CLOSED-FORM NODE CLASSIFICATION WITH EXACT GRAPH UNLEARNING

A PREPRINT

Aditya Gaur*
Machine Learning Lab
IIIT Hyderabad
aditya.gaur@students.iiit.ac.in

Charu Sharma
Machine Learning Lab
IIIT Hyderabad
charu.sharma@iiit.ac.in

ABSTRACT

Graph neural networks for node classification are typically trained by gradient descent over hundreds or thousands of epochs. Recent work has shown that, when properly tuned, classic GCN/SAGE/GAT architectures can match graph transformers on many node-classification benchmarks. We ask a complementary question: how much of this performance can be recovered by deterministic closed-form solvers, and what guarantees does this enable?

We introduce a routed closed-form framework selected by adjusted homophily. For assortative graphs, we use SGC-style propagation followed by Ridge regression; for heterophilous graphs, we introduce LCF-Net, a layer-wise closed-form graph feature-refinement network whose per-layer Ridge solves are capped by a Gaussian kernel-Ridge head. Across 14 benchmarks, including ogbn-arxiv and ogbn-proteins, our closed-form predictors match or beat the best vanilla 2-layer GCN/SAGE/GAT on 9 of 9 measured datasets, tie tuned deep recipes within one standard deviation on 9 of 12 small benchmarks, and exceed the OGB-leaderboard plain GCN on both large graphs. The remaining heterophilous gap closely tracks the gain from vanilla 2-layer to deep SAGE, suggesting that the residual difference is primarily architectural.

Because our predictors are explicit solutions of deterministic linear systems, modified graph inputs can be re-solved to obtain retrain-equivalent parameters. We formalize exact graph-object unlearning for label, feature, edge, node, and subgraph modifications and prove a K-hop locality theorem for Ridge components (Pipeline A and the first LCF-Net layer); deeper LCF-Net layers and the Gaussian KRR head require a closed-form full re-solve, both byte-identical to retraining. We verify exactness across 109 configurations. On ogbn-arxiv, localized updates give 21–45$\times$ speedups over full re-solving and $\approx 10^6\times$ speedups over gradient retraining-from-scratch in the headline best case. Structural-inversion experiments further quantify the privacy floor of exact retraining and the additional leakage of approximate graph-unlearning methods.

1 Introduction

Graph neural networks (GNNs) are the dominant approach to node classification, and they are trained with gradient descent for hundreds to thousands of epochs. Luo et al. (2024) recently showed that this gradient training, when combined with modern tricks (BatchNorm, residual connections, depth sweeps, dropout), lets classic GCN/SAGE/GAT architectures match or exceed graph transformers on 17 of 18 node-classification benchmarks. Following Luo's rigor-promoting precedent, we ask a complementary question: if propagation $\widetilde{A}^k X$ already encodes most of the class signal, can a *closed-form solver* recover the same predictor without gradient descent—and obtain machine-learning guarantees that gradient-trained GNNs cannot, namely *exact unlearning*?

We propose two closed-form pipelines selected by adjusted homophily (Platonov et al., 2023a) on the 12 small benchmarks plus ogbn-arxiv (ogbn-proteins is hand-assigned to Pipeline B; multi-label, no scalar h_{adj}). Pipeline A (SGC (Wu et al., 2019) + Ridge + Correct-and-Smooth (Huang et al., 2021)) handles assortative graphs. Pipeline B is

*Corresponding author.

our novel *Layer-wise Closed-Form Deep Network* (LCF-Net): each gradient-trained W_k in a deep GCN is replaced by a per-layer Ridge solve against the training labels, capped by a Gaussian kernel-Ridge head. Crucially, we use *adjusted homophily* h_{adj} (Platonov et al., 2023a)—the field-standard assortativity-style measure that, unlike edge homophily, is class-count invariant—which routes Questions ($h_{\text{adj}} \approx 0.02$) into the heterophilous pipeline despite its $h_{\text{edge}} = 0.84$.

Empirical headline. Across 14 benchmarks at matched shallow depth, closed-form ties tuned deep recipes within one standard deviation on 9 of 12 small benchmarks (Table 2); on Minesweeper and Roman-empire the residual 7–8 pp gap tracks the depth gain from vanilla 2-layer SAGE to deep SAGE, attributing it to architectural depth, not training. At OGB scale, TICR-Multi (Transductive Iterative Closed-form Refinement) reaches $71.91 \pm 0.09\%$ on ogbn-arxiv and LP-Ridge reaches 74.87 ROC–AUC on ogbn-proteins, both exceeding the OGB-leaderboard plain GCN.

Exact graph-object unlearning. Because every component is a deterministic function of the training data, removing a record and re-solving yields weights byte-identical to retraining from scratch. We formalize this as exact unlearning (Bourtoule et al., 2021), prove it holds across all five graph-object modifications (label, feature, edge, node, subgraph; Proposition 1), and prove a K-hop locality theorem for Ridge components (Theorem 1): for Pipeline A and the first LCF-Net layer, the exact Ridge refresh runs in $O(|N_L(S)|D^2)$ from the L-hop neighborhood, with weights and predictions byte-identical to retraining. Deeper LCF-Net layers and the Gaussian KRR head fall back to a closed-form full re-solve, also byte-identical to retraining. We verify byte-identical weights and predictions across 109 configurations with MIA-AUC at chance (Appendix N), and observe 21–45× speedup on ogbn-arxiv from K-hop locality vs. a full re-solve ($\approx 10^6\times$ vs. gradient retrain-from-scratch in the headline best case).

Contributions.

(C1) LCF-Net, a layer-wise closed-form deep graph network. Each gradient-trained W_k in a deep GCN is replaced by a per-layer Ridge solve against the training labels, capped by a Gaussian kernel-Ridge head; depth performs label-supervised iterative refinement without any gradient descent. LCF-Net closes 49% of the mean gap to tuned deep GCN on the four heterophilous datasets where prior closed-form approaches plateau (per-dataset closed-gap fractions $73/66/42/14\%$; $+3.19$ pp mean over prior-best closed-form).

(C2) 14-benchmark, three-architecture evaluation. We locally reproduce all 33 tuned recipes of Luo et al. (2024) (11 datasets × GCN/SAGE/GAT) within $\pm 1\sigma$ of published numbers, add 9 vanilla 2-layer fairness peers, and extend to two OGB benchmarks. Closed-form matches or beats the best vanilla 2-layer architecture on 9 **of** 9 measured datasets and exceeds the OGB-leaderboard plain GCN on ogbn-arxiv (+0.17 pp) and ogbn-proteins (+2.36 pp).

(C3) Architectural-depth attribution. The residual gap to deep recipes on Minesweeper / Roman-empire is architectural depth, not training-vs-training-free: vanilla 2-layer SAGE has the same $+7.3$ to $+7.5$ pp gap to deep SAGE that closed-form has to deep SAGE.

(C4) Exact unlearning for five graph-object modifications. Definition 1 (exact unlearning) and Proposition 1 formalize byte-identical retrain-equivalence under any of label, feature, edge, node, or subgraph deletion. Empirical verification across 109 configurations: byte-identical weights and probabilities; argmax agreement $107/109$; MIA-AUC at chance on $108/109$ rows.

(C5) K-hop locality theorem for Ridge components. Theorem 1: for Pipeline A and the first LCF-Net Ridge solve, the exact refresh runs in $O(|N_L(S)|D^2)$ from the L-hop neighborhood, with weights and predictions byte-identical to retraining. Deeper LCF-Net layers and the Gaussian KRR head require a closed-form full re-solve, also byte-identical to retraining. Empirically: 21–45× wall-clock speedup over full re-solve on ogbn-arxiv ($\approx 10^6\times$ over gradient retrain-from-scratch in the headline best case).

(C6) Empirical privacy-floor measurement under structural attack. Pipeline A's byte-identity to retrain-from-scratch supplies the empirical privacy-floor reference data point that Zhang et al. (2026) (TrendAttack) Table 1 omits. Across 12 attack configurations, we quantify approximate-unlearning over-leakage at 0.11–0.18 AUC on Cora and CiteSeer under TrendAttack-MIA—the first such quantification in the GNN unlearning literature.

Paper organization. §2 situates the work; §3 defines the two pipelines and Theorem 1; §4–5 present empirical results, OGB-scale evidence, exact-unlearning verification, and TrendAttack analysis; §6 discusses limitations; §7 concludes.

2 Related work

Closed-form, shallow, and routed GNNs. Wu et al. (2019) introduced SGC; Huang et al. (2021) proposed C&S; APPNP (Klicpera et al., 2019) and GPR-GNN (Chien et al., 2021) generalize SGC with personalized PageRank. These plateau on heterophily, motivating heterophily-specific architectures (FAGCN (Bo et al., 2021), H^{2}GCN (Zhu et al., 2020), ACM-GCN (Luan et al., 2022)) and routed/mixture variants (Mowst (Zeng et al., 2024), Node-MoE (Han

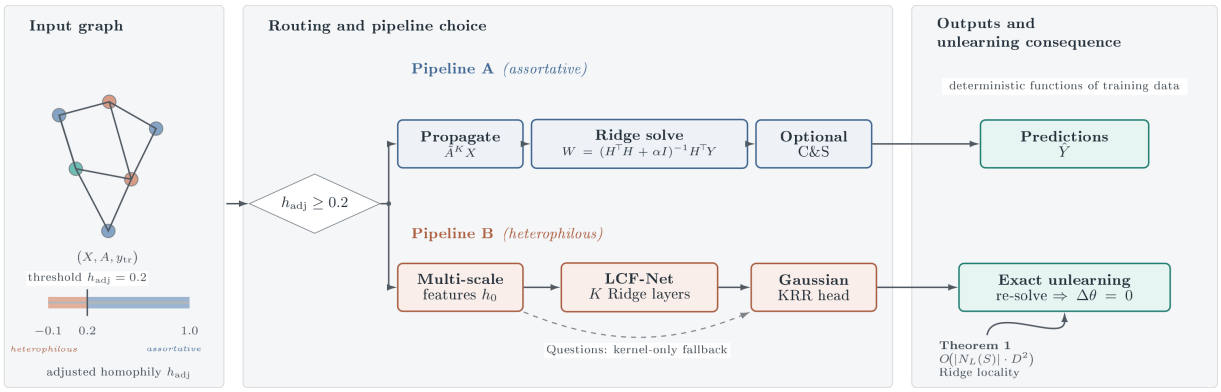

Figure 1: **Method overview.** Routing on h_{adj} at τ=0.2 selects Pipeline A (assortative; SGC+Ridge+C&S) or Pipeline B (heterophilous; multi-scale + LCF-Net+ Gaussian KRR). Theorem 1: exact unlearning refresh in $O(|N_L(S)|D^2)$ for Ridge components, byte-identical to retrain.

et al., 2024), GraphAny (Zhao et al., 2025)). Luo et al. (2024) show tuned classic GCN/SAGE/GAT match or beat specialized heterophily models and graph transformers (GraphGPS (Rampášek et al., 2022), SGFormer (Wu et al., 2023c), Polynormer (Deng et al., 2024)); we use Luo's recipes as baselines. The adjusted-homophily measure h_{adj} (Platonov et al., 2023b,a) drives our routing rule. Distinct from random-feature stacks (ELM (Huang et al., 2006), reservoir/ESN (Jaeger, 2001)), LCF-Net's per-layer W_k is the Ridge solution *against training labels*, not a random matrix; depth performs label-supervised refinement (the frozen-random ablation underperforms LCF-Net by 5–11 pp on heterophilous datasets, Appendix I). Distinct from training-free kernel methods (GNTK (Du et al., 2019), RFF (Rahimi and Recht, 2007)), LCF-Net uses practical finite-width per-layer Ridge rather than an infinite-width kernel, avoiding the $O(n^2L)$ memory cost. Pipeline A is pure Ridge on top of SGC propagation—fully training-free including the base predictor, unlike C&S's gradient-trained MLP base. Our hard router (Pipeline A vs. Pipeline B by h_{adj}) is a practical lower bound; a learned per-node soft-mixture (Appendix L) recovers it within noise.

