Pi Z_v(q_j - \bar{q}) = 0$ for every evaluated v and all j. If $\bar{q} \in \mathcal{C}$, then $\Gamma \leq -J$. Given fixed gates and logits, J is label-invariant; this does not imply that the trained gates or logits carry no task information. These are direct ensemble/Bregman calculations (Banerjee et al., 2005; Wood et al., 2023).

C.2 INTERNAL NONLINEAR GATES

For negative CE and an arbitrary logit map, let $z_v^* = \mathbb{E}_P z_v(PQ)$ and $F^* = m^{-1}\sum_{v\in T} s_v(z_v^*)$. Then

$$J = \underbrace{F(Q_{\bar{q}}) - F^*}_{B} + \underbrace{\frac{1}{m}\sum_{v\in T}\mathbb{E}_P \mathrm{KL}(\mathrm{softmax}(z_v^*)\|\mathrm{softmax}(z_v(PQ)))}_{D\geq 0}, \tag{17}$$

$$|B| \leq \frac{\sqrt{2}}{m}\sum_{v\in T}\|\Pi[z_v(Q_{\bar{q}}) - z_v^*]\|_2. \tag{18}$$

The signed term B measures the mismatch between mean-gate and mean-logit evaluation. It vanishes when the gate field maps affinely to logits; it need not vanish for internal ReLUs. For positive binary labels and logits $((2q-1)_+, 0)$, with $q = 0, 1$ equally likely, $J \simeq -0.190$ nats. Accuracy admits no nonnegative CE-dispersion interpretation.

C.3 What balanced permutations remove

If every node receives every learned row once in a set of permutations, then any separable additive score is integrated exactly by that set. Cyclic shifts provide such a design. For affine logits the empirical mean logits are also exactly $Z_v\bar{q} + b_v$. Ordinary random permutations estimate both mean scores and mean logits unbiasedly, but with sampling variation; nonlinear functions of the estimated mean logits can be biased. These exact-marginal guarantees do not generally integrate coupled computations.

C.4 A restricted optimality certificate

Let $f(c)$ be mean CE of a frozen, separable affine-logit model over a specified evaluation set and a compact convex constant class. It is convex. The Frank–Wolfe gap (Jaggi, 2013)

$$g(c) = \max_{s \in \mathcal{C}} \langle \nabla f(c), c - s \rangle \tag{19}$$

gives $f(c) - g(c) \le f(c^*) \le f(c)$. If f_Q is learned-gate CE on that same set, then the negative-CE advantage is bracketed by

$$f(c) - g(c) - f_Q \le \mathrm{ben} \le f(c) - f_Q. \tag{20}$$

This is an optimization certificate for that objective and evaluation set, not a generalization guarantee. A validation certificate does not certify test optimality, and a CE certificate is not an accuracy certificate. Our terminal-interface search satisfies the affine condition; the first-layer and joint searches do not, so we do not apply this interval to them.

C.5 Binary-factor identity

Replacing two layers separately provides four scores. Writing them as a function of two binary choices makes it possible to distinguish the effect of each replacement from their interaction. For two layer gates encoded by $x_1, x_2 \in \{0, 1\}$, let 0 mean constant and 1 mean learned, so $F_{00} = F_{\mathrm{CC}}$, $F_{10} = F_{\mathrm{LC}}$, $F_{01} = F_{\mathrm{CL}}$, and $F_{11} = F_{\mathrm{LL}}$. Then

$$F(x_1, x_2) = F_{00} + a x_1 + b x_2 + I x_1 x_2, \tag{21}$$

where $a = F_{10} - F_{00}$, $b = F_{01} - F_{00}$, and $I = F_{11} - F_{10} - F_{01} + F_{00}$. Opposing conditional effects can cancel even if $I = 0$. With a common temperature, CE interaction can be nonzero even when $z_{11} - z_{10} - z_{01} + z_{00} = 0$, because log-sum-exp is nonlinear. Using a different fitted temperature in every arm would change the comparison and is not this same-model factorization.

C.6 What a finite intervention family covers

The same question can be asked node by node: if two replacement fields have been evaluated, what follows about a field assembled from their rows? Under separability, the answer depends on whether one retains the individual node scores or only their averages.

Corollary 5 (Coverage of fixed node-wise replacements)**.** *Fix two permitted gate fields Q^0, Q^1. For $b \in \{0,1\}^N$, set $Q(b)_v = Q_v^{b_v}$ and $f(b) = F(Q(b))$. Under separability and an additive node score,*

$$f(b) = f(0) + \sum_{v=1}^{N} b_v \big[f(e_v) - f(0) \big]. \tag{22}$$

Thus baseline and singleton scores determine every combined score in this fixed family. Two endpoint forwards also suffice if their node-wise *scores are retained; their two aggregate scores alone do not.*

Proof. The score at node v is $(1-b_v)s_v(z_v(Q^0))+b_v s_v(z_v(Q^1))$. Sum over T; bits outside T have zero effect. □

This is a structural implication, not a test for separability. It fixes the graph, weights, and two alternatives per node; it does not certify other gate values, unseen inputs, or repeated execution. The following interpolation argument explains why low-order observations alone cannot supply the required restriction. Corollary 5 uses one bit per *node*, choosing its complete tuple from two fixed gate fields. This differs from the preceding two *layer* bits. For node masks, define $a_v^i = s_v(z_v(Q^i))$. Separability gives the stronger pointwise identity $s_v(z_v(Q(b))) = a_v^{b_v}$, so retaining both endpoint score vectors reconstructs every mixed score without additional forwards. Accuracy and negative CE satisfy this statement despite their nonlinearities, since each node term depends on only one bit. Aggregate endpoint scores discard the node-wise allocation and do not suffice. This observation is not a new verification algorithm; its validity comes from the independently justified intervention interface.

