CURVATURE FINDS THE POISON: ONE-SHOT ADVERSARIALLY ROBUST GRAPH LOTTERY TICKETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph lottery tickets, a sparse adjacency matrix paired with a sparse graph neural network, deliver the inference and memory savings of pruning at little cost to clean accuracy. Their robustness, however, is brittle. When an adversary flips a small number of edges before training, such a ticket loses a large amount of accuracy. The only existing remedy restores robustness, but only by pruning and retraining many times, by assuming that neighbours share a label, and by relying on pseudo-labels. We show that a single second-order measurement of the loss on the observed graph is enough. Curvature-Aware Adversarial Winning Tickets (CA-AWT) estimates the Hessian diagonal once after a short warm-up and derives both masks from it. Weights whose removal would raise the loss are kept and edges with high curvature are removed, because an injected adversarial edge is precisely one along which the loss is fragile, so the edge ranking acts as an attack detector. This is what a learned mask cannot see. Averaged over three poisoning attacks and four perturbation rates, curvature recovers 88% of the injected edges and over 98% at high sparsity, while a mask learned by standard sparsification recovers 30%, below random pruning. A teacher distilled from the curvature-cleaned graph, a global cross-layer budget and a budget schedule on the fixed ranking complete the recipe. The resulting sparse subnetwork carries Lipschitz, complexity and PAC-Bayes generalization guarantees that hold regardless of how the mask is chosen. Across citation graphs of increasing scale, one-shot CA-AWT beats the standard graph lottery ticket in 98% of cells and matches the only robust method, which prunes and retrains many times, while assuming nothing about homophily. The detector transfers unchanged to a strongly heterophilous graph.

1 INTRODUCTION

Graph neural networks (GNNs) are the standard model for relational data (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017), and as graphs grow to millions of nodes and edges, however, dense GNNs incur a heavy cost in memory, inference latency, and energy, which motivates a growing body of work on sparsification. The Lottery Ticket Hypothesis (Frankle & Carbin, 2019) showed that a dense network contains a sparse subnetwork that, trained in isolation from an early checkpoint, matches the dense model. For GNNs this idea extends naturally to two sparsities at once. A Graph Lottery Ticket (GLT) is a pair of a sparse adjacency matrix and a sparse set of weights that together preserve accuracy while cutting the compute of message passing (Chen et al., 2021).

A separate line of work has established that GNNs are acutely vulnerable to adversarial perturbations of the graph structure (Zügner et al., 2018; Zügner & Günnemann, 2019; Xu et al., 2019). In the poisoning threat model an adversary flips a small fraction of edges once, before training, and the defender sees only the perturbed graph. Because message passing propagates information along edges, structural perturbations are markedly more damaging than feature perturbations (Zügner & Günnemann, 2019), and a model trained naively on the poisoned graph inherits the attack. A range of defenses learns to clean or re-weight the structure (Jin et al., 2020; Zhang & Zitnik, 2020; Entezari et al., 2020), but these operate on dense models and do not deliver the efficiency of a lottery ticket.

The two goals, sparsity and robustness, do not go hand in hand. Rather, they are often in conflict. Recent work shows that GLTs obtained by standard sparsification are themselves highly fragile under structure poisoning, dropping sharply in accuracy (Chowdhury et al., 2023). The only existing remedy, ARGS (Chowdhury et al., 2023), restores robustness by jointly pruning the adjacency matrix