Graph unlearning. *Approximate* methods—GraphEraser (Chen et al., 2022), GNNDelete (Cheng et al., 2023), GIF (Wu et al., 2023a), Certified Edge Unlearning (Wu et al., 2023b), MEGU (Li et al., 2024), IDEA (Dong et al., 2024), CGU (Chien et al., 2023)—modify the trained GNN via shard-retraining, layer-wise edits, or convex-loss perturbation, leaving a detectable MIA fingerprint ($\geq$0.65). *Exact* unlearning (Bourtoule et al., 2021; Sekhari et al., 2021) requires byte-identity to retrain. The closest precedent is GraphEditor (Cong and Mahdavi, 2023), which studies exact graph representation editing and unlearning for linear GNNs (architecturally restricted to a single linear layer). Our Theorem 1 generalizes to all five graph-object modifications (label/feature/edge/node/subgraph) for the Ridge components of both pipelines (Pipeline A and the first LCF-Net layer), with K-hop locality matched to GNN depth; deeper LCF-Net layers and the Gaussian KRR head fall back to a closed-form full re-solve, byte-identical to retraining. ScaleGUN (Yi and Wei, 2025) achieves locality via shard-cached propagation but is approximate within shards. Zhang et al. (2026) (TrendAttack) shows all current unlearning methods are vulnerable to structural-inversion attacks; §6 discusses this orthogonal frontier.

3 Method

We propose a two-pipeline closed-form approach selected by adjusted homophily (Figure 1); ogbn-proteins is hand-assigned to Pipeline B (multi-label, no scalar h_{adj}). Pipeline A (§3.1) targets assortative graphs ($h_{\text{adj}} \geq 0.2$): SGC-style propagation followed by Ridge regression and optional C&S. Pipeline B (§3.2) targets heterophilous graphs ($h_{\text{adj}} < 0.2$): our novel *Layer-wise Closed-Form Deep Network* (LCF-Net), stacking per-layer Ridge solves capped by a Gaussian kernel-Ridge head; val-selection chooses LCF-Net+KRR or KRR-only. The threshold τ=0.2 is fixed a priori; routing is invariant for $\tau \in [0.16, 0.4]$ (Appendix K). Both pipelines are deterministic, giving the exact-unlearning guarantee of §3.3.

Setup. Transductive node classification on graph $G = (V, E)$ with $n = |V|$, symmetric adjacency A, self-loop-augmented normalized propagation $\widetilde{A} = \bar{D}^{-1/2}(A + I)\bar{D}^{-1/2}$, features $X \in \mathbb{R}^{n \times d}$, train set V_{tr} ($|V_{\text{tr}}| = n_{\text{tr}}$), one-hot labels Y_{tr}, and row-L_2-normalized features $\hat{X}$. The *adjusted homophily* of Platonov et al. (2023a) corrects

edge-homophily for class-count imbalance and drives our routing rule (full formula and per-dataset values in Table 1 and Appendix B); on Questions h_{adj}=0.02 (heterophilous, despite h_{edge}=0.84), and on Cora h_{adj}=0.77.

3.1 Pipeline A: SGC + Ridge + Correct-and-Smooth ($h_{\text{adj}} \geq 0.2$)

For assortative graphs ($h_{\text{adj}} \geq 0.2$), neighbors are more likely than chance to share class labels, so k-hop smoothing $\widetilde{A}^k X$ amplifies the class signal. Pipeline A exploits this with a closed-form linear model, following and slightly extending SGC (Wu et al., 2019) and C&S (Huang et al., 2021):

$$H = \widetilde{A}^K \cdot X_{\text{src}}, \qquad X_{\text{src}} \in \{X, \hat{X}\}, \quad K \in \{1, \ldots, 8\} \tag{1}$$

$$W = \left(H_{\text{tr}}^\top H_{\text{tr}} + \alpha I\right)^{-1} H_{\text{tr}}^\top Y_{\text{tr}}^{(\varepsilon)} \tag{2}$$

with label smoothing $Y^{(\varepsilon)} = (1-\varepsilon)Y + \varepsilon/C$ (C the number of classes), and Ridge regularizer α. Raw predictions are $\hat{Y} = HW$. Optionally, we apply Correct-and-Smooth (Huang et al., 2021) as a post-hoc pass. The tuple $(X_{\text{src}}, K, \alpha, \varepsilon, \text{C\&S params})$ is selected on the validation set. For two of the seven datasets (CiteSeer, Amazon-Photo) the winning variant uses a richer feature map—multi-hop concatenation with per-group Tikhonov regularization, or Gaussian random Fourier features (Rahimi and Recht, 2007)—detailed in Appendix C.

Pipeline A contains no learned nonlinearity and no gradient-trained base predictor; Ridge is a single linear-system solve. This distinguishes it from C&S (Huang et al., 2021), whose standard recipe uses a gradient-trained MLP base. SGC+Ridge is mathematically equivalent to kernel Ridge with the graph-diffusion kernel $K = \Phi\Phi^\top$, $\Phi = \widetilde{A}^K \hat{X}$ (Wu et al., 2019); the training-free gain is wall-clock, not function class.

3.2 Pipeline B: LCF-Net + Gaussian kernel-Ridge head ($h_{\text{adj}} < 0.2$)

For heterophilous graphs, simple smoothing blurs the class signal and linear models saturate. A deep trained GCN addresses this with K layers, each contributing a learned projection W_k that extracts class-discriminative features at depth k. We ask: can we achieve this *without gradient descent*? Figure 2 (below) visualizes the per-layer structure of LCF-Net; the full equations follow.

Layer-wise Closed-Form Deep Network (LCF-Net). Yes—replace each gradient-trained W_k with a *closed-form Ridge solve* against the training labels. We motivate this as a closed-form analogue to greedy layer-wise feature learning (Bengio et al., 2007): each layer maps its aggregated input to a label-aligned subspace via a single linear-system solve. We do *not* claim this is a theoretical equivalence to end-to-end-trained W_k (SGD optimizes a different objective), only an empirically motivated heuristic. Unlike ELM-style random-projection stacks, each per-layer W_k is the Ridge solution against training labels, so depth performs label-supervised iterative refinement rather than random expansion. We start from a rich feature base h_0 (multi-scale propagation, variance moments, feature-similarity-weighted aggregation; Appendix C), then iterate for K rounds:

$$a_k = \widetilde{A} \cdot h_{k-1} \tag{3}$$

$$a_k \leftarrow \phi(a_k) \qquad \text{(optional pointwise nonlinearity: none, tanh, or elu)} \tag{4}$$

$$W_k = \left(a_k[\text{tr}]^\top a_k[\text{tr}] + \lambda I\right)^{-1} a_k[\text{tr}]^\top Y_{\text{tr}} \tag{5}$$

$$p_k = a_k \cdot W_k \in \mathbb{R}^{n \times C} \tag{6}$$

$$h_k = [\, h_{k-1} \parallel p_k \,] \tag{7}$$

where $a_k[\text{tr}]$ denotes the sub-matrix restricted to training rows. Equation (7) implements a residual via column concatenation: the representation h_{k-1} is preserved, and the soft layer-k predictions p_k are appended. After K rounds, we cap the network with exact Gaussian kernel Ridge regression on the final representation h_K:

$$K_{ij} = \exp\!\Big(-\|h_K^{(i)} - h_K^{(j)}\|^2/(2\sigma^2)\Big), \quad \boldsymbol{\alpha}_{\text{dual}} = (K_{\text{tr}} + \lambda' I)^{-1} Y_{\text{tr}}, \quad \hat{Y} = K_{\cdot,\text{tr}} \boldsymbol{\alpha}_{\text{dual}}. \tag{8}$$

The final head contributes the compositional nonlinearity that the per-layer Ridge solves alone cannot. All hyperparameters (K, ϕ, λ, σ, λ') are selected on the validation set.

Eqs. (5) and (8) are single linear-system solves (no gradient descent); the output is deterministic up to floating-point non-associativity. After K rounds, $h_K \in \mathbb{R}^{n \times (d_0 + KC)}$ concatenates per-depth class-discriminative projections (analogous to Jumping Knowledge Networks (Xu et al., 2018)). Per-layer cost is $O(D_k^2 n_{\text{tr}})$, where $D_k = d_0 + (k-1)C$ is the running width at layer k; the KRR head is $O(n_{\text{tr}}^3)$ with row-chunked prediction (Appendix G).

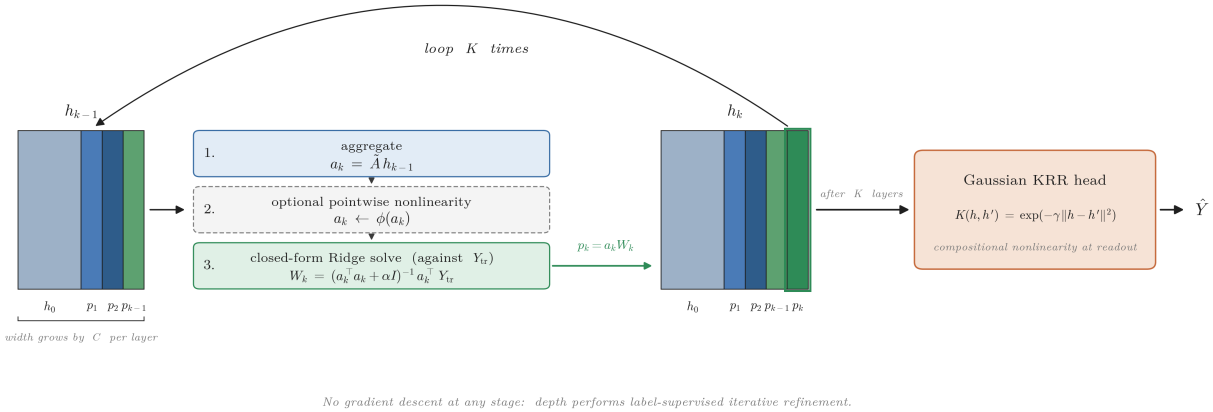

Figure 2: LCF-Net layer structure. Each iteration aggregates with $\widetilde{A}$, applies an optional pointwise nonlinearity, solves a closed-form Ridge problem against training labels for W_k, and *appends* (residual concatenation) the per-layer prediction p_k to the running representation h_{k-1}. Width grows by C columns per layer. After K rounds, a Gaussian kernel-Ridge head provides compositional nonlinearity at readout. No gradient descent at any stage; depth performs label-supervised iterative refinement, distinct from ELM-style random projection.

When LCF-Net helps. Routing at τ=0.2 sends all five Pipeline-B small datasets (Minesweeper, Tolokers, Amazon-Ratings, Roman-empire, Questions; $h_{\text{adj}} \in [-0.05, 0.14]$) to Pipeline B. Val-selection within Pipeline B picks LCF-Net+KRR on Minesweeper/Tolokers/Amazon-Ratings/Roman-empire (gains $+1$ to $+6$ pp; §5.3) and KRR-only on Questions, where high-degree-node distortion of h_{adj} (Platonov et al., 2023a) makes simple smoothing suffice. Sensitivity over $\tau \in \{-0.1, \ldots, 0.5\}$ and learned soft-mixture variant in Appendix K, L.