For a general fixed family with d binary intervention choices, every scalar discrepancy δ has a unique multilinear representation

$$\delta(b) = \sum_{S\subseteq[d]} c_S \prod_{i\in S} b_i, \qquad c_S = \sum_{U\subseteq S} (-1)^{|S|-|U|}\delta(\mathbf{1}_U). \tag{23}$$

Here δ may compare a measured score with a specified reference score on the same masks. If its multilinear degree is independently known to be at most r, and $\delta(\mathbf{1}_U) = 0$ for all $|U| \le r$ (including the empty mask), then δ vanishes everywhere: the displayed inversion kills coefficients through order r, and the degree restriction kills the rest. These are standard interpolation facts, not a new generalization theorem (Sundararajan et al., 2020).

Without that restriction, for $0 \le r < d$, $\delta(b) = \prod_{i\in S} b_i$ with $|S| = r+1$ is zero on every tested mask of weight at most r but nonzero on $\mathbf{1}_S$. This is a fixed-family coverage obstruction, not a neural-architecture realization or a lower bound on finding failures with unrestricted tests. Successful low-order checks do not certify low degree; neither does network depth alone. The degree here counts mask interactions, not the graph degree in Proposition 4. We establish no useful degree certificate for the internal nonlinear GNN in the audit.

The earlier custom-model four-arm layer audit covers all 2^2 choices for its declared checkpoint and selected constants, with no interpolation assumption. It certifies neither other constants nor behavior on other graphs or repeated executions, and is distinct from the new study's three intervention interfaces. Scalar agreement can also hide node-wise or logit changes. A deterministic validation script checks fixed-node reconstruction, the insufficiency of aggregate endpoints, Boolean inversion, and the higher-order obstruction on finite examples; these checks supplement the proofs and do not certify trained networks.

D HISTORICAL RESTRICTED-RECIPE AUDIT IMPLEMENTATION

This section and the historical result tables below describe the earlier study. They are retained for provenance, are not the stronger-recipe experiment in Section 5, and are not pooled with it. The main study's protocol appears in Appendix A.

Original custom-model implementation. Each expert is a learned affine map of neighbour mean, feature-minus-neighbour-mean, or unaggregated features, respectively. The model has two width-64 layers, three experts per layer, and the label-free router configuration. Gates mix expert outputs before a ReLU at each layer; a final linear classifier maps the second layer to class logits. The configured dropout probability is 0.35, disabled throughout these evaluation-mode interventions. Features are shifted by their global minimum and divided row-wise by the row sum clamped below at one; undirected edges are deduplicated. Original masks and router-feature conventions are retained. Actor's sparse features are binary and indexed by original node ID. Roman-empire retains the historical large-graph convention that sets the clustering router feature to zero; it was not silently replaced during this audit.

The 225 initial grid pairs use two three-expert simplexes. Fourteen Adam starts comprise the uniform and mean gates, nine near-vertex pairs (permutations of $(0.98, 0.01, 0.01)$), and three Dirichlet starts.

Each trajectory contributes the initial point and 200 updates at learning rate 0.05. Candidate count is therefore $225 + 2 + 14 \cdot 201 = 3041$. The accuracy selector chooses the largest validation correct count in this same pool, with validation CE and first occurrence as tie breakers. Test labels are not supplied to either selector. The test-score superiority statement concerns the two selected policies, not all 3,041 candidates.

One deterministic node permutation is shared across the two layers in each shuffle, preserving the joint tuple of learned layer gates. The bank has 100 permutations per checkpoint, including replay of the original 20. These are conditioned-on empirical observations in the main text. Theoretical expectation statements refer to uniform permutations; no unreported Monte Carlo confidence interval is attached to the recorded bank.

Separate validation temperatures are fitted after policy selection, with inverse temperature constrained to $[0.01, 100]$. Accuracy is unaffected by positive temperature scaling. Arm-calibrated CE compares policies, while the algebraic CE decomposition and layerwise primary CE comparison use a common temperature. The search and temperature fitting do not jointly optimize the gate and temperature.

Original verification recomputed 91,230 stored candidate objectives per dataset bank, as well as saved metrics and integrity checks. The original Torch implementation remains the neural-forward/gradient reference. Our final asset verification re-aggregates saved cell scores, checks them against the retained summaries, and hashes its inputs. Neither operation reconstructs every training or Adam trajectory independently. Construction validation and plot reproduction are separate from neural-checkpoint reproduction.