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

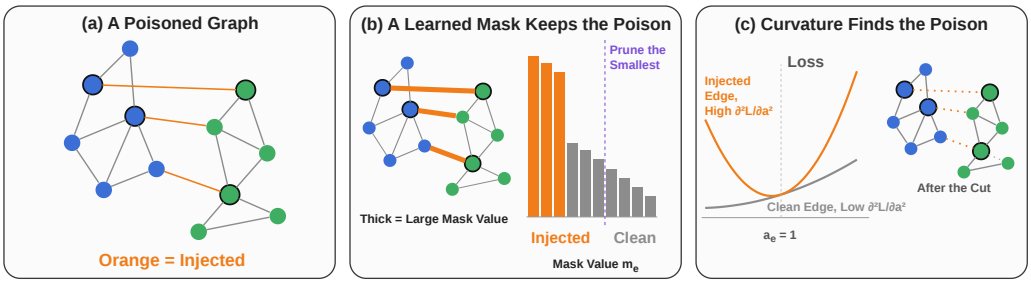

Figure 1: **Why a learned mask keeps the poison and curvature removes it.** (a) The defender receives a graph containing attacker-injected edges. (b) A mask learned on that graph assigns the injected edges the largest values, because they help fit the poisoned labels, so pruning the smallest entries keeps them. (c) Along an injected edge the loss has the same slope as along a clean edge but far higher curvature, so one Hessian-diagonal pass removes the injected edges first.

and the weights while optimizing a homophily-aware objective that combines the true labels of the training nodes with pseudo-labels on the test nodes, applied iteratively over many prune-and-retrain rounds. Each of the many rounds requires a full retraining, the pseudo-labels must be recomputed as the model changes, and the homophily assumption is baked into the loss. This raises a natural question that we take as the starting point of this paper.

Can a robust graph lottery ticket be found in a single shot? We argue that it can. On a poisoned graph the injected edges are the ones the attacker chose to move the training loss, so the loss is fragile along them, and fragility is what curvature measures (Figure 1c). The same curvature tells us which weights the network can spare. One estimate of the Hessian diagonal, taken once after a short warm-up, therefore ranks both edges and weights, with opposite rules. High-curvature edges are removed, and weights with high Optimal-Brain-Damage saliency (LeCun et al., 1990) are kept. This is also what a learned mask cannot do. The mask is trained to minimize the poisoned loss, and the injected edges help to minimize it, so the mask gives them large values and a prune-the-smallest rule keeps them (Figure 1b). Curvature looks at the second derivative instead of the first, and for an injected edge the two point in opposite directions. On citation benchmarks, the curvature score recovers 88% of the injected edges on average, while a learned mask recovers 30%, less than random pruning would.

We introduce Curvature-Aware Adversarial Winning Tickets (CA-AWT), which turns this observation into a one-shot method, and make the following contributions.

- A curvature criterion that reads redundant weights and adversarial edges from one estimate of the Hessian diagonal, taken once after a short warm-up. We characterize exactly which edges any label-driven criterion can see, so the blind spot of the score is a stated and bounded property rather than a hidden failure mode.
- A one-shot framework that extracts a robust ticket from the curvature scores. A global cross-layer budget makes extreme sparsity attainable, a budget schedule on the fixed ranking keeps the method one-shot, and a dense teacher trained on the curvature-cleaned graph is distilled into the sparse student. We prove that the summed directed curvature is the second-order cost of removing an edge, that the annealed mask is uniformly stable, and that the ticket keeps Lipschitz, complexity and PAC-Bayes guarantees for any mask.
- An evaluation on citation graphs of three scales under three poisoning attacks and four perturbation rates. Curvature recovers 88% of the injected edges on average. CA-AWT beats UGS in 98% of the cells and tracks the iterative ARGS baseline with a single scoring pass and no homophily assumption, and the detector transfers unchanged to a strongly heterophilous graph.

2 Related Work

Graph lottery tickets (Chen et al., 2021; Hui et al., 2023; Wang et al., 2023) prune the adjacency and the weights together for the clean setting, and Chowdhury et al. (2023) show that they inherit and can amplify the fragility of GNNs to structure poisoning (Zügner et al., 2018; Zügner & Günnemann, 2019; Xu et al., 2019). Defences against poisoning clean or re-weight the structure (Wu et al., 2019; Entezari et al., 2020; Jin et al., 2020; Zhang & Zitnik, 2020; Li et al., 2023); most encode a