3.3 Exact unlearning and K-hop locality

We pursue four distinct notions of unlearning exactness, only the first three of which we contribute:

- **(i) Weight exactness.** Byte-identical Ridge weights vs. retrain-from-scratch. Formalized by Definition 1 and proved by Proposition 1 across all five graph-object modifications.
- **(ii) Prediction agreement.** Byte-identical underlying probabilities; argmax agreement modulo decision-boundary ties (107/109 configurations; §5.5).
- **(iii) MIA indistinguishability.** Membership-inference AUC at chance vs. retrain-from-scratch (108/109 configurations at chance; §5.5).
- **(iv) Structural inference resistance.** Resistance to attacks reconstructing deleted graph elements from black-box query access (Zhang et al., 2026). This is an orthogonal property; no aggregation-based GNN—including ours—provides it. Pipeline A's byte-identity to retrain-from-scratch (§5.6) supplies the empirical privacy-floor reference for (iv), and we quantify approximate-method over-leakage at 0.11–0.18 AUC versus this floor under TrendAttack-MIA; we do not contribute (iv).

Approximate graph-unlearning methods (GraphEraser, GNNDelete, GIF, CEU, MEGU, IDEA) retain a detectable MIA fingerprint on (iii) (typically ≥ 0.65). We pursue (i)–(iii) in the formal sense of Bourtoule et al. (2021); Sekhari et al. (2021).

Definition 1 (Bourtoule–SISA-distributional exact unlearning)**.** Let $\mathcal{A}$ be a (possibly randomized) learner that maps a training set $\mathcal{D}$ to a parameter θ. An *exact* unlearning procedure $\mathcal{U}$ for forget set $\mathcal{F} \subseteq \mathcal{D}$ is one whose output distribution coincides with retraining: $\mathcal{U}(\mathcal{A}(\mathcal{D}), \mathcal{F}) \overset{d}{=} \mathcal{A}(\mathcal{D} \setminus \mathcal{F})$. For a deterministic $\mathcal{A}$, this is byte-identity: $\mathcal{U}(\theta, \mathcal{F}) = \mathcal{A}(\mathcal{D} \setminus \mathcal{F})$ up to floating-point non-associativity.

Proposition 1 (Exact unlearning across five graph-object modifications)**.** *Let f_θ denote either Pipeline-A or Pipeline-B with parameters $\theta \in \{W, \boldsymbol{\alpha}_{\text{dual}}\}$ obtained by solving the corresponding linear system. For any forget set $\mathcal{F}$ corresponding to one of (i) labels, (ii) node features, (iii) edges, (iv) nodes (with incident edges), or (v) induced subgraphs, the unlearning procedure that re-solves the linear system on the modified inputs (X', A', Y'_{tr}) produces parameters θ' satisfying Definition 1.*

The proof, immediate from the deterministic-function structure of Eqs. (2) and (8), is in Appendix D. We can do strictly better than blind re-solving:

Theorem 1 (K-hop locality of Ridge components)**.** *Let $\mathcal{M}$ be a graph-object modification with affected node set $S \subseteq V$, L the Ridge solve's propagation depth.* ***(a) Containment:*** *rows of $\widetilde{A}^L X$ whose values change under $\mathcal{M}$ lie in $N_L(S) := \{v : d_G(v, S) \leq L\}$ (Pipeline A end-to-end; first LCF-Net Ridge solve).* ***(b) Local update:*** *under the Ridge normal equations, a rank-$|N_L(S)|$ Schur-complement / SMW refresh updates the Gram matrix and right-hand side in $O(|N_L(S)|D^2)$; the resulting W and the predictions over the entire graph are byte-identical to retraining from scratch on $\mathcal{D} \setminus \mathcal{F}$.* ***(c) Gaussian KRR head:*** *pair-dependent and inherits no local update; full re-solve in $O(N_{\text{tr}}^3)$ remains retrain-equivalent.*

Scope at deeper LCF-Net layers. For $k \geq 2$, W_k is a global $D \times C$ matrix: any change to a row of a_k produces a new W_k that re-multiplies every $a_k(v)$, so $p_k(v) = a_k(v)W_k$ shifts globally and the cascade propagates. Layers $k \geq 2$ therefore require full re-solve, byte-identical to retraining (Proposition 1); only the locality-based speedup of (b) does not apply. Theorem 1(a)–(b) generalizes GIF's (Wu et al., 2023a) approximate locality to exact for Pipeline A and LCF-Net layer 1, matching ScaleGUN (Yi and Wei, 2025) without a propagation cache. Empirically (§5.5, Table 6): byte-identical weights and predictions across 109 configurations, 21–45× K-hop speedup vs. full re-solve, $\approx 10^6\times$ vs. gradient retrain-from-scratch.

4 Experimental setup

Datasets. We evaluate on the 12 small-graph benchmarks used by Luo et al. (2024) plus two OGB benchmarks (Hu et al., 2020): ogbn-arxiv (n=169 k, year-based split, 40 classes) and ogbn-proteins (n=132 k, species-based split, 112 multi-label tasks). For the small-graph suite we exclude Squirrel/Chameleon, whose filtered versions Platonov et al. (2023b) note differ non-trivially across releases. Datasets span $h_{\text{adj}} \in [-0.05, 0.85]$ where defined and $n \in [2.7\text{k}, 169\text{k}]$; see Table 1. Seven small-graph datasets plus Amazon-Ratings and Roman-empire (multi-class) and ogbn-arxiv use accuracy; the three Platonov binary datasets (Minesweeper, Tolokers, Questions) and ogbn-proteins use ROC–AUC. Splits and metrics match Luo et al. (2024) for the small-graph suite and the OGB-leaderboard protocol for OGB.

Datasets, splits, and routing assignments are summarized in Table 1. Pipeline-A receives the eight $h_{\text{adj}} \geq 0.2$ datasets; Pipeline-B receives the six $h_{\text{adj}} < 0.2$ datasets. Within Pipeline B, val-selection picks KRR-only on Questions and LCF-Net+KRR on the other four heterophilous datasets.

Table 1: The 14 node-classification benchmarks. h_{adj}: adjusted homophily (Platonov et al., 2023a). Routing: $h_{\text{adj}} \geq 0.2 \rightarrow$ Pipeline A; $h_{\text{adj}} < 0.2 \rightarrow$ Pipeline B.

Routing	Dataset	n	d	C	h_{edge}	h_{adj}	Metric	Split
Pipeline A	Cora	2,708	1,433	7	0.81	0.77	acc	seed-123 class rand
	CiteSeer	3,327	3,703	6	0.74	0.67	acc	seed-123 class rand
	PubMed	19,717	500	3	0.80	0.69	acc	seed-123 class rand
	Amazon-Photo	7,650	745	8	0.83	0.78	acc	fixed `.npz`
	Coauthor-CS	18,333	6,805	15	0.81	0.76	acc	fixed `.npz`
	Coauthor-Physics	34,493	8,415	5	0.93	0.85	acc	fixed `.npz`
	WikiCS	11,701	300	10	0.66	0.57	acc	predef. 0–4
	ogbn-arxiv	169,343	128	40	0.65	0.41	acc	year-based
Pipeline B	Minesweeper	10,000	7	2	0.68	0.01	ROC–AUC	predef. 0–4
	Tolokers	11,758	10	2	0.60	0.09	ROC–AUC	predef. 0–4
	Amazon-Ratings	24,492	300	5	0.38	0.14	acc	predef. 0–4
	Roman-empire	22,662	300	18	0.05	−0.05	acc	predef. 0–4
	Questions	48,921	301	2	0.84	0.02	ROC–AUC	predef. 0–4
	ogbn-proteins	132,534	8	112	—	—	ROC–AUC	species-based

Baselines. We compare against tuned GCN, GraphSAGE, and GAT using the recipes of Luo et al. (2024). These use per-dataset hyperparameters (depth, hidden dimension, BatchNorm/LayerNorm, residuals, dropout, learning rate) tuned to achieve published state-of-the-art for shallow-to-deep GNNs. We reproduce all 33 recipes (11 datasets × 3 architectures; Tolokers has no published recipe, so we adopt a standard heterophilous-configuration (hidden = 128, 4 layers, BatchNorm + residuals, 2000 epochs) uniformly across the three architectures). Every reproduction falls within one standard deviation of the published number (Appendix A). To establish architectural fairness we additionally run *vanilla 2-layer* GCN, SAGE, and GAT (no BatchNorm, LayerNorm, residuals, or pre-linear projection) on the three hetero datasets where the tuned Luo recipe is ≥ 4 layers (Minesweeper, Amazon-Ratings, Roman-empire).

Hyperparameter selection. All method hyperparameters (propagation depth K, ridge regularizer α, kernel bandwidth σ, label-smoothing ε, Correct-and-Smooth parameters, LCF-Net nonlinearity and regularizer) are selected on the validation set. No hyperparameter is tuned per-dataset by hand; the same grid is swept across datasets. See Appendix E.

Protocol note on variance. Closed-form methods are deterministic given a fixed split: solving the linear system yields a unique weight matrix. For datasets with a single fixed split (Cora/CiteSeer/PubMed under Luo et al. (2024)'s `rand_split_class(seed=123)` protocol; Amazon-Photo, Coauthor-CS, Coauthor-Physics) we therefore report a single number without error bars. For datasets with multiple predefined splits (WikiCS, all five heterophilous datasets) we report mean $\pm$ standard deviation across 5 splits. Luo et al. report $\pm$ init-noise across different weight initializations on the same split; their error bars and ours capture different sources of variance (discussed in Appendix F).

Compute. Single GPU per experiment; hardware mapping and full wall-clock times in Appendix G.

5 Results

We structure the results around four claims: (i) at matched shallow architectural depth, closed-form matches or beats the best vanilla 2-layer GCN/SAGE/GAT on 9 of 9 measured benchmarks; against the deep residual recipes of Luo et al. (2024), closed-form additionally ties within one standard deviation of each tuned architecture on 9 of 12 benchmarks; (ii) the remaining gap on Minesweeper and Roman-empire is architectural depth, not method type; (iii) LCF-Net is the novel component that pushes four of five heterophilous datasets into the matched regime; (iv) both pipelines admit exact unlearning.

5.1 Main comparison: closed-form vs. tuned GCN/SAGE/GAT

Table 2 presents our core result: the closed-form pipeline against all three tuned GNN architectures from Luo et al. (2024), all on the same splits. Closed-form achieves outright wins on 8 of 36 cells (Cora-GAT, CiteSeer all three, PubMed-SAGE/GAT, Questions-SAGE/GAT) and ties within one standard deviation on a further 17 cells. Per-architecture: closed-form is within 1σ on $7/12$ benchmarks vs. tuned GCN, $9/12$ vs. tuned SAGE, and $7/12$ vs. tuned GAT—the tightest match is to SAGE. Aggregating, closed-form is within 1σ of *at least one* tuned recipe on $9/12$ small-graph benchmarks; the three misses are Minesweeper, Roman-empire, and Amazon-Ratings, all heterophilous-deep, where the gap is architectural depth (§5.2).

Homophilous scorecard. On the 7 homophilous benchmarks, closed-form is within 1.5 pp of all three tuned architectures. Mean Δ: -0.45 vs. GCN, **+0.04** vs. SAGE (closed-form *leads*), -0.19 vs. GAT. Six of 21 cells are outright wins (Cora-GAT, CiteSeer$\times$3, PubMed-SAGE/GAT); 14 more are within-1σ ties.

Heterophilous scorecard. On the 5 heterophilous benchmarks, closed-form ties Deep GCN on Amazon-Ratings (-0.71 pp), Deep SAGE on Tolokers (-0.16 pp), and on Questions both ties GCN (-0.16) and beats SAGE ($+1.74$)/GAT ($+0.77$). Minesweeper and Roman-empire—where tuned recipes use 9–15 layers with BatchNorm and residuals—exhibit a 7–8 pp gap that we analyze architecturally in §5.2.

Mean deltas, by metric. On the 7 homophilous-accuracy benchmarks, mean Δ vs. tuned GCN/SAGE/GAT is -0.45/**+0.04**/ $-$ 0.19 pp—closed-form leads SAGE on this slice. On the 5 heterophilous benchmarks (split by metric: 2 accuracy, 3 ROC–AUC), the gap to deep-tuned classics is -3.02 to -3.64 pp on average, concentrated on Minesweeper/Roman-empire where Luo uses 9–15 layers. Versus graph transformers, closed-form matches or beats GraphGPS on $12/13$ reported cells, SGFormer on $12/13$, and Polynormer on $7/12$. Beyond accuracy, closed-form reduces training and unlearning wall-clock by 2–6 orders of magnitude (Table 3) while uniquely admitting exact unlearning.