Official ACM-GCN matched training and frozen consumer. The official ACM-GCN implementation is pinned at Git commit `75b4853f29b2ce0844283e4a64c4b1120feb295c`. Its low/high/identity channels, native ReLUs, threefold gate scale and both attention interfaces remain unchanged. Across three datasets, ten original splits, five predeclared arms and 12 HPO configurations there are 1,800 tuning fits; the validation-selected configuration for each dataset/split/arm is then trained under three final seeds, giving 450 final fits. The common hidden width is 64. The 12 configurations cross learning rates $\{0.01, 0.05\}$, weight decays $\{0, 0.0005, 0.001\}$ and dropouts $\{0, 0.5\}$. Tuning uses seed 1601 and final fits use seeds 1701–1703. Each fit has at most 500 epochs, a minimum stopping epoch of 20, patience 50 and validation-CE improvement threshold 10^{-12}; the minimum stopping epoch is distinct from the saved best-checkpoint epoch. Expert initializations and HPO/training budgets are matched for adaptive, learned-constant and fixed-hash arms. GCN and MLP have distinct active parameter sets and serve as contextual baselines. The original split masks and node order are retained; test labels are redacted from every optimization worker bundle.

For each selected adaptive checkpoint, a native-hook consumer records both learned gate fields and final expert channels. Joint constants lie in $\Delta_3 \times \Delta_3$; final-only constants lie in Δ_3. The 3,169 joint candidates comprise 225 grid pairs (15×15), uniform and node mean, 14 trajectories of 201 observed points, and 128 uniformly sampled learned tuples broadcast across the graph. A separate SHA256-derived PCG64 stream controls each auxiliary sampling bank. The original permutation seed is $710000+1000d+10s+i$, with dataset index $d \in \{0, 1, 2\}$, split $s \in \{0, \ldots, 9\}$ and final-seed index $i \in \{0, 1, 2\}$; the same 100 full-node permutations are used at both interfaces. Joint shuffles preserve each source node's two-layer gate tuple. Candidate gates and IDs, validation CE/correct counts, selected policies, FW iterates, permutation indices, and SHA256 provenance are retained. Accuracy selection is a sensitivity over the CE-generated pool, not a test-driven search.

For the final-only interface, fixing the native first layer gives $z_v(c) = 3 \sum_{k=1}^{3} c_k C_{vk}$. We minimize its FP64 validation CE from uniform by Frank–Wolfe with 2,000 maximum updates, 10^{-6} gap target and 50 derivative-bisection steps for an interior line minimum. Its dual gap gives a convex validation-CE interval, not a test or accuracy guarantee. The final-only selected control pool comprises the 15 grid points, uniform, mean, 128 broadcast source rows and all recorded FW iterates. The final-only native-forward and direct-mixture logits agree on canaries. For the full Chameleon coupled diagnostic, every learned source tuple is broadcast to all recipients; the paired count identity is checked using original node IDs and the same 100 shuffles. A final-only census gives exactly zero class-level paired replacements.

D.1 AUDIT CHRONOLOGY

The original three-graph resampling calculation was prescribed before its held-out release and implemented afterward. Extension policies and prediction banks were also fixed before scoring. The detailed extension chronology, including the post-release CE reconstruction and AUC-selected sensitivity, is in Appendix H.3.

The historical producer archive, checkpoint outputs and restricted bundles were hash-audited. An early consumer attempt failed exact replay and was excluded before held-out release. With matched producer settings, all 90 adaptive checkpoints replayed exactly; the corrected run fixed all validation selections and prediction files before scoring. A second scorer rechecked the historical matched/frozen test conditions. Three complete example candidate pools were independently rescored on validation, while every selected policy and output hash was checked. Detailed execution records remain in the companion packet. No held-out outcome changed a search budget or selector.

E COUPLING DIAGNOSTICS AND SAMPLING SCOPE

Zero residual with maximal changes. For two adjacent positive-label nodes, let $z_v(Q) = (2g_u - 1, 0)$ where u is the other node and $Q = (0, 1)$. Under the identity permutation, shuffled correctness is $(1, 0)$, whereas the paired own-gate broadcasts have correctness $(0, 1)$; the swap reverses both vectors. Every permutation has $G_M = L_M = 1/2$, $R_M = D_M = 1$, $\mu_M = \bar{b} = A = 1/2$, and $U_M = \eta_M = 0$. Conversely, logits $(2 + g_u, 0)$ are genuinely coupled while predictions remain correct for every allowed field. No zero accuracy-level diagnostic certifies logit separability.

Finite-bank interpretation. For a structurally separable network $a_{iv} = b_{iv}$ pointwise, so $R_M = D_M = 0$ and $U_M = \eta_M$. A finite unbalanced permutation bank therefore need not give $U_M = 0$ even under separability. Write the evaluation set in order as $T = (t_1, \ldots, t_m)$. With broadcast correctness stored as an $N \times m$ array C, the paired indicator is $b_{i,t_\ell} = C_{P_i(t_\ell),\ell}$. Evaluation positions must not replace original graph node IDs. The identity alone cannot detect a consistently misindexed b, so the validator tests keyed node-ID alignment separately. All layers of a source node are broadcast jointly. Pair counts are not independent statistical sample sizes.