homophily or low-rank prior and none yields a sparse ticket. ARGS (Chowdhury et al., 2023; 2024) is the only method that pursues robustness and double sparsity together, through a homophily-aware mask objective re-learned in every prune-and-retrain round, and its authors name non-homophilous graphs as the open limitation. Second-order pruning dates to Optimal Brain Damage (LeCun et al., 1990; Hassibi & Stork, 1993) and is made affordable by Hutchinson's estimator and Hessian-vector products (Hutchinson, 1990; Pearlmutter, 1994; Yao et al., 2020); on graphs, curvature has so far meant the discrete Ricci curvature of the structure (Topping et al., 2022), not the curvature of the training loss along an edge. A full discussion is in Appendix A.1.

3 PRELIMINARIES

Notation. We consider an undirected graph $G = (\mathcal{V}, \mathcal{E})$ with $N = |\mathcal{V}|$ nodes, adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$ and node features $\mathbf{X} \in \mathbb{R}^{N\times F}$. Message passing treats each undirected edge $\{u, v\}$ as two directed entries, $u \to v$ and $v \to u$; we write $\vec{\mathcal{E}}$ for the set of directed entries, so $|\vec{\mathcal{E}}| = 2|\mathcal{E}|$. A graph neural network $f_{\boldsymbol{\theta}}(\mathbf{A}, \mathbf{X}) \in \mathbb{R}^{N\times C}$ with weights $\boldsymbol{\theta} = \{\mathbf{W}_1, \ldots, \mathbf{W}_L\}$, flattened to a vector in $\mathbb{R}^d$, produces class log-probabilities for C classes. Writing $\mathcal{V}_{\mathrm{tr}}$ for the labelled nodes and y_i for their labels, the supervised loss is,

$$\mathcal{L}(\boldsymbol{\theta}; \mathbf{A}, \mathbf{X}) = \frac{1}{|\mathcal{V}_{\mathrm{tr}}|} \sum_{i\in\mathcal{V}_{\mathrm{tr}}} \ell\big(f_{\boldsymbol{\theta}}(\mathbf{A}, \mathbf{X})_i,\, y_i\big), \tag{1}$$

where ℓ is the cross-entropy. The adjacency is binary, so $\mathcal{L}$ has no derivative with respect to an edge. We therefore attach a continuous weight a_e to every directed entry $e \in \vec{\mathcal{E}}$, collected in $\mathbf{a} \in \mathbb{R}^{|\vec{\mathcal{E}}|}$, and write the forward pass as $f_{\boldsymbol{\theta}}(\mathbf{a}, \mathbf{X})$. The vector $\mathbf{a}$ is not the adjacency matrix. It holds one weight per edge present in $\mathbf{A}'$ and nothing for absent edges. At $\mathbf{a} = \mathbf{1}$ every present edge has weight one, so the model sees exactly $\mathbf{A}'$.[1] All structural derivatives are taken at this point. The vector $\mathbf{a}$ is never updated by gradient descent, in the warm-up or in the retraining, unlike the trainable edge mask of UGS and ARGS (Appendix A.3).

Threat model. We defend against structure poisoning. An adversary flips up to B edges of the clean graph once, before training, producing $\mathbf{A}'$. The defender observes only $\mathbf{A}'$ and must train a model that is accurate on the clean test nodes. Since the attack is already realised in $\mathbf{A}'$, we take the training loss on the observed graph as the adversarial risk,

$$\mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}) \;:=\; \mathcal{L}(\boldsymbol{\theta}; \mathbf{A}', \mathbf{X}), \tag{2}$$

so that no inner maximization is required. Every derivative below is taken with respect to $\mathcal{L}_{\mathrm{adv}}$.