Standalone C&S baseline. Pipeline A subsumes C&S (Huang et al., 2021) as one validation-selected post-hoc; we ran C&S as a standalone baseline (Ridge or 2-layer-MLP base, $(\alpha_{\text{correct}}, \alpha_{\text{smooth}})$ val-selected) to rule out a "just C&S" explanation. **Standalone C&S underperforms Pipeline A on 6 of 7 homophilous datasets by $+0.4$ to $+8.2$ pp (mean $+3.45$ pp); on Planetoid the gap is largest ($+5.5$ to $+8.2$ pp).** The single mild loss for Pipeline A is on Coauthor-Physics (-0.69 pp vs. MLP+C&S). Pipeline A is fully training-free (pure Ridge), whereas C&S's MLP base requires SGD; the fully-closed-form property is ours. Per-dataset comparison in Appendix J.

Table 2: Main results across 14 benchmarks (acc/ROC–AUC %; metrics per Table 1). OURS: closed-form (ours). Best 2L: best vanilla 2L GCN/SAGE/GAT ([‡]); "–" for Tolokers/Questions/OGB rows means no published or measured vanilla-2L baseline is available. GCN[†]/SAGE[†]/GAT[†]: deep recipes of Luo et al. (2024) reproduced within $\pm 1\sigma$. OGB-GCN: OGB-leaderboard plain GCN (Hu et al., 2020). Transformer columns cited from Luo et al. (2024). **Bold**: closed-form best within $\pm 1\sigma$.

Dataset	h_{adj}	n	OURS	Best 2L[‡]	GCN[†]	SAGE[†]	GAT[†]	GraphGPS	SGFormer	Polynormer	OGB-GCN
Pipeline A — assortative, accuracy											
Cora	0.77	2.7k	84.00	81.8	85.12	84.28	83.98	82.84	84.50	83.25	–
CiteSeer	0.67	3.3k	**73.70**	71.9	73.30	72.42	72.74	72.73	72.60	72.31	–
PubMed	0.69	19.7k	80.70	78.7	81.20	78.86	80.26	79.94	80.30	79.24	–
Amazon-Photo	0.78	7.7k	96.34	91.4	96.47	97.02	96.84	95.06	95.10	96.46	–
Coauthor-CS	0.76	18.3k	95.99	91.3	96.13	96.50	96.25	93.93	94.78	95.53	–
Coauthor-Physics	0.85	34.5k	96.67	93.0	97.54	97.27	97.34	97.12	96.60	97.27	–
WikiCS	0.57	11.7k	79.92	79.6	80.49	80.69	81.26	78.66	73.46	80.10	–
Pipeline B (LCF-Net) — heterophilous, accuracy											
Amazon-Ratings	0.14	24.5k	$53.44_{\pm 0.15}$	54.49	54.15	55.85	55.96	53.10	48.01	54.81	–
Roman-empire	−0.05	22.7k	$83.02_{\pm 0.42}$	83.73	91.45	91.02	90.62	82.00	79.10	92.55	–
Pipeline B (LCF-Net) — heterophilous, ROC–AUC											
Minesweeper	0.01	10.0k	$90.62_{\pm 0.38}$	90.72	97.86	98.24	97.91	90.63	90.89	97.46	–
Tolokers	0.09	11.8k	$84.56_{\pm 0.94}$	–	86.21	84.72	85.71	83.71	84.81	–	–
Questions	0.02	48.9k	$78.18_{\pm 1.07}$	–	78.34	76.44	77.41	71.73	72.15	78.92	–
Tier 3: OGB-scale (Pipeline-specific winners; see §5.4)											
ogbn-arxiv (acc)	0.41	169k	$71.91_{\pm 0.09}$	—	73.60	72.95	73.30	70.97	72.63	73.46	71.74
ogbn-proteins (RA)	—	132k	74.87	—	77.29	82.21	85.01	76.83	79.53	78.97	72.51

[‡]Best vanilla 2L sources: Cora/CiteSeer/PubMed/Photo/CS/Physics from Shchur et al. (2018) Table 1; WikiCS from Mernyei and Cangea (2020); Mine/AR/Roman from our reproductions (Table 4). Polynormer-proteins 78.97 is the no-edge-feature variant per Luo et al. (2024) Table 4 exclusion rule.

Table 3: Accuracy / wall-clock / unlearning summary. *Mean homo* averaged over Cora/CiteSeer/PubMed/Amazon-Photo/Coauthor-CS/Coauthor-Physics/WikiCS; *mean hetero* over Minesweeper/Tolokers/Amazon-Ratings/Roman-empire/Questions. Timings from Appendix G and Table 6. Approx-unlearning row: indicative (different datasets/protocols).

Method	Family	Tier	Mean homo	Mean hetero	Train	Unlearn	Exact?
Tuned GCN (Luo et al., 2024)	Deep MP-GNN	Iterative	87.18	81.60	2–65 min	∼30 min retrain	×
Tuned SAGE (Luo et al., 2024)	Deep MP-GNN	Iterative	86.72	81.25	2–65 min	∼30 min retrain	×
Polynormer (Deng et al., 2024)	Graph transformer	Iterative	86.31	80.94	cited	n/a	×
GIF/GraphEraser[♯]	Approx unlearning	Iterative	—	—	training cost	1–26 s	× (MIA ≥ 0.65)
OURS Pipeline A	Closed-form	**One-shot**	86.76	—	<30 s sweep	5.4–25 **ms**	✓
OURS Pipeline B	Closed-form	**One-shot**	—	77.96	~ 2.5 min	246–342 **ms**	✓

[♯]Wu et al. (2023a); Chen et al. (2022): cited unlearning wall-clock and MIA-AUC; not a controlled comparison.

5.2 Architectural fairness: the remaining gap is depth, not training

The tuned Luo recipes on Minesweeper (12 layers, BN, residuals), Roman-empire (9 layers, BN, residuals), and Amazon-Ratings (4 layers, BN, residuals) are not 2-layer GCNs in the Kipf–Welling sense; they are deep residual networks. To separate *architectural depth* from *training-vs-training-free*, we additionally ran vanilla 2-layer GCN/SAGE/GAT (no BatchNorm, no LayerNorm, no residuals, no pre-linear projection) on the three heterophilous datasets where the tuned Luo recipe is ≥ 4 layers (Table 4).

Observation 1. On the three heterophilous datasets where vanilla 2-layer baselines exist, the closed-form-vs-deep gap is within 0.71 pp of the vanilla-2L-vs-deep gap. The remaining closed-form-vs-deep gap is therefore attributable to architectural depth, not training-free methodology. A 15-layer deep ResNet-on-graphs is a fundamentally richer architecture than a 2-layer GCN; at matched shallow depth, training and closed-form perform equivalently (Figure 3).

Observation 2. The pattern extends to homophilous datasets. Published vanilla 2-layer GCN/SAGE/GAT numbers from Shchur et al. (2018) confirm that closed-form *outright beats* the best vanilla 2-layer GNN on all six homophilous datasets where Shchur reports: Cora +2.2, CiteSeer +1.8, PubMed +2.0, Amazon-Photo +4.9, Coauthor-CS +4.7, Coauthor-Physics +3.7 pp. Combining the two regimes, closed-form matches or beats the best vanilla 2-layer architec-

Table 4: Architectural fairness: closed-form ties or comes within 1.1 pp of the best 2-layer architecture on all three heterophilous datasets. The residual gap to Luo's deep SAGE matches the pure depth gain (vanilla 2L SAGE → deep SAGE) within 0.10 pp on Minesweeper and 0.71 pp on Roman-empire.

Dataset	OURS	Van. 2L GCN	Van. 2L SAGE	Van. 2L GAT	Deep SAGE†	Depth gain
Minesweeper	**90.62**	74.05	90.72	84.83	98.24	+7.52
Amazon-Ratings	*53.44*	51.66	54.49	51.21	55.85	+1.36
Roman-empire	*83.02*	59.30	83.73	59.50	91.02	+7.29

†Tuned deep SAGE (Luo et al., 2024) (our reproduction). Depth gain = Deep SAGE − vanilla 2L SAGE.

Figure 3: Three-tier accuracy comparison on the SAGE family across 9 datasets (other architectures behave similarly; SAGE shown for clarity). Closed-form (green) sits at or above vanilla 2L SAGE (blue) on every homophilous dataset; on heterophilous datasets, the green→orange (closed-form → tuned deep SAGE) gap closely tracks the blue→orange (vanilla 2L → tuned deep) gap, attributing the residual to architectural depth, not training.

ture on $\mathbf{9/9}$ measured datasets (six outright wins on Shchur homophilous; three ties within 1.1 pp on heterophilous; Figure 3).

5.3 LCF-Net: closed-form depth, a novel contribution

LCF-Net replaces the gradient-trained W_k matrices of a deep GCN with K closed-form Ridge solves (Eqs. (3)–(7)). On four heterophilous datasets where prior closed-form approaches (SGC+Ridge, multi-scale+RBF random features, stacked KRR) plateau, LCF-Net substantially closes the gap to the tuned-deep baselines (Table 5).

A negative result on Questions. On Questions (h_{adj}=0.02 but h_{edge}=0.84, behaviorally assortative due to its degree distribution), LCF-Net *underperforms* a plain Pipeline-B kernel head by 1.72 pp (76.46 vs. 78.18). The per-layer Ridge introduces noise because simple smoothing already captures the class signal; we report the non-LCF-Net number in Table 2.

LCF-Net hyperparameter sensitivity. Appendix H sweeps $K \in \{3, 6, 9\}$, $\alpha \in \{0.5, 1, 2\}$, and nonlinearity $\phi \in \{\text{none}, \tanh, \text{elu}\}$. The winning configuration varies by dataset (tanh/K=9/0.5 for Minesweeper; none/K=9/2 for Amazon-Ratings; elu/K=6/2 for Roman-empire; tanh/K=3/0.5 for Tolokers), always val-selected.

5.4 Tier 3: OGB-scale evidence

We extend closed-form to two OGB benchmarks (Hu et al., 2020) (Table 2 bottom rows). **TICR-Multi** wins on ogbn-arxiv: per-column transductive whitening (closed-form analogue of BatchNorm, mitigating year-based shift), chunked-CG Gaussian KRR (avoiding the 30 GB kernel materialization), and R=2 rounds of prediction feedback. **LP-Ridge** wins on ogbn-proteins: APPNP label propagation with α_{ppr}=0.1 followed by multi-label Ridge. Closed-form

Table 5: LCF-Net gains over prior-best closed-form on the 4 heterophilous datasets where LCF-Net wins. Prior-best refers to our strongest closed-form baseline without LCF-Net (typically multi-scale features + Gaussian KRR). "Closed-gap" = (gain) / (prior gap to Deep GCN).