Proposition 6 (Off-node influence)**.** *Write $r_v(Q) = s_v(z_v(Q))$. Suppose uniformly over allowed fields agreeing at v,*

$$|r_v(Q) - r_v(Q')| \le \sum_{u \ne v} L_{vu} \|q_u - q'_u\|. \tag{24}$$

For $d_{\mathrm{pair}} = \{N(N-1)\}^{-1} \sum_{i \ne j} \|q_i - q_j\|$,

$$|U| \le B_{\mathrm{cross}} := \frac{d_{\mathrm{pair}}}{m} \sum_{v \in T} \sum_{u \ne v} L_{vu}, \qquad \Gamma \le B_{\mathrm{cross}} - S. \tag{25}$$

Couple PQ with broadcasting $q_{P(v)}$ at node v. For $u \ne v$, $(P(u), P(v))$ is uniform over ordered distinct indices; averaging the assumed inequality proves the claim. In a Lipschitz one-layer aggregator, L_{vu} can incorporate adjacency weights. Accuracy requires a margin argument, not a global Lipschitz assumption. We do not claim this bound is numerically useful for the audited model.

Broadcast sampling. If a sample of $B < N$ broadcast scores is drawn uniformly without replacement, its mean $\widehat{A}_B$ has estimated standard error $s_B \sqrt{(1 - B/N)/B}$, with sample variance s_B^2 using denominator $B-1$. A complete census has zero broadcast-sampling error. Neither quantity accounts for uncertainty in μ_M or graph-population generalization. Writing $e_B = \widehat{A}_B - A$, Eq. (27) gives $\widetilde{U}_{M,B} = U_M - e_B$ and $\widetilde{S}_B = F(Q_{\hat{c}}) - A - e_B$. Their difference cancels e_B exactly; a zero residual is an algebra check, not independent validation of A. If $A \le A_{\mathrm{upper}}$ is independently justified, then $S \ge \max\{0, F(Q_{\hat{c}}) - A_{\mathrm{upper}}\}$. No such confidence bound is inferred from the reported standard error. The retained raw-CE coupling files use loss differences; higher-is-better negative-CE residuals require the opposite sign.

F CONDITIONAL EXPECTED-SHUFFLE CERTIFICATE

Proposition 7 (Conditional simultaneous upper bound)**.** *Declare a family of H dataset/interface comparisons and a failure probability $\alpha \in (0,1)$. Fix the graphs, labels, 30 selected checkpoints per dataset, their gate fields and their validation-selected constant policies. For one comparison, let $X_{j,r} \in [0,1]$ be whole-graph test accuracy from uniform gate permutation $r \in \{1,\dots,100\}$ at checkpoint $j \in \{1,\dots,30\}$. Assume these $n = 3000$ permutation draws are independent conditional on the fixed objects; their score distributions need not be identical. Let $\bar{X} = n^{-1}\sum_{j,r} X_{j,r}$ and $V = (n-1)^{-1}\sum_{j,r}(X_{j,r} - \bar{X})^2$. Let C_j be the score of the found constant, and S_j^* the maximum over the declared constant class, so $C_j \le S_j^*$. Put*

$$\bar{\Gamma} = \frac{1}{30}\sum_{j=1}^{30}\left(\mathbb{E}_r X_{j,r} - S_j^*\right), \qquad L = \log\frac{2H}{\alpha}.$$

With conditional probability at least $1-\alpha$, simultaneously for all H comparisons,

$$\bar{\Gamma} \le \bar{X} - \frac{1}{30}\sum_j C_j + \sqrt{\frac{2VL}{n}} + \frac{7L}{3(n-1)}. \tag{26}$$

This is an upper bound on the optimal-*constant composition gap; a negative upper bound certifies its negative sign under the conditional sampling assumption.*

Proof. Apply the one-sided empirical Bernstein inequality for independent bounded, not necessarily identically distributed variables (Maurer & Pontil, 2009, Theorem 11) to each comparison with failure probability α/H. Since each checkpoint contributes 100 draws, $\mathbb{E}\bar{X} = 30^{-1}\sum_j \mathbb{E}_r X_{j,r}$. The theorem bounds this expectation above by $\bar{X} + \sqrt{2VL/n} + 7L/[3(n-1)]$. Subtract $30^{-1}\sum_j S_j^* \ge 30^{-1}\sum_j C_j$, then union-bound the H events. No independence between comparisons is needed. □

The displayed results multiply Eq. (26) by 100 to express percentage points and round each reported upper bound *upward*. They are conditional on ideal independent uniform permutations; distinct pseudorandom seeds alone do not prove physical independence. Shared graphs, overlapping splits and fixed checkpoint choices do not enter the sampling unit and hence receive no population-level coverage claim. The new study declares $H = 15$ and $\alpha = .05$ before scoring. Historical $H = 6$ and $H = 4$ results remain separately labeled families, not a pooled simultaneous guarantee. Validation-CE optimality intervals are not accuracy or test-optimality certificates, and the bounded-accuracy inequality does not directly cover unbounded cross-entropy.

G HISTORICAL EXPLORATORY DESCRIPTIVE TABLES

G.1 EARLIER CUSTOM-MODEL EVIDENCE

What the earlier custom-model audit adds. Its mean joint shuffle drops are 4.55, 1.81, and 29.52 points, versus 0.15, -0.01, and 19.53 against validation-CE-selected constants. This larger magnitude did *not* replicate in the earlier restricted-recipe ACM-GCN study, particularly on Chameleon and Actor. It does show that the same inference error can arise under different routing implementations. The older Roman learned gates exceed both found constants in all 30 cells; the restricted-recipe ACM result was weaker and must not be pooled with it. Neither comparison describes the stronger-recipe results. The custom-model tables and layerwise analysis follow below.