Doubly-sparse graph lottery tickets. A graph lottery ticket is a pair of binary masks, a weight mask $\mathbf{m}_{\boldsymbol{\theta}} \in \{0,1\}^d$ and an edge mask $\mathbf{m}_{\mathbf{A}} \in \{0,1\}^{|\mathcal{E}|}$, that define the sub-model $f_{\mathbf{m}_{\boldsymbol{\theta}}\odot\boldsymbol{\theta}}(\mathbf{m}_{\mathbf{A}} \odot \mathbf{A}', \mathbf{X})$, where $\odot$ is the Hadamard product and an edge mask removes both directed entries of an edge together. The masks meet target sparsities p_θ for weights and p_e for edges, that is, they keep $k_\theta = \lceil (1 - p_\theta)\, d \rceil$ weights and remove $k_e = \lfloor p_e\, |\mathcal{E}| \rfloor$ edges. Our goal is to select both masks from a single measurement of $\mathcal{L}_{\mathrm{adv}}$.

4 CURVATURE-AWARE ADVERSARIAL WINNING TICKETS

CA-AWT extracts both masks from one estimate of the Hessian diagonal of $\mathcal{L}_{\mathrm{adv}}$ (Figure 2). Section 4.1 defines the scores that rank weights and edges and states what the edge score can see, Section 4.2 gives their estimator, and Section 4.3 gives the procedure.

4.1 CURVATURE SCORING FOR WEIGHTS AND EDGES

Let $\boldsymbol{\theta}_{T_w}$ be the weights after a short warm-up on $\mathbf{A}'$. The second-order expansion of $\mathcal{L}_{\mathrm{adv}}$ around $\boldsymbol{\theta}_{T_w}$ along a displacement $\Delta\boldsymbol{\theta}$ is,

$$\mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}_{T_w} + \Delta\boldsymbol{\theta}) \approx \mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}_{T_w}) + \nabla\mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}_{T_w})^\top \Delta\boldsymbol{\theta} + \tfrac{1}{2}\, \Delta\boldsymbol{\theta}^\top \mathbf{H}\, \Delta\boldsymbol{\theta}, \tag{3}$$

with $\mathbf{H} = \nabla^2_{\boldsymbol{\theta}}\mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}_{T_w})$. Since $\nabla\mathcal{L}_{\mathrm{adv}}(\boldsymbol{\theta}_{T_w}) \approx \mathbf{0}$ at convergence, the loss change from perturbing a coordinate is dominated by the quadratic term, so curvature measures importance.

[1] Not a complete graph. Setting $a_e = 0$ would remove edge e. We never set any entry, we only differentiate with respect to it.

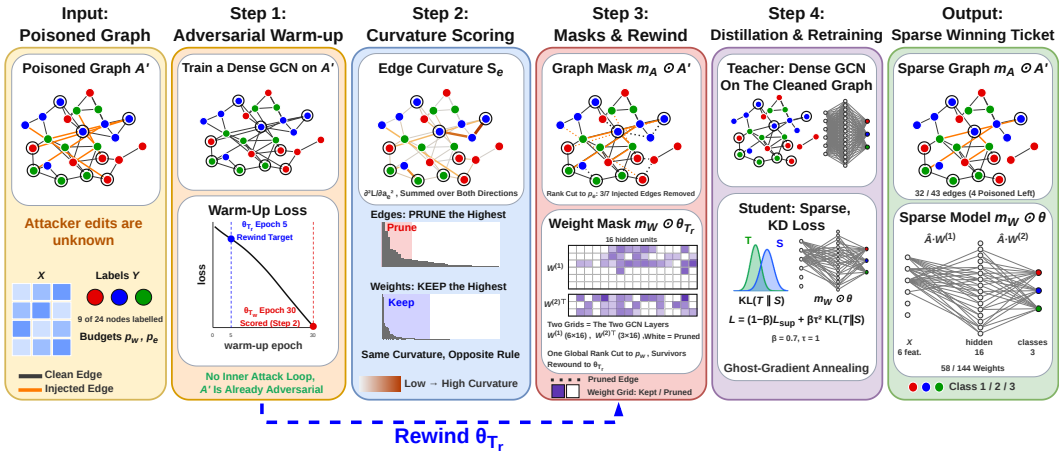

Figure 2: **Overview of CA-AWT on a toy poisoned graph.** Step 1 trains a dense GCN on the observed graph $\mathbf{A}'$ and keeps a rewind snapshot $\boldsymbol{\theta}_{T_r}$. Step 2 estimates the Hessian diagonal once and reads two rankings from it, edges are removed from the top and weights are kept from the top. Step 3 applies one global rank cut to each object and rewinds the surviving weights. Step 4 distils a dense teacher trained on the cleaned graph into the sparse student under an annealed mask. Injected edges are shown in orange for the reader only.