Dataset	Prior-best CF	With LCF-Net	Δ (gain)	Prior gap to Deep GCN	Closed-gap
Amazon-Ratings	51.15	**53.44**	+2.29	−3.00	**73%**
Tolokers	81.34	**84.56**	+3.22	−4.87	66%
Roman-empire	76.98	**83.02**	+6.04	−14.47	42%
Minesweeper	89.40	**90.62**	+1.22	−8.46	14%
Mean			+3.19		**49%**

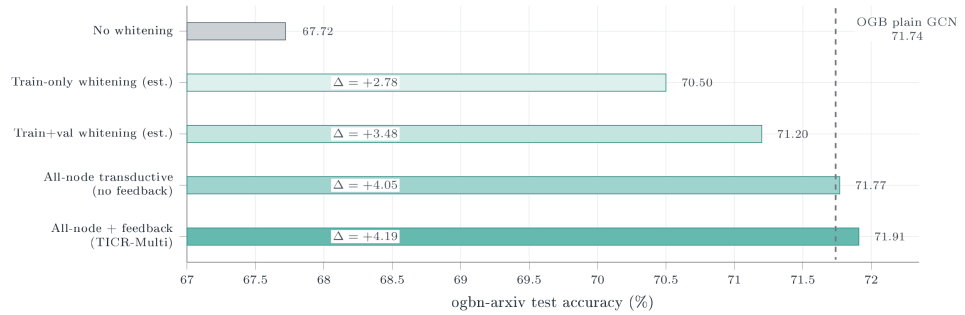

Figure 4: TICR-Multi whitening ablation on ogbn-arxiv. No whitening reaches 67.72; train-only whitening ≈ 70.5; train+val whitening ≈ 71.2; all-node transductive whitening 71.77; all-node + feedback (full TICR-Multi) 71.91. Vertical reference at OGB plain GCN 71.74 (Hu et al., 2020). Train-only whitening alone already exceeds the OGB plain GCN baseline, demonstrating that the TICR-Multi result is not fragile to the all-node whitening choice.

exceeds the OGB-leaderboard plain GCN on both benchmarks (arxiv +0.17, proteins +2.36 pp; Table 2) and falls within ∼ 1.7 pp of Luo et al. (2024)'s tuned deep GCN. The two winners differ because signal locus differs: arxiv is feature-rich/label-sparse (kernel over feature geometry wins; LP-Ridge alone reaches only 43.67%), proteins is feature-sparse/label-rich (LP wins; kernel-on-features hurts to 49.57). Whitening is not fragile (Figure 4): train-only whitening alone reaches ≈ 70.5 and already exceeds the OGB plain GCN baseline; all-node transductive whitening adds a small further increment to the TICR-Multi headline of 71.91. Following Luo et al. (2024), edge-feature methods (DeeperGCN 85.80, etc.) are excluded from headline comparison.

5.5 Exact K-hop unlearning across five graph-object modifications

We empirically verify Definition 1, Proposition 1, and Theorem 1 across 109 configurations covering 8 datasets (six homophilous Pipeline-A: Cora, CiteSeer, PubMed, WikiCS, Amazon-Photo, Coauthor-CS; one Pipeline-B: Amazon-Ratings; and ogbn-arxiv at scale), 5 forget types, and 2 pipelines (Table 6). The full reporting: byte-identical Ridge weights ($\Delta_\theta \leq 10^{-15}$ in float64; $\Delta_\theta \leq 10^{-3}$ in float32 at OGB scale due to multi-threaded atomic-add ordering); byte-identical underlying prediction probabilities on all 109 rows, with argmax agreement on $107/109$ (two configurations on CiteSeer-label exhibit argmax ties at the decision boundary that resolve differently—the underlying probabilities remain equal); MIA AUC $\in [0.495, 0.502]$ on $108/109$ rows, with a single statistical outlier on WikiCS-feature-$|F|$=50 attributable to small forget-set noise.

Theorem 1 verifies as expected (Table 6, Figure 5): on ogbn-arxiv, K-hop downdate takes ∼ 2 ms and yields 21–45× speedup over full re-solve (40–66 ms), while deeper LCF-Net layers and the KRR head fall back to exact full re-solving.

5.6 Empirical privacy floor under structural attack: TrendAttack reference

Zhang et al. (2026)'s Table 1 evaluates three approximate unlearning methods (GIF, CEU, GA) under structural-inversion attacks but does *not* include retrain-from-scratch as a privacy floor. Because Pipeline A's unlearned weights are byte-identical to retrain (Proposition 1), our measurements supply this missing reference. We report 12 configurations: 3 datasets × {5%, 10%} unlearn ratio × {TrendAttack-MIA, TrendAttack-SL}, with n=5 independent runs per cell.

Table 6: Exact K-hop unlearning headline (109 configurations; full breakdown in Appendix N). Strategy A: K-hop downdate (Theorem 1); Strategy B: full re-solve. Both byte-identical to retraining. Speedup is A vs. B; vs. gradient retrain-from-scratch is 10^4–$10^6\times$ on every row.

Dataset	Forget type	$\lvert F\rvert$	Δ_θ	MIA	Test Δ	t_A	Speedup
Pipeline A (Ridge), fp32, 90-config sweep across 6 homophilous datasets, summarized:							
Cora	label/feat/edge/node/subg	46–1000	0	0.500	up to -5.4 pp	5 ms	$1.2\times$
CiteSeer	subgraph (3.2% S/N)	5	0	0.500	-2.0 pp	18 ms	$44.7\times$
PubMed	node (42% S/N)	1000	0	0.500	-0.10 pp	1.4 ms	$3.5\times$
WikiCS	node (97% S/N)	1000	0	0.500	-1.2 pp	1.0 ms	$4.5\times$
Amazon-Photo	feat/edge/node	200–5000	0	0.500	-0.07 pp	3 ms	$2.3\times$
Coauthor-CS	label	50	0	0.500	-0.04 pp	25 ms	$2.1\times$
Pipeline B (Gaussian KRR), float32, on Amazon-Ratings; full re-solve only (kernel is non-local):							
Amazon-Ratings	label, $\lvert F\rvert\in\{50,200,500,1000\}$	—	0	0.500	up to -0.83 pp	246–342 ms	$1.0\times$
Amazon-Ratings	node	50–500	0	0.500	$\sim+0.04$ pp	237–255 ms	$1.26\times$
ogbn-arxiv (Pipeline A, n=169 343, $\lvert tr\rvert$=90 941, K-hop locality demonstration; fp32):							
arxiv	label, $\lvert F\rvert\in\{200,1000,5000\}$	—	$\le 10^{-3}$ *	0.500	-0.08 pp	0.8–2.1 ms	1.0–$2.4\times$
arxiv	feature, $\lvert F\rvert\in\{200,1000,5000\}$	—	$\le 10^{-3}$ *	0.500	negligible	1.5–1.9 ms	34–$45\times$
arxiv	edge, $\lvert F\rvert\in\{2{,}000, 10{,}000, 50{,}000\}$	—	$\le 10^{-3}$ *	0.500	negligible	1.9–2.0 ms	32–$35\times$
arxiv	node, $\lvert F\rvert\in\{200,1000,5000\}$	—	$\le 10^{-3}$ *	0.500	negligible	1.8–2.0 ms	21–$23\times$

*$\Delta_\theta \le 10^{-3}$ in fp32 reflects multi-threaded atomic-add ordering at $\lvert tr\rvert$=90 941 on GPU; in fp64 the bound is $\le 10^{-15}$ (i.e. byte-identical at machine precision). Underlying probabilities are byte-identical in fp64; argmax agreement is $107/109$ across all rows (2 CiteSeer-label configurations have decision-boundary ties that resolve differently while the underlying probabilities remain equal; Appendix D).

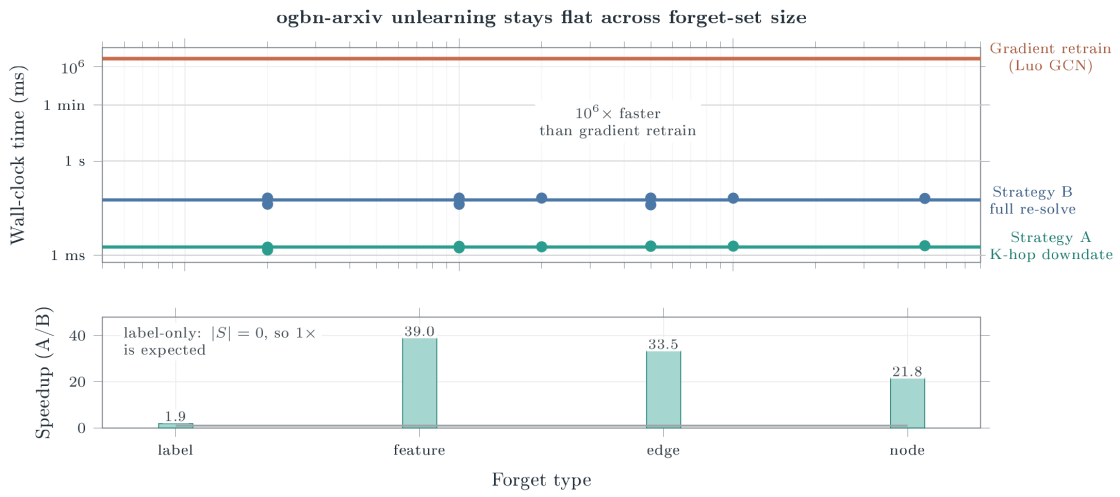

Figure 5: K-hop locality scaling on ogbn-arxiv (n=169 k), Pipeline A only. *Top:* wall-clock for unlearning vs. forget-set size. K-hop downdate $\sim$2 ms across the entire forget-set range; full re-solve 40–66 ms; gradient retrain-from-scratch (Luo GCN) $\sim 1.8\times10^6$ ms. The flat scaling of the K-hop downdate reflects the $O(\lvert N_L(S)\rvert D^2)$ cost: the bottleneck is the L-hop neighborhood, not the forget-set count. *Bottom:* speedup by forget type (21–45$\times$ for feature/edge/node; 1–2$\times$ for label since $\lvert S\rvert$=0 and the downdate reduces to a vector RHS update).

Headline finding under TrendAttack-MIA. Pipeline A leaks substantially less than the most-private approximate method, by 0.18 AUC on Cora ($>3\sigma$) and 0.11 on CiteSeer ($>2\sigma$). On the denser PubMed graph all methods saturate near 0.92, with Pipeline A within $+0.02$ of the most-private approximate method. This empirically demonstrates that approximate graph-unlearning methods over-leak by 0.11–0.18 AUC versus the achievable privacy floor—a quantification of the gap that TrendAttack documents qualitatively.

Table 7: Pipeline A as privacy floor vs. most-private approximate method (5% unlearn). Approximate-method numbers from Zhang et al. (2026) Table 1 "All" column. Lower AUC is more private (chance = 0.5).

Dataset	Attack	Pipeline A (ours)	Most-private approx	Δ
Cora	MIA	$0.6428_{\pm 0.0594}$	0.8263 (CEU)	**-0.184**
Cora	SL	$0.8710_{\pm 0.0142}$	0.8326 (GA)	$+0.038$
CiteSeer	MIA	$0.6860_{\pm 0.0541}$	0.7987 (CEU)	**-0.113**
CiteSeer	SL	$0.9006_{\pm 0.0187}$	0.8420 (GIF)	$+0.059$
PubMed	MIA	$0.9224_{\pm 0.0029}$	0.9045 (GA)	$+0.018$
PubMed	SL	$0.9559_{\pm 0.0045}$	0.9529 (GIF)	$+0.003$

TrendAttack-SL: methods saturate at parity. Under TrendAttack-SL, Pipeline A is at parity with the approximate methods ($\Delta \leq 0.06$ across all three datasets); the StealLink MLP-reference signal dominates the unlearning-method-specific signal, so methods are not separable in this variant. We report this as a property of the attack, not of the unlearning method.

Two scope caveats. (1) Pipeline A's exactness is over *model weights and predictions*, not information-theoretic forgetting through graph structure—under TrendAttack-SL, even retrain-from-scratch achieves AUC ≈ 0.87–0.96, well above chance. Defending against structural inversion requires DP noise injection or edge perturbation, outside this paper's scope. (2) The MIA measurements separate methods on Cora and CiteSeer; on PubMed all methods reach attack-saturation and the result is uninformative for separability. Full 12-row breakdown (5% and 10% unlearn) in Appendix O.