Paired coupling and finite broadcast samples. The earlier custom-model audit measures every learned source tuple broadcast on Chameleon and Actor, but samples $B = 2000$ of $N = 22662$ sources on Roman. If $\widehat{A}_B$ is that broadcast average, its sampled quantities obey

$$\widetilde{U}_{M,B} = \mu_M - \widehat{A}_B, \quad \widetilde{S}_B = F(Q_{\hat{c}}) - \widehat{A}_B, \quad \widehat{\Gamma}_M = \widetilde{U}_{M,B} - \widetilde{S}_B. \tag{27}$$

The shared sampling error cancels in the last identity but not in either component. In that older Chameleon audit, a small signed mean residual coexists with nonzero paired correctness changes

on every checkpoint ($R_M = 3.139\%$, $D_M = 7.363\%$). The predeclared *historical width-64 ACM* Chameleon census is a useful negative control: mean replacement and class-change rates are only 0.016 and 0.048 percentage points, and the final-only replacement rate is exactly zero in all 30 cells, as separability predicts. Thus coupling is possible and measurable, but its magnitude is architecture- and checkpoint-specific. Neither a small ACM residual nor the older cancellation result proves a positive-Γ trained-network extreme; that extreme remains synthetic.

The high raw CE of the custom Chameleon model also limits interpretation of its magnitude: the learned model has approximately 3.95 nats, compared with $\log 5 \simeq 1.61$ for uniform class probabilities. Poor calibration or overconfident errors can produce this combination of accuracy and loss. Separate validation-fitted temperatures improve the calibration comparison but do not repair the raw-score model or turn it into an independent replication. We retain these results for transparency, not as evidence of competitive model quality.

Table 13: Simultaneous 95% conditional expected-shuffle certificates (percentage points). Negative upper bounds certify overstatement of optimal constant-class advantage; Actor remains inconclusive. The six comparisons use a Bonferroni family and 3,000 whole-graph permutation scores each.

Dataset	Joint $\overline{\Gamma}$ upper	Final-only $\overline{\Gamma}$ upper	Conclusion
Chameleon-filtered	-2.23	-0.78	overstatement certified
Actor	+0.41	+0.33	inconclusive
Roman-empire	-0.96	-0.19	overstatement certified

Table 14: Predeclared frozen-checkpoint interpolation. V_1, V_2 are all-node RMS gate variations; Δ is mean test accuracy at broadcast node-mean gates ($t = 0$) minus learned gates ($t = 1$), with one cell-level SD over 30 fixed cells.

Dataset	V_1	V_2	Joint t=0 Δ (pp)	Final t=0 Δ (pp)
Chameleon-filtered	0.0033	0.0094	+0.07 $\pm$ 0.35	+0.02 $\pm$ 0.23
Actor	0.0026	0.0196	+0.01 $\pm$ 0.12	+0.01 $\pm$ 0.13
Roman-empire	0.1649	0.0702	-1.57 $\pm$ 0.75	-1.02 $\pm$ 0.61

Table 15: Original custom-model frozen audit: mean held-out accuracy (%) on 30 historical checkpoints per dataset. These are not the ACM-GCN results.

Frozen gate policy	Chameleon	Actor	Roman-empire
Learned	31.52	34.71	41.27
Shuffled (100 samples)	26.98	32.90	11.75
Mean gate	27.45	34.22	16.27
Uniform gate	28.41	30.77	13.58
Constant (validation CE)	31.37	34.72	21.74
Constant (validation accuracy)	32.70	34.86	30.94
Train-majority label	24.99	25.24	13.96

Table 16: Original custom-model broadcast decomposition. Roman uses a source sample, not a census. These measurements must not be pooled with either ACM-GCN study.

Dataset	Broadcasts	$\widetilde{U}$	Range	> 0	$\widetilde{S}_{\mathrm{Acc}}$
Chameleon	890	-0.04	[-0.96, +0.82]	14/30	+5.68
Actor	7,600	-0.08	[-0.57, +0.17]	9/30	+1.88
Roman-empire	2,000/22,662	-1.88	[-3.75, +0.67]	1/30	+17.30

Table 17: Paired learned-minus-selected-constant accuracy differences: descriptive spread and counts, not equivalence tests or independent-binomial evidence.

Dataset	Selector	Mean	SD	Range	$+/-/0$	$\lvert\Delta\rvert \leq 1$	$\lvert\Delta\rvert \leq 2$
Chameleon	CE	+0.15	4.02	[-5.91, +10.11]	11/17/2	6	12
Chameleon	Accuracy	-1.18	3.82	[-8.07, +5.49]	10/19/1	4	8
Actor	CE	-0.01	1.11	[-2.17, +2.43]	14/14/2	20	27
Actor	Accuracy	-0.15	1.15	[-2.30, +2.24]	10/19/1	18	27
Roman-empire	CE	+19.53	4.79	[+11.60, +28.77]	30/0/0	0	0
Roman-empire	Accuracy	+10.34	3.17	[+5.24, +14.88]	30/0/0	0	0

Table 18: Chameleon four-condition means for both validation selectors. Constants are unchanged across hybrids.

Constant selector	CC	CL	LC	LL	Interaction (pp)
CE	31.369	33.895	29.205	31.522	-0.209
Accuracy	32.702	34.757	28.549	31.522	+0.917

Table 19: Raw and arm-calibrated CE, in nats (lower is better). Each calibrated arm receives its own validation-fitted temperature; these entries are not a shared-temperature decomposition.