Weight saliency. Removing weight θ_j is the displacement $\Delta\theta_j = -\theta_j$ with all other coordinates fixed, so its predicted cost is the Optimal Brain Damage saliency (LeCun et al., 1990),

$$s_j^{\boldsymbol{\theta}} = \tfrac{1}{2} H_{jj}\, \theta_j^2, \qquad j = 1, \ldots, d. \tag{4}$$

A large $s_j^{\boldsymbol{\theta}}$ marks a weight whose removal would raise the loss, so CA-AWT keeps the weights with the largest saliency.

Edge score. The same expansion in $\mathbf{a}$ at $\mathbf{a} = \mathbf{1}$ gives, for each directed entry $e \in \vec{\mathcal{E}}$, the curvature of the loss along that entry,

$$s_e^{\mathbf{a}} = \left| \frac{\partial^2 \mathcal{L}_{\text{adv}}}{\partial a_e^2} \right|_{\mathbf{a}=\mathbf{1}}. \tag{5}$$

For edges, the rule is reversed. The attacker picks the flips that raise the training loss most, so the loss is fragile along an injected edge, and fragility is a large second derivative. The first derivative does not expose such an edge, because after the warm-up the injected edges help to minimize the loss, which is why a mask learned on that loss keeps them. Therefore, CA-AWT removes the edges with the largest curvature. Since an edge mask removes both directed entries of an edge, the score of the undirected edge $\{u, v\}$ is the predicted cost of the joint move $(\Delta a_{u\to v}, \Delta a_{v\to u}) = (-1, -1)$, which under the diagonal approximation of Eq. (3) is the sum of the two directed curvatures (Proposition 2),

$$s_{uv}^{\mathbf{A}} = s_{u\to v}^{\mathbf{a}} + s_{v\to u}^{\mathbf{a}}, \qquad \{u, v\} \in \mathcal{E}. \tag{6}$$

What the edge score can see. A message along the entry $u \to v$ is aggregated at v, so in an L-layer network $\mathcal{L}_{\text{adv}}$ depends on $a_{u\to v}$ only if v lies within $L-1$ hops of a labelled node; otherwise its first and second derivatives with respect to $a_{u\to v}$ are exactly zero (Proposition 1). Let $\mathcal{S} \subseteq \vec{\mathcal{E}}$ be the set of entries that satisfy this condition and define the coverage of the edge score as the fraction of undirected edges with at least one entry in $\mathcal{S}$,

$$c(\mathbf{A}') = \frac{1}{|\mathcal{E}|} \left| \{\{u, v\} \in \mathcal{E} : \; u \to v \in \mathcal{S} \text{ or } v \to u \in \mathcal{S}\} \right|. \tag{7}$$

Edges outside the coverage score zero, sit at the bottom of the ranking, so every removal is informed as long as $p_e \le c(\mathbf{A}')$. The same set $\mathcal{S}$ is the support of the gradient of Eq. (1), so any mask trained on the supervised loss is blind to exactly the same entries.