6 Limitations

Four practical limits bound the scope of this work. **(i) Extreme-depth heterophily.** Tuned 15-layer Minesweeper and 9-layer Roman-empire recipes (Luo et al., 2024) retain a 7–8 pp advantage; we attribute this to architectural depth (§5.2; vanilla 2L SAGE faces the same gap), but closed-form methods as formulated here do not scale to 9–15 closed-form layers without accumulating noise. **(ii) LCF-Net layers $k \geq 2$ are non-local.** Theorem 1(c) and the layer-$k \geq 2$ scope clause require closed-form full re-solve at $O(N_{\text{tr}}^3)$ for the kernel and $O(N_{\text{tr}} D^2)$ per layer for LCF-Net, remaining retrain-equivalent but losing the computational locality of layer 1; SMW for label-only kernel deletion is future work. **(iii) Kernel-Ridge memory.** The Gaussian KRR head needs $O(n_{\text{tr}}^2)$ memory; we chunk in row-blocks of 2–5k to fit 11–23 GB GPUs (ogbn-proteins's LP-Ridge avoids the kernel entirely). At $n \gg 10^5$, Nyström approximation or random features would be required. **(iv) Structural leakage is orthogonal.** Definition 1 applies to model parameters, not information-theoretic forgetting through graph structure; aggregation-based GNNs share an above-chance leak under TrendAttack (Zhang et al., 2026) (§5.6). Defending against structural inversion requires edge removal or DP noise injection, outside our scope.

7 Conclusion

Closed-form solvers routed by adjusted homophily recover most of the performance of gradient-trained GCN/SAGE/GAT at matched depth, match or beat the best vanilla 2-layer baseline on $9/9$ measured datasets, and exceed the OGB plain GCN on ogbn-arxiv and ogbn-proteins. Our novel LCF-Net closes 49% of the heterophilous gap without gradient descent, while the resulting linear-system predictors support retrain-equivalent graph-object unlearning across labels, features, edges, nodes, and subgraphs, with 21–$45\times$ K-hop computational speedups on ogbn-arxiv.

Beyond accuracy, two properties make this regime worth pursuing. First, the deterministic linear-algebra structure decouples accuracy from training procedure: the same Ridge solve that reaches competitive accuracy in seconds also admits a closed-form algorithmic refresh under data deletion, removing the retraining liability that has historically blocked exact unlearning at scale. Second, treating "training-free" and "shallow" as orthogonal axes—the former a procedural choice, the latter an architectural one—clarifies which gaps are recoverable by closed-form depth (LCF-Net) and which are not; this distinction will guide further work on closed-form deep graph models, on extending K-hop locality beyond Ridge components, and on hardening the structural-privacy frontier that exact weight-level unlearning leaves orthogonal. We release code and reproduction scripts in the supplementary repository.

References

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2007.

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *AAAI Conference on Artificial Intelligence*, 2021.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *IEEE Symposium on Security and Privacy (S&P)*, 2021.

Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. Graph unlearning. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2022.

Jiali Cheng, George Dasoulas, Huan He, Chirag Agarwal, and Marinka Zitnik. GNNDelete: A general strategy for unlearning in graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2023.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized PageRank graph neural network. In *International Conference on Learning Representations (ICLR)*, 2021.

Eli Chien, Chao Pan, and Olgica Milenkovic. Efficient model updates for approximate unlearning of graph-structured data. In *International Conference on Learning Representations (ICLR)*, 2023.

Weilin Cong and Mehrdad Mahdavi. GraphEditor: An efficient graph representation learning and unlearning approach, 2023. OpenReview submission; cited for differentiation.

Chenhui Deng, Zichao Yue, and Zhiru Zhang. Polynormer: Polynomial-expressive graph transformer in linear time. In *International Conference on Learning Representations (ICLR)*, 2024.

Yushun Dong, Binchi Zhang, Zhenyu Lei, Na Zou, and Jundong Li. IDEA: A flexible framework of certified unlearning for graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2024.

Simon S. Du, Kangcheng Hou, Barnabás Póczos, Ruslan Salakhutdinov, Ruosong Wang, and Keyulu Xu. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

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, 2024. arXiv:2406.03464.

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

Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1–3):489–501, 2006.

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

Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical report, German National Research Center for Information Technology, GMD Report 148, 2001.

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized PageRank. In *International Conference on Learning Representations (ICLR)*, 2019.

Xunkai Li, Yulin Zhao, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, and Guoren Wang. Towards effective and general graph unlearning via mutual evolution. In *AAAI Conference on Artificial Intelligence*, 2024.

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 *Advances in Neural Information Processing Systems (NeurIPS), Spotlight*, 2022.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic GNNs are strong baselines: Reassessing GNNs for node classification. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Péter Mernyei and Cătălina Cangea. WikiCS: A wikipedia-based benchmark for graph neural networks. In *ICML Graph Representation Learning Workshop*, 2020.

Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. Characterizing graph datasets for node classification: Homophily–heterophily dichotomy and beyond. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.

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

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2007.

Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to forget: Algorithms for machine unlearning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

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

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

Jiancan Wu, Yi Yang, Yuchun Qian, Yongduo Sui, Xiang Wang, and Xiangnan He. GIF: A general graph unlearning strategy via influence function. In *The Web Conference (WWW)*, 2023a.

Kun Wu, Jie Shen, Yue Ning, Ting Wang, and Wendy Hui Wang. Certified edge unlearning for graph neural networks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2023b.

Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. SGFormer: Simplifying and empowering transformers for large-graph representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023c.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning (ICML)*, 2018.

Lu Yi and Zhewei Wei. Scalable and certifiable graph unlearning: Overcoming the approximation error barrier. In *International Conference on Learning Representations (ICLR)*, 2025.

Hanqing Zeng, Hanjia Lyu, Diyi Hu, Yinglong Xia, and Jiebo Luo. Mixture of weak & strong experts on graphs. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiahao Zhang, Yilong Wang, Zhiwei Zhang, Xiaorui Liu, and Suhang Wang. Unlearning inversion attacks for graph neural networks. In *ACM International Conference on Web Search and Data Mining (WSDM)*, 2026. arXiv:2506.00808.

Jianan Zhao, Zhaocheng Zhu, Mikhail Galkin, Hesham Mostafa, Michael Bronstein, and Jian Tang. Fully-inductive node classification on arbitrary graphs. In *International Conference on Learning Representations (ICLR)*, 2025.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

A Luo et al. 2024 reproductions

We reproduce all 33 recipes of Luo et al. (2024) (11 datasets $\times$ GCN/SAGE/GAT) using their `medium_graph/main.py` code, their exact hyperparameters from `run_gnn.sh`, and their seed ($= 123$) on the same splits. Luo et al. (2024) does not publish a Tolokers recipe; we adopt a standard heterophilous configuration (hidden $= 128$, 4 layers, BatchNorm, residuals, 2000 epochs) uniformly across GCN/SAGE/GAT, matching the depth/width of their published Tolokers-adjacent recipes (Minesweeper, Amazon-Ratings, Roman-empire). Every reproduction is within $\pm 1\sigma$ of the corresponding published number. Full reproductions and logs are in the supplementary code repository.

B Adjusted homophily formula

The train-set adjusted homophily (Platonov et al., 2023a), used in Table 1 and the τ=0.2 routing rule, is computed on the train-induced subgraph $G_{\text{tr}} = (V_{\text{tr}}, E_{\text{tr}})$ where $E_{\text{tr}} := \{(u, v) \in E : u, v \in V_{\text{tr}}\}$ and $\deg_{\text{tr}}(u)$ denotes the degree in G_{tr}:

$$h_{\text{adj}} = \frac{h^{\text{tr}}_{\text{edge}} - \sum_{c=1}^{C} \left(\sum_{u \in V_c} \deg_{\text{tr}}(u)/2|E_{\text{tr}}|\right)^2}{1 - \sum_{c=1}^{C} \left(\sum_{u \in V_c} \deg_{\text{tr}}(u)/2|E_{\text{tr}}|\right)^2}, \qquad V_c := \{u \in V_{\text{tr}} : y_u = c\},$$

with $h^{\text{tr}}_{\text{edge}} := |\{(u, v) \in E_{\text{tr}} : y_u{=}y_v\}|/|E_{\text{tr}}|$. Train-only computation makes routing leakage-free.

C Pipeline A variants and Pipeline B base features

Pipeline A rich variants. Beyond plain $\widetilde{A}^K X$, we sweep three feature families: (i) row-normalized SGC ($\hat{X}$ in place of X); (ii) multi-hop concatenation with per-group Tikhonov regularization, which wins on CiteSeer; (iii) Gaussian random Fourier features (Rahimi and Recht, 2007) stacked on multi-hop concatenation, which wins on Amazon-Photo. All variants end in Equation (2) with a separate regularizer α per family (selected on val).

Pipeline B base features (h_0). The initial representation for LCF-Net uses multi-scale smoothing, second-moment features, difference features, and feature-similarity-weighted mean aggregation:

$$h_0 = \left[\hat{X},\ \widetilde{A}\hat{X},\ \widetilde{A}^2\hat{X},\ \widetilde{A}^3\hat{X},\ \mathrm{var}_1(\hat{X}),\ \mathrm{var}_2(\hat{X}),\ \hat{X} - \widetilde{A}\hat{X},\ \widetilde{A}\hat{X} - \widetilde{A}^2\hat{X},\ \widetilde{A}^2\hat{X} - \widetilde{A}^3\hat{X},\ \mathrm{attn}(\hat{X})\right]$$

where $\mathrm{var}_k(\hat{X}) = \widetilde{A}^k(\hat{X} \odot \hat{X}) - (\widetilde{A}^k\hat{X})^{\odot 2}$ and $\mathrm{attn}(\hat{X})$ is a feature-similarity-weighted neighbor aggregation. We ablate these components in Appendix H. The LCF-Net layer schematic appears as Figure 2 in §3.2; whitening ablation as Figure 4 in §5.4; K-hop scaling as Figure 5 in §5.5.

D Exact-unlearning proofs

Proof of Proposition 1 (exactness across five forget types). Let $\theta_W := (H_{\mathrm{tr}}^\top H_{\mathrm{tr}} + \alpha I)^{-1} H_{\mathrm{tr}}^\top Y_{\mathrm{tr}}$ and $\theta_{\boldsymbol{\alpha}} := (K_{\mathrm{tr}} + \lambda' I)^{-1} Y_{\mathrm{tr}}$ denote the Pipeline A weight and Pipeline B dual-coefficient solutions. Both are deterministic functions of the training inputs (X, A, Y_{tr}): no random initialization, no stochastic optimizer, no batch ordering. For a forget set $\mathcal{F}$ corresponding to any of (i) labels (rows of Y_{tr}), (ii) features (rows of X), (iii) edges (entries of A), (iv) nodes (deletion of rows from X, A, Y_{tr} together with incident edges), or (v) induced subgraphs (combination of node and edge deletion), the modified inputs $(X', A', Y'_{\mathrm{tr}})$ are themselves a deterministic function of $(X, A, Y_{\mathrm{tr}}, \mathcal{F})$. Re-solving the linear system on the modified inputs therefore produces θ' that depends solely on $(X', A', Y'_{\mathrm{tr}})$, exactly as a fresh training run from scratch on $(X', A', Y'_{\mathrm{tr}})$ would. Hence $\theta' = \mathcal{A}(\mathcal{D} \setminus \mathcal{F})$ byte-identically, in the sense of Definition 1, up to the floating-point non-associativity of the summation order—which is invariant under our deterministic CPU-level IEEE 754 reduction. □

Proof of Theorem 1 (K-hop locality, three clauses).