Frozen gate policy	Cham. raw	Cal.	Actor raw	Cal.	Roman raw	Cal.
Learned	3.9486	1.5534	1.5057	1.4347	1.8436	1.8412
Shuffled	3.9745	1.5876	1.5543	1.4725	3.9876	2.8438
Mean	3.6452	1.5851	1.4936	1.4498	2.6923	2.6287
Uniform	3.0161	1.5741	1.5908	1.4965	2.7081	2.6390
Const. (CE)	2.3039	1.5364	1.4631	1.4337	2.5910	2.5876
Const. (accuracy)	2.7886	1.5416	1.4869	1.4370	2.6390	2.6133

Table 20: Paired Chameleon diagnostic: 30 historical checkpoints, 100 permutations each. Gains, losses, and change rates are percentages of evaluated node/permutation pairs within a cell; signed differences are percentage points. Means weight checkpoints equally, not pairs globally. Marginal imbalance is retained rather than silently set to zero.

Metric	Mean	Cell range
Correctness gains G	1.549	[0.145, 3.230]
Correctness losses L	1.590	[0.119, 3.029]
Changed correctness R	3.139	[0.264, 5.878]
Changed class D	7.363	[0.604, 13.773]
Signed paired change V	-0.040	[-0.797, 1.006]
Marginal imbalance η	-0.000404	[-0.363, 0.335]
Observed gap U_M	-0.041	[-0.956, 0.821]

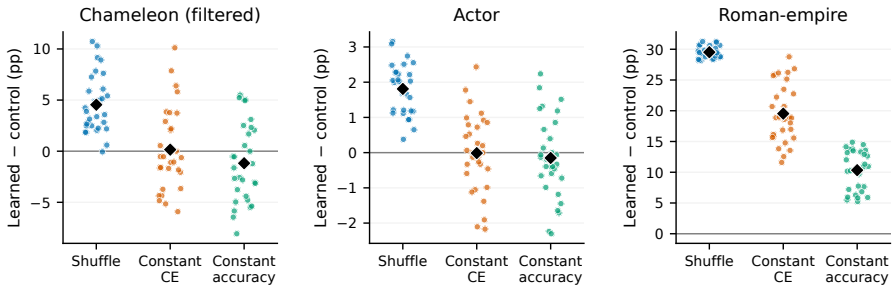

Figure 5: Original custom-model frozen contrasts. The much larger magnitudes, especially on Roman, are not reproduced by the historical width-64 ACM-GCN checkpoints. Panel y-scales differ.

H Historical width-64 ACM-GCN detailed results

H.1 Cell-level descriptive spread

Table 21: Equal-cell mean and sample SD across 30 split-seed cells for every arm and their paired adaptive-minus-constant difference. Accuracy rows show mean ± SD in percent/percentage points; AUC mean and SD occupy separate rows on the 0–1 scale. SD describes dispersion across overlapping splits and seeds, not standard error, an independent-graph confidence interval, or an equivalence test. This is a post-release descriptive aggregation of the unchanged 750-model scores.

Dataset / metric	Adaptive	Const.	Hash	GCN	MLP	Paired Δ
Chameleon (acc.)	33.5±2.9	33.3±2.8	29.9±2.8	38.4±3.0	30.8±3.7	+0.16±1.35
Actor (acc.)	35.1±1.0	35.1±1.1	26.5±1.1	29.9±1.1	36.0±1.1	−0.03±1.13
Roman-empire (acc.)	61.6±2.4	60.5±0.9	24.0±2.7	29.2±1.1	60.9±2.3	+1.13±2.69
Amazon-ratings (acc.)	31.5±1.3	30.5±1.5	27.2±0.3	40.3±1.0	33.3±1.0	+1.02±1.62
Minesweeper (AUC mean)	0.9062	0.9063	0.7596	0.7231	0.5080	-0.00012
Minesweeper (AUC SD)	0.0057	0.0059	0.0117	0.0088	0.0105	0.00118
Minesweeper (acc.)	86.0±0.7	86.1±0.7	80.5±0.5	80.2±0.2	80.0±0.0	−0.07±0.30

H.2 Published benchmark context

Table 22: Earlier width-64 study versus published performance, *not* the new stronger-recipe study or a matched literature comparison. Earlier means are rounded from the retained historical table. Literature means are from Tables 2 and 4 of Platonov et al. (2023); the published ResNet is not the earlier MLP architecture. All entries are on a 0–100 scale.

Graph (metric)	Earlier GCN	Published GCN	Earlier MLP	Published ResNet
Chameleon-filtered (accuracy)	38.4	40.89	30.8	36.73
Roman-empire (accuracy)	29.2	73.69	60.9	65.88
Amazon-ratings (accuracy)	40.3	48.70	33.3	45.90
Minesweeper (100×AUC)	72.31	89.75	50.80	50.89

The published models use width 512, residual connections, layer normalization and post-aggregation MLPs, with a depth search from one to five layers and 1,000 training steps. The earlier fixed two-layer width-64 recipe is narrower and is not architecture- or compute-matched to those models. The gap does not by itself identify an implementation error, but it shows why matching budgets within that earlier study was insufficient evidence of competitive absolute performance. The new study in Section 5 instead retunes all routed and static arms under a common declared routing-arm budget, with separately trained residual context baselines. It does not substitute published means for new measurements.