4.2 Scalable Estimation of the Hessian Diagonal

The exact diagonal of $\mathbf{H}$ costs $O(d^2)$. We use Hutchinson's unbiased estimator (Hutchinson, 1990) with Rademacher probes $\mathbf{v} \in \{-1, +1\}^d$, for which $\mathbb{E}[\mathbf{v}\mathbf{v}^\top] = \mathbf{I}$,

$$\operatorname{diag}(\mathbf{H}) = \mathbb{E}_{\mathbf{v}}\big[\mathbf{v} \odot (\mathbf{H}\mathbf{v})\big] \approx \frac{1}{P} \sum_{p=1}^{P} \mathbf{v}^{(p)} \odot \big(\mathbf{H}\mathbf{v}^{(p)}\big), \tag{8}$$

where each Hessian-vector product is one additional backward pass by Pearlmutter's identity (Pearlmutter, 1994),

$$\mathbf{Hv} = \nabla_{\boldsymbol{\theta}}\big(\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathrm{adv}} \cdot \mathbf{v}\big). \tag{9}$$

The estimator costs $2P$ backward passes, $O(Pd)$ time and $O(d)$ memory, the same order as first-order training. The identical procedure with $\mathbf{a}$ in place of $\boldsymbol{\theta}$ gives the edge diagonal of Eq. (5), since every message-passing layer multiplies its messages by the edge weights and $\mathcal{L}_{\mathrm{adv}}$ is therefore differentiable in $\mathbf{a}$. Under a weakly-coupled (quasi-diagonal) Hessian, the Gershgorin circle theorem gives $\lambda_{\max}(\mathbf{H}) \approx \max_j |H_{jj}|$, so the diagonal scores rank parameters consistently with the dominant curvature directions; The diagonal is used as a per-coordinate ranking signal, not as an estimate of the spectrum.

4.3 Ticket extraction and retraining

CA-AWT scores once and then never differentiates the loss twice again. The dense network is first trained on $(\mathbf{A}', \mathbf{X})$ by gradient descent on $\mathcal{L}_{\mathrm{adv}}$ for T_w epochs, reaching $\boldsymbol{\theta}_{T_w}$, with a snapshot $\boldsymbol{\theta}_{T_r}$ saved at an early epoch $T_r < T_w$. At epoch T_w, with weights $\boldsymbol{\theta}_{T_w}$, the weight and edge diagonals are estimated with P probes by Eqs. (8)–(9), which yields the saliencies of Eq. (4) and the edge scores of Eq. (6). Everything that follows operates on these two fixed rankings.

Mask construction. Both masks are built by rank. Writing $\mathrm{top}_k(\mathbf{s})$ for the index set of the k largest entries of a score vector $\mathbf{s}$, with ties broken by index,

$$\mathbf{m}_{\boldsymbol{\theta}} = \mathbf{1}\big[\, j \in \mathrm{top}_{k_\theta}(\mathbf{s}^{\boldsymbol{\theta}})\,\big], \qquad \mathbf{m}_{\mathbf{A}} = \mathbf{1}\big[\, \{u,v\} \notin \mathrm{top}_{k_e}(\mathbf{s}^{\mathbf{A}})\,\big]. \tag{10}$$

The weight mask keeps the k_θ most salient weights of the whole network under a single cut pooled across all layers, rather than a fixed fraction of each layer. The budget therefore flows to the layers where the criterion finds capacity redundant and, at extreme sparsity, the small output layer is not starved. The edge mask removes the k_e highest-curvature undirected edges, both directed entries together, so the pruned graph remains undirected. Selection by rank rather than by a score threshold guarantees that exactly k_θ weights and k_e edges are selected even when scores tie, which they do on every edge outside the coverage. The surviving weights are rewound to the snapshot (Renda et al., 2020),

$$\boldsymbol{\theta} \leftarrow \mathbf{m}_{\boldsymbol{\theta}} \odot \boldsymbol{\theta}_{T_r}. \tag{11}$$