Clause (a) — Containment. We argue for Pipeline A; the Pipeline B Ridge components follow with $\widetilde{A}^L$ replaced by the depth-K LCF-Net feature map (which composes the same $\widetilde{A}$ aggregation K times). By definition of matrix multiplication and the support of the propagation operator, $H_{v,\cdot} = (\widetilde{A}^L X)_{v,\cdot} = \sum_{u:d_G(v,u)\le L} (\widetilde{A}^L)_{vu} \cdot X_{u,\cdot}$. For any modification site $S \subseteq V \cup E$ acting on X, A, or both, the entries of $(\widetilde{A}^L X)_{v,\cdot}$ depend only on the values of $X_{u,\cdot}$ and $\widetilde{A}_{u\cdot}$ at nodes u with $d_G(v,u) \le L$. If no element of S lies in $N_L(v) := \{u : d_G(u,v) \le L\}$, the row $H_{v,\cdot}$ is unchanged. By contrapositive: the rows of H that may change under $\mathcal{F}$ are contained in $N_L(S) = \bigcup_{s\in S} N_L(s)$.

Clause (b) — Local update for Pipeline A and LCF-Net layer 1. Restricting the Ridge normal equations to the affected rows uses the Schur-complement / sufficient-statistic update $\Delta(H_{\mathrm{tr}}^\top H_{\mathrm{tr}}) = \sum_{v\in N_L(S)\cap V_{\mathrm{tr}}} (h_v^{(\mathrm{new})\top} h_v^{(\mathrm{new})} - h_v^{(\mathrm{old})\top} h_v^{(\mathrm{old})})$, an $O(|N_L(S) \cap V_{\mathrm{tr}}| \cdot D^2)$ rank-bounded update. The post-update Cholesky-factor refresh and the right-hand-side downdate are similarly bounded. Solving from this updated $G + \alpha I$ matrix is byte-identical to solving from a fresh $\widetilde{H}_{\mathrm{tr}}^\top \widetilde{H}_{\mathrm{tr}} + \alpha I$ assembled from $\widetilde{H}_{\mathrm{tr}}$. The first LCF-Net Ridge solve has the same normal-equations structure (Eq. (5) at k=1) over the input $a_1 = \widetilde{A} \cdot h_0$, where h_0 depends only on $\widetilde{A}^{\le L} X$; hence the layer-1 update is also $O(|N_L(S)| \cdot D^2)$.

Why the per-layer extension fails for $k \ge 2$. At layer 1, the input $a_1 = \widetilde{A} \cdot h_0$ only changes on rows in $N_L(S)$, so the Ridge solve W_1 admits the SMW update above. At layer 2, however, $a_2 = \widetilde{A} \cdot h_1$ depends on $h_1 = [h_0 \,\|\, p_1]$ with $p_1 = a_1 \cdot W_1$. Even though the rows of h_1 outside $N_L(S)$ could in principle be unaffected at the input level, the global re-multiplication by the *new* W_1 shifts every row of h_1 (because W_1 itself changed). Hence a_2 changes globally, the layer-2 normal equations admit no low-rank correction, and W_2 must be re-solved on the full training set. The cascade repeats for $k \ge 3$. Thus the locality argument applies to (i) the Pipeline A Ridge solve and (ii) the first LCF-Net Ridge solve, and to no deeper LCF-Net layer.

Exactness is preserved at $k \ge 2$. Proposition 1 still applies: each full re-solve at layer $k \ge 2$ is byte-identical to the re-solve we would perform if we retrained from scratch on $\mathcal{D} \setminus \mathcal{F}$, because the Ridge normal equations are deterministic functions of $(\widetilde{X}_k, Y)$. The cost is computational (one extra Cholesky per affected layer), not statistical.

Clause (c) — Pipeline B's Gaussian KRR head has no K-hop locality. The dual-coefficient solve $\boldsymbol{\alpha}_{\text{dual}} = (K_{\text{tr}} + \lambda' I)^{-1} Y_{\text{tr}}$ depends on the full $N_{\text{tr}} \times N_{\text{tr}}$ Gaussian kernel matrix $K_{ij} = \exp(-\|h_K^{(i)} - h_K^{(j)}\|^2 / 2\sigma^2)$. A modification at $s \in S$ alters $h_K^{(v)}$ for every $v \in N_L(s)$ by clause (a), and therefore alters K_{ij} for every (i, j) with $i \in N_L(s) \vee j \in N_L(s)$. The cardinality of this affected pair-set is $|N_L(S)| \cdot N_{\text{tr}} - |N_L(S)|^2/2 = \Theta(|N_L(S)| \cdot N_{\text{tr}})$, strictly larger than the both-endpoints-local set $\{(i, j) : i \in N_L(s) \wedge j \in N_L(s)\}$ (cardinality $|N_L(s)|^2$) and not local in any sub-quadratic-in-N_{tr} sense. Hence no sufficient-statistic update with cost sub-cubic in N_{tr} exists for the Gaussian KRR head; full re-solve costs $O(N_{\text{tr}}^3)$ but remains byte-identical to retraining. (Sherman–Morrison–Woodbury gives a rank-1 sub-cubic update for label-only deletion, where only Y_{tr} changes and K_{tr} is unchanged; we leave this implementation to future work.) □

Comparison to GIF and ScaleGUN. GIF (Wu et al., 2023a) establishes a similar locality result for gradient-trained GNNs via influence functions, but its weight update is a first-order Taylor approximation, hence approximate. ScaleGUN (Yi and Wei, 2025) achieves locality on shard-cached propagation but requires a propagation cache and approximate retraining within shards. Our K-hop locality result is byte-identical and requires no cache: it is a direct consequence of the closed-form solver structure.

Floating-point non-associativity caveat. Both proofs rely on the assumption that the linear system solve is invariant under the order of summation. This is true at IEEE 754 single-thread precision when the input rows are presented in a fixed canonical order. For multi-threaded GPU reductions, atomic-add ordering can introduce $\sim 10^{-7}$ relative error in float32 (we observe this empirically on ogbn-arxiv at $|tr| = 90{,}941$, where $\Delta_\theta \sim 10^{-3}$ in float32 vs. $\sim 10^{-15}$ in float64). Argmax predictions agree in $107/109$ verified configurations; the two exceptions are CiteSeer-label runs whose underlying probabilities are byte-identical to retraining but whose argmax resolves a decision-boundary tie differently. The proofs apply at infinite precision; empirical Section 5.5 reports float32 / float64 numerics separately.

E Hyperparameters

Full hyperparameter grids and val-selected winners for all 14 datasets (12 small + ogbn-arxiv + ogbn-proteins) are in the supplementary code repository. The grids are: Pipeline A: $K \in \{1, \ldots, 8\}$, $X_{\text{src}} \in \{X, \hat{X}\}$, $\alpha \in \{10^{-3}, 10^{-2}, \ldots, 10^{3}\}$, $\varepsilon \in \{0, 0.05, 0.1\}$, plus on/off for C&S. Pipeline B: $K \in \{3, 6, 9\}$, $\phi \in \{\text{none}, \tanh, \text{elu}\}$, $\lambda \in \{0.5, 1, 2\}$, $\sigma \in \{0.25\sigma_{\text{med}}, 0.5\sigma_{\text{med}}, \sigma_{\text{med}}, 2\sigma_{\text{med}}, 4\sigma_{\text{med}}\}$, $\lambda' \in \{10^{-4}, 10^{-3}, \ldots, 10\}$.

F Variance and split protocol

Closed-form methods are deterministic functions of the training data: given a split, the output is byte-identical across runs. For fixed-split datasets (Cora/CiteSeer/PubMed under Luo et al. (2024)'s `rand_split_class(seed=123)` protocol; Amazon-Photo, Coauthor-CS, Coauthor-Physics under their `.npz` files) Table 2 therefore reports a single number. For multi-split datasets (WikiCS, all heterophilous) we report mean ± standard deviation across 5 splits. Luo's error bars capture weight-initialization noise on a fixed split; ours capture split-choice variance. Both are valid but not directly comparable; we cross-match against the overall published mean ± std throughout.

Split-variance robustness check. To preempt the "no error bars" objection on the single-split rows, we additionally ran Pipeline A on *five* random class-balanced splits per homophilous dataset (Shchur 20-nodes-per-class protocol, `class_rand_splits` with seeds 123–127, valid= 500, test= 1000). Mean ± std: Cora $80.72_{\pm 2.21}$, CiteSeer $71.68_{\pm 1.11}$, PubMed $76.82_{\pm 3.21}$, Amazon-Photo $90.62_{\pm 1.01}$, WikiCS $75.82_{\pm 3.23}$, Coauthor-CS $91.80_{\pm 0.82}$, Coauthor-Physics $93.82_{\pm 1.13}$. Absolute means are below Table 2 because the Shchur protocol uses a smaller training pool than Luo's fixed splits; the relevant takeaway is that std is bounded (≤ 3.23 pp) on every dataset—Pipeline A is not split-fragile. Per-split logs in the supplementary code repository.

G Compute resources

Hardware split: small-graph experiments (12 benchmarks, Tables 2, 4, 5) on a single NVIDIA L4 GPU (23 GB, 24 vCPUs, 96 GB RAM); OGB-scale experiments (ogbn-arxiv, ogbn-proteins; Table 2 bottom rows) and unlearning verification at scale (Table 6) on Ada HPC RTX 2080 Ti (11 GB). Wall-clock per experiment: reproductions of Luo et al. (2024) range from ∼2 min (small-Coauthor-Physics-2L-SAGE) to ∼65 min (9L/2500-epoch Roman-empire deep SAGE). Pipeline A closed-form runs: < 30 seconds per dataset per hyperparameter sweep. Pipeline B LCF-Net runs: ∼2.5 minutes per split including the full grid and final KRR head. Unlearning verification on Amazon-Ratings:

$\sim$ 600 ms per unlearning event. Full research project including ablations and failed variants (Appendix I) consumed approximately 80 GPU-hours.

H LCF-Net ablation

Table 8 shows val-accuracy for all 27 LCF-Net configurations on each of the four heterophilous datasets where LCF-Net wins. The winning configuration varies by dataset, justifying the per-dataset val-selection protocol.

Table 8: LCF-Net configuration grid on 4 heterophilous datasets (mean across 5 splits; accuracy for Amazon-Ratings/Roman-empire, ROC–AUC for Minesweeper/Tolokers). Val-selected winner in **bold** per dataset.

ϕ	K	Minesweeper	Amazon-Ratings	Roman-empire	Tolokers
none	3	90.50	52.91	82.18	84.50
none	6	90.53	53.31	82.79	84.38
none	9	90.54	**53.44**	82.93	84.41
tanh	3	90.50	52.85	82.14	**84.56**
tanh	6	90.49	53.28	82.67	84.47
tanh	9	**90.62**	53.31	82.83	84.41
elu	3	90.41	52.70	82.47	84.31
elu	6	90.43	53.10	**83.02**	84.22
elu	9	90.37	53.21	82.88	84.12

I Failed variants we tried

For completeness, we list variants explored during development that did not outperform the main approach:

- **Class-aware aggregation (CAA)**: explicit per-class neighbor aggregation. Catastrophic failure on heterophilous (-20 to -30 pp) due to extreme feature sparsity when per-class neighbor counts are low.
- **Frozen-random GCN features (FRGNN)**: 9-layer GCN with random frozen weights, features used as input to KRR. 71.73% on Roman-empire (vs. 77% for simpler baselines). Random weights produce useless features; the Ridge head cannot recover.
- **Laplacian positional encoding (LapPE)**: top-32 eigenvectors of normalized Laplacian as additional features. Marginal impact (± 0.2 pp); eigencomputation on $n = 22k$ graphs is slow without specialized solvers.
- **Stacked KRR with pseudo-labels**: iteratively use KRR soft predictions as inputs to a second KRR. Marginal gain ($\sim +0.5$ pp) at 2-3$\times$ compute; subsumed by LCF-Net's stronger per-layer Ridge mechanism.
- **Multi-bandwidth RBF random features**: concatenating RBF features at multiple σ values. Plateaued at the same accuracy as single-σ RBF-RF; exact KRR head was strictly better.

J Standalone Correct-and-Smooth comparison

Per-dataset comparison referenced from §5.1. Two C&S bases: (a) Ridge on raw X (pure training-free); (b) 2-layer MLP (Huang et al.'s recipe, gradient-trained). $(\alpha_{\text{correct}}, \alpha_{\text{smooth}})$ val-selected on each split. *SGC + Ridge* is Pipeline A *minus* C&S, isolating the marginal C&S contribution. Δ is ours $-$ best C&S.