Table 23: Paired adaptive-minus-trained-constant accuracy difference and descriptive 2.5–97.5 percentiles from 10,000 hierarchical split/seed resamples (PCG64 seed 917001). Historical splits overlap; these are not independent-graph confidence intervals or equivalence tests.

Dataset	Δ (pp)	2.5–97.5 percentiles (pp)
Chameleon	+0.16	[-0.45, +0.75]
Actor	-0.03	[-0.49, +0.45]
Roman-empire	+1.13	[-0.53, +2.41]

Table 24: Historical width-64 ACM-GCN held-out accuracy (%) and CE (nats/node, lower is better), equal-cell means. Each model entry is accuracy/CE. Majority is selected from training labels separately for each split; as a hard-only predictor it has no declared finite CE.

Dataset	adaptive Acc/CE	learned constant Acc/CE	fixed hash Acc/CE	gcn Acc/CE	mlp Acc/CE	Majority Acc
Chameleon	33.46/1.467	33.30/1.468	29.86/1.557	38.40/1.415	30.83/1.587	24.99
Actor	35.06/1.428	35.09/1.429	26.52/1.586	29.92/1.539	36.04/1.414	25.24
Roman-empire	61.65/1.560	60.52/1.669	24.00/2.767	29.24/2.346	60.88/1.729	13.96

Table 25: Historical width-64 ACM-GCN parameters per selected final fit, shown as a single value or minimum–maximum over 30 cells. Structural active counts follow the executed forward path; observed-gradient counts were recorded after backpropagation; allocated and requires_grad counts describe stored tensors and optimizer eligibility. These definitions need not coincide.

Dataset	Arm	Structural active	Observed gradient	Allocated	requires_grad
Chameleon	adaptive	447585	447585	509756	509756
Chameleon	learned_constant	447366	447366	509762	447366
Chameleon	fixed_hash	447360	447360	509756	447360
Chameleon	gcn	149120	149120	149120	149120
Chameleon	mlp	149120	149120	149120	149120
Actor	adaptive	180129	180129	705290	705290
Actor	learned_constant	179910	179910	705296	179910
Actor	fixed_hash	179904	179904	705290	179904
Actor	gcn	59968	59968	59968	59968
Actor	mlp	59968	59968	59968	59968
Roman-empire	adaptive	61320	61320	1920508	1920508
Roman-empire	learned_constant	61062	61062	1920514	61062
Roman-empire	fixed_hash	61056	61056	1920508	61056
Roman-empire	gcn	20352	20352	20352	20352
Roman-empire	mlp	20352	20352	20352	20352

Table 26: Both historical width-64 ACM-GCN interfaces, equal-cell mean test differences. Shuffle, CE gap, and Acc gap are learned minus the indicated control in percentage points; $\widehat{\Gamma}$ is shuffle mean minus the accuracy-selected constant. Final two columns are control minus learned CE in nats/node. Selectors were fixed from validation.

Dataset	Interface	Shuffle	CE gap	Acc gap	$\widehat{\Gamma}$	CE sel. ΔCE	Acc sel. ΔCE
Chameleon	joint	-0.05	-2.77	-2.88	-2.83	-0.026	-0.000
Chameleon	final-only	-0.04	-1.01	-1.42	-1.38	-0.004	+0.014
Actor	joint	-0.00	-0.31	-0.08	-0.08	-0.004	+0.008
Actor	final-only	-0.01	-0.13	-0.17	-0.16	-0.002	+0.007
Roman-empire	joint	+2.61	+1.63	+1.07	-1.53	+0.194	+0.225
Roman-empire	final-only	+1.45	+1.07	+0.72	-0.74	+0.056	+0.082

Table 27: Historical width-64 ACM-GCN final-affine Frank–Wolfe validation CE gaps (nats/node), 30 checkpoints per dataset. Unmet gap tolerance is reported rather than called converged. These intervals do not cover joint nonlinear optimality or held-out accuracy.

Dataset	Median gap	Maximum gap	$\leq 10^{-6}$/30
Chameleon	9.59e-07	4.83e-05	22
Actor	6.30e-06	4.12e-05	12
Roman-empire	7.69e-07	9.77e-07	30

The allocated/active parameter difference comes partly from upstream per-node structural tensors registered as parameters but unused by the executed ACM-GCN branch. On Roman, the two `struc_low` tensors dominate this allocation gap; their storage does not provide extra active computation. Structural-active and observed-gradient counts, rather than allocated counts alone, describe the participating model. We retain the upstream tensors and checkpoints without reinterpreting them as active capacity.

The joint accuracy-selected learned-minus-found-constant outward-rounded ranges are $[-8.70, 3.15]$, $[-1.78, 0.93]$, and $[-0.27, 2.69]$ points on Chameleon, Actor, and Roman, respectively. Their positive-cell counts are 5, 17, and 28 of 30. For Roman, $\widehat{\Gamma}_{\mathrm{Acc}} < 0$ in all 30 cells (range $[-4.14, -0.32]$ points); the corresponding finite-bank overstatement is robust across the recorded split-seed cells, though not a population-permutation interval. The primary matched-training gap ranges are $[-2.52, 2.75]$, $[-2.37, 2.04]$, and $[-5.84, 4.57]$ points. The Roman split means are $3.29, 1.83, 1.21, 1.89, 2.02, 1.36, 0.89, -5.15, 0.60, 3.32$ points in split order. Reporting the negative split prevents the mean from obscuring instability.