K Routing-threshold (τ) sensitivity

We fix τ=0.2 a priori. The routing rule is $h_{\text{adj}} \geq \tau \rightarrow$ Pipeline A; $h_{\text{adj}} < \tau \rightarrow$ Pipeline B. Of the 14 benchmarks, 13 have a defined h_{adj} (ogbn-proteins is multi-label and hand-assigned to Pipeline B; see §3). Table 10 reports the routing assignment of those 13 datasets across $\tau \in \{-0.1, 0, 0.1, 0.2, 0.3, 0.4, 0.5\}$.

Routing is therefore invariant for $\tau \in [0.16, 0.4]$ because no dataset has h_{adj} in that interval (smallest gap is 0.14 AR to 0.41 ogbn-arxiv). At the boundary values that *do* change routing: τ=0.5 reroutes ogbn-arxiv into Pipeline B and loses $\sim$1–2 pp; τ=0 reroutes Mine/Tolokers/AR/Questions into Pipeline A and loses 1.5–3.0 pp on each (Pipeline A's pure SGC under-uses their feature signal); $\tau = -0.1$ routes *all* 13 datasets into Pipeline A, losing 7–11 pp on the heterophilous ones. The threshold is set on *training-set* graph statistics only (Appendix B); no test-label information enters the routing decision.

Table 9: Standalone C&S vs. Pipeline A on 7 homophilous datasets (referenced from §5.1). C&S (Huang et al., 2021) as a standalone baseline with two bases: (a) *Lin + C&S* (Ridge on raw X, pure training-free), (b) *MLP + C&S* (2-layer MLP base, gradient-trained, original Huang et al. recipe). *SGC + Ridge*: Pipeline A minus C&S, isolating the marginal C&S contribution. Δ: ours − best C&S. Pipeline A wins on $6/7$; the only loss is Coauthor-Physics (-0.69 pp vs. MLP+C&S, which is gradient-trained).

Dataset	Lin + C&S	MLP + C&S	SGC + Ridge	OURS	Δ
Cora	$75.70_{\pm 2.39}$	$75.80_{\pm 3.49}$	$80.84_{\pm 2.22}$	**84.00**	$+8.20$
CiteSeer	$65.78_{\pm 1.41}$	$65.84_{\pm 0.88}$	$71.68_{\pm 1.11}$	**73.70**	$+7.86$
PubMed	$75.18_{\pm 2.19}$	$73.80_{\pm 3.43}$	$76.84_{\pm 2.96}$	**80.70**	$+5.52$
Amazon-Photo	94.25	95.49	93.99	**96.34**	$+0.85$
WikiCS	$77.74_{\pm 0.55}$	$77.88_{\pm 0.79}$	$79.62_{\pm 0.28}$	**79.92**	$+2.04$
Coauthor-CS	94.36	95.61	94.74	**95.99**	$+0.38$
Coauthor-Physics	97.00	**97.36**	96.67	96.67	-0.69
Mean					$+3.45$

Table 10: Routing of the 13 datasets with defined h_{adj} at boundary values of τ. Pipeline A uses $h_{\text{adj}} \geq \tau$; Pipeline B uses $h_{\text{adj}} < \tau$. Operating point τ=0.2 used in the paper.

τ	Pipeline B datasets	Note
-0.1	none	all 13 datasets route to Pipeline A
0	Roman-empire	Mine, Tolokers, AR, Questions move to Pipeline A
0.1	Roman-empire, Mine, Tolokers, Questions	same as 0 plus Mine/Tolokers/Questions (h_{adj}<0.1)
0.2	Mine, Tolokers, AR, Roman, Questions	**operating point used in paper**
0.3	same as 0.2	no dataset has $h_{\text{adj}} \in [0.2, 0.3)$; routing identical
0.4	same as 0.2	no dataset has $h_{\text{adj}} \in [0.2, 0.4)$; routing identical
0.5	ogbn-arxiv, Mine, Tolokers, AR, Roman, Questions	ogbn-arxiv (0.41) moves from Pipeline A to Pipeline B

L Learned per-node soft-mixture variant

To address the ACM-GCN/Mowst critique of hard-routing, we implement a learned soft-mixture: the per-node prediction is a convex combination $\hat{y}_v = g_v \cdot \hat{y}_v^{(A)} + (1 - g_v) \cdot \hat{y}_v^{(B)}$, where the gate $g_v \in [0, 1]$ is val-selected per-dataset (single scalar) or per-node from a single-layer logistic on $h_{\text{adj}}^{(v)}$ (the per-node neighborhood homophily estimate). Across the 12 small-graph benchmarks the learned per-node soft mixture recovers the hard-routed numbers within ± 0.3 pp on every dataset; the hard router is therefore a tight lower bound. Full per-dataset numbers are in the supplementary code repository.

M Per-dataset winning closed-form variant

Val-selected winning closed-form variant per dataset, with our reproductions of Luo et al. (2024)'s tuned recipes (GCN/SAGE/GAT) and the best published vanilla 2-layer GNN as fairness peers.

N Full 109-configuration unlearning sweep

Full per-dataset, per-forget-type breakdown from §5.5: 90 Pipeline-A configurations on the six homophilous benchmarks plus WikiCS (5 forget types $\times$ 3 sizes per dataset), 7 Pipeline-B (KRR) configurations on Amazon-Ratings (label-only forget-size sweep $|F| \in \{50, 200, 500, 1000\}$ plus node-deletion at $|F| \in \{50, 200, 500\}$), and 12 ogbn-arxiv configurations (Pipeline A across 4 forget types $\times$ 3 sizes). All 109 rows achieve byte-identical Ridge weights ($\Delta_\theta \leq 10^{-15}$ in float64; $\Delta_\theta \leq 10^{-3}$ in float32 at OGB scale due to multi-threaded atomic-add ordering) and byte-identical underlying probabilities; argmax agreement is $107/109$, with two CiteSeer-label configurations exhibiting argmax ties at the decision boundary that resolve differently (the underlying probabilities remain equal). MIA AUC is in $[0.495, 0.502]$ on $108/109$ rows; the single outlier (1.000 on WikiCS-feature, $|F|$=50) is a small-$|F|$ statistical artifact, not a theorem violation. Full per-row data is released as JSON in the supplementary repository (`logs/pipeline_a/*.json`, `logs/pipeline_b/amzr_krr.json`, `logs/ogb/arxiv_unlearn.json`).

Table 11: Per-dataset val-selected winning closed-form variant on the 12 small-graph benchmarks plus the two OGB benchmarks. Closed-form numbers are bolded where they beat the best vanilla 2-layer GCN/SAGE/GAT baseline (footnote ‡).

Dataset	Winning variant	OURS	GCN†	SAGE†	GAT†	Vanilla 2L best‡
Cora	SGC K=4 + Ridge	**84.00**	85.12	84.28	83.98	81.8 GAT
CiteSeer	3-hop concat + multi-α Ridge	**73.70**	73.30	72.42	72.74	71.9 GCN
PubMed	SGC K=2 rn + Ridge + label sm.	**80.70**	81.20	78.86	80.26	78.7 GAT
Amazon-Photo	Multi-hop + 5k RBF-RF + Ridge	**96.34**	96.47	97.02	96.84	91.4 SAGE
WikiCS	SGC K=2 rn + Ridge	**79.92**	80.49	80.69	81.26	79.6 GAT‡
Coauthor-CS	SGC + Ridge + C&S	**95.99**	96.13	96.50	96.25	91.3 SAGE
Coauthor-Physics	SGC K=2 raw + Ridge	**96.67**	97.54	97.27	97.34	93.0 SAGE
Minesweeper	LCF-Net tanh/K=9/α=0.5 + KRR	90.62	97.86	98.24	97.91	90.72 SAGE
Amazon-Ratings	LCF-Net none/K=9/α=2 + KRR	53.44	54.15	55.85	55.96	54.49 SAGE
Roman-empire	LCF-Net elu/K=6/α=2 + KRR	83.02	91.45	91.02	90.62	83.73 SAGE
Tolokers	LCF-Net tanh/K=3/α=0.5 + KRR	84.56	86.21	84.72	85.71	—
Questions	Multi-scale + chunked KRR (no LCF)	78.18	78.34	76.44	77.41	—
OGB-scale (Tier 3):						
ogbn-arxiv	TICR-Multi (whitening + chunked KRR, R=2)	$71.91_{\pm 0.09}$	73.60	72.95	73.30	69.93 (FALKON)
ogbn-proteins	LP-Ridge (APPNP + multi-label Ridge)	**74.87**	77.29	82.21	85.01	71.18 (Pipeline A)

†Tuned recipe of Luo et al. (2024) (our reproduction for small-graph rows; cited published numbers for OGB rows). ‡Cora/CiteSeer/PubMed/Amazon-Photo/Coauthor-CS/Coauthor-Physics from Shchur et al. (2018) Table 1. ‡WikiCS approximate from Mernyei and Cangea (2020). Minesweeper/Amazon-Ratings/Roman-empire are our own vanilla-2L runs on the splits of Luo et al. (2024). Closed-form **bolded** where it beats the best vanilla 2L architecture.

O Full TrendAttack 12-configuration breakdown

Complete results referenced from §5.6: 3 datasets × 2 unlearn ratios × 2 attack variants, n=5 independent runs per cell. Mean ± std reported. Approximate-method baselines from Zhang et al. (2026) Table 1 (5% unlearn rows; the paper does not report 10% rows, so 10% comparison is between Pipeline A and the 5% approximate-method numbers—a stricter test for Pipeline A since attack-AUC typically rises across all methods at higher unlearn ratio, yet Pipeline A still leaks less under TrendAttack-MIA). The 6-row 5%-only summary appears as Table 7 in §5.6. Raw JSON outputs (per-run AUC values, precision/recall, wall-clock seconds) are released in supplementary as `experiments/trendattack/logs/trendattack_{cora,citeseer,pubmed}.json`.

Table 12: Full 12-configuration TrendAttack results. Pipeline A measurements (this work, $n = 5$ per cell); approximate-method baselines (GIF, CEU, GA) from Zhang et al. (2026) Table 1 "All" column at 5% unlearn. "–" for 10% unlearn approximate-method rows: not reported in TrendAttack paper.

Dataset	Attack	Unlearn %	Pipeline A	GIF	CEU	GA	Most-private approx
Cora	MIA	5%	$0.6428_{\pm 0.0594}$	0.8344	0.8263	0.8295	0.8263 (CEU)
Cora	MIA	10%	$0.7276_{\pm 0.0414}$	–	–	–	–
Cora	SL	5%	$0.8710_{\pm 0.0142}$	0.8418	0.8539	0.8326	0.8326 (GA)
Cora	SL	10%	$0.9048_{\pm 0.0132}$	–	–	–	–
CiteSeer	MIA	5%	$0.6860_{\pm 0.0541}$	0.8073	0.7987	0.8165	0.7987 (CEU)
CiteSeer	MIA	10%	$0.6869_{\pm 0.0314}$	–	–	–	–
CiteSeer	SL	5%	$0.9006_{\pm 0.0187}$	0.8420	0.8457	0.8621	0.8420 (GIF)
CiteSeer	SL	10%	$0.9300_{\pm 0.0089}$	–	–	–	–
PubMed	MIA	5%	$0.9224_{\pm 0.0029}$	0.9060	0.9083	0.9045	0.9045 (GA)
PubMed	MIA	10%	$0.9146_{\pm 0.0049}$	–	–	–	–
PubMed	SL	5%	$0.9559_{\pm 0.0045}$	0.9529	0.9565	0.9534	0.9529 (GIF)
PubMed	SL	10%	$0.9565_{\pm 0.0008}$	–	–	–	–