H.3 Extension chronology and post-release sensitivity analyses

The historical Amazon-ratings and Minesweeper extension reused the pinned official implementation, five arms, ten native splits, 12 validation-CE-selected HPO configurations, tuning seed 1601 and three final seeds. Its 1,200 HPO fits, 300 finals and 60 adaptive frozen cells were completed on deterministic CPU sparse operators. All 300 selected checkpoints replayed exactly. The validation policies and 100-permutation prediction banks were fixed before the held-out release; exact timestamps are in the companion provenance record. Its four-comparison conditional 95% upper bounds on expected accuracy Γ were, in Amazon joint/final and Minesweeper joint/final order, $+0.510, +0.470, +0.188, +0.368$ percentage points (rounded upward). All are inconclusive, and this $H = 4$ family is separate from the earlier $H = 6$ and new $H = 15$ families.

The original extension release omitted a validation-AUC-selected sensitivity constant for Minesweeper and retained only argmax classes for the 12,000 shuffled evaluations across two interfaces. After test exposure we evaluated *exactly the same* checkpoints, gate candidate IDs and saved permutations. Every regenerated shuffled class matched its original sealed counterpart; a separate implementation recomputed all 12,000 CE draws and the 60 AUC-selected controls. The post-release AUC control was chosen on validation by highest margin-score AUC, then lower validation CE and candidate ID, from the unchanged CE-generated pool. These are post-release analyses, not retrospectively preregistered controls or fresh confirmation. The historical Minesweeper primary AUC follows the original float32 softmax-positive-probability evaluator; logit-margin AUC is retained separately, with at most 0.000065 difference across the 300 matched-model cells from finite-precision ties. AUC does not satisfy the additive-score theorem.

Table 28: Historical extension loss audit in nats per node, higher-is-better $F = -\mathrm{CE}$. Trained CE gain is trained-constant CE minus adaptive CE. The other three columns use frozen validation-CE-selected constants and the original 100 permutations. Means are over 30 split-seed cells. All loss contrasts were computed after the original test release from fixed artifacts.

Dataset	Interface	Trained CE gain	Shuffle drop	Frozen found gain	$\widehat{\Gamma}_M$
Amazon-ratings	joint	+0.0643	+0.0218	-0.0016	-0.0234
Amazon-ratings	final-only	+0.0643	+0.0152	+0.0335	+0.0183
Minesweeper	joint	+0.0002	+0.0149	+0.0065	-0.0084
Minesweeper	final-only	+0.0002	+0.0023	+0.0009	-0.0014

On Amazon, paired training CE gain is $+0.0643$ nats with 20/30 positive cells, but adaptive accuracy remains below the majority and GCN baselines. Split-0 signed row-sum normalization gives a maximum absolute feature magnitude above 5,298. The final gate has mean across-class node-wise standard deviation only 0.00991 over the 30 checkpoints. Replacing it by its node mean changes no hard predicted class anywhere in those checkpoints but changes logits and CE; zero final-only accuracy contrast therefore does not imply identical computation. Near-zero signed row sums can amplify feature magnitudes; finite values alone do not establish suitable preprocessing. This is an unresolved recipe-quality limitation, not an experimentally corrected result. No result-driven normalization or selective cell replacement was applied. Minesweeper paired training CE gain is $+0.00018$ nats, with its descriptive split/seed interval $[-0.00048, +0.00091]$. The supplemental AUC-selected constant scores mean test margin AUC 0.90304 jointly and 0.90612 final-only; the learned model scores approximately 0.90618. These are finite-pool post-release sensitivities, not certificates of constant-class optimality. The package retains all cells, sign counts, spreads, score vectors, selectors and exposure chronology; overlapping splits are not independent graphs.

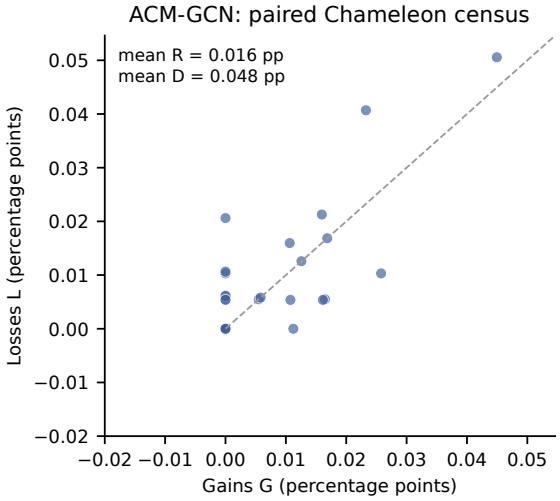

Figure 6: Historical width-64 ACM-GCN joint paired Chameleon census: 100 permutations, every learned source tuple broadcast to every recipient, on 30 checkpoints. Gains and losses are percentages of evaluated permutation-node pairs; replacement $R = G + L$ averages just 0.016 percentage points and class changes D average 0.048. Nonzero coupling is measurable but far smaller than in the custom-model audit.