Annealed masks and budget schedule. The ticket is retrained from epoch T_w to T under masks that anneal from dense pass-through to the hard cut with decay rate $\alpha > 0$,

$$\mathbf{m}(t) = \mathbf{m} + (\mathbf{1} - \mathbf{m})\, e^{-\alpha(t - T_w)}, \qquad t \geq T_w. \tag{12}$$

applied to both $\mathbf{m}_{\boldsymbol{\theta}}$ and $\mathbf{m}_{\mathbf{A}}$, so that pruned connections carry decaying gradients early in retraining and gradient flow is preserved before the topology hardens. At high weight sparsity, $p_\theta \geq p^\star$, the weight budget is in addition reached gradually along the fixed ranking,

$$p_\theta(t) = p_\theta\Big(1 - \big(1 - \tfrac{t - T_w}{T_{\mathrm{ramp}}}\big)^3\Big), \qquad T_w \leq t \leq T_w + T_{\mathrm{ramp}}. \tag{13}$$

with $\mathbf{m}_{\boldsymbol{\theta}}$ at epoch t given by Eq. (10) at sparsity $p_\theta(t)$ and T_{ramp} is the ramp length in epochs, set to 60% of the retraining span (Appendix A.7). Because the ranking is fixed the pruned sets are nested, and because no score is recomputed the method remains one-shot.

Distillation from the cleaned graph. A separate dense network is trained on the cleaned graph $\mathbf{m}_{\mathbf{A}} \odot \mathbf{A}'$ with the loss of Eq. (1) and provides logits $\mathbf{z}_i^t$ for every node i. Writing $\mathbf{z}_i^s$ for the student logits and σ for the softmax, the ticket minimizes,

$$\mathcal{L}_{\mathrm{KD}}(\boldsymbol{\theta}) = (1-\beta)\,\mathcal{L}\big(\mathbf{m}_{\boldsymbol{\theta}}(t) \odot \boldsymbol{\theta};\, \mathbf{m}_{\mathbf{A}}(t) \odot \mathbf{A}', \mathbf{X}\big) + \beta\,\tau^2\,\frac{1}{N}\sum_{i=1}^{N} \mathrm{KL}\big(\sigma(\mathbf{z}_i^t/\tau)\,\big\|\,\sigma(\mathbf{z}_i^s/\tau)\big), \tag{14}$$

with weight $\beta \in [0,1]$ and temperature $\tau > 0$. The teacher is trained on the cleaned graph rather than on the raw poisoned graph, because a teacher trained on poisoned structure transfers the poisoning to the student, and never on the clean graph, which the defender does not observe.

The teacher needs a cleaned graph, and how much to clean is not known in advance, because the defender does not observe the perturbation rate. The cleaning budget r_t of the teacher is selected by

a greedy search along the fixed ranking. A dense teacher is trained on the graph cut at each sparsity budget of the lattice and the one with the best validation accuracy is kept. Validation accuracy peaks where the cut stops removing injected edges and starts removing clean ones, so the selected budget tracks the poisoning level without knowledge of the perturbation rate (Section 6.4). One teacher then serves every ticket of that graph, at no more cost than a teacher retrained at every budget. The checkpoint with the best validation accuracy among the epochs at which the schedule has reached p_θ is restored, and the hard masks are applied. The complete procedure is summarized in Algorithm 1 in Appendix A.2 and illustrated in Figure 2.

4.4 Theoretical properties

Three results are specific to the method and three hold for any binary mask; all proofs are in Appendix A.4.

Proposition 1 (Support of a label-driven criterion)**.** *In an L-layer message-passing network, if the target v of the entry $u \to v$ is more than $L-1$ hops from every labelled node, the first and second derivatives of the loss of Eq.* (1) *with respect to $a_{u\to v}$ are exactly zero at $\mathbf{a} = \mathbf{1}$. The gradient, the curvature and any mask trained on that loss therefore share the support $\mathcal{S}$, and the largest edge sparsity such a criterion can rank is the coverage $c(\mathbf{A}')$ of Eq.* (7)*.*

Proposition 2 (The symmetric sum is the second-order edge saliency)**.** *Under the diagonal expansion of Eq.* (3)*, the predicted change in loss from removing the undirected edge $\{u, v\}$ equals $\frac{1}{2}\big(H_{u\to v,\,u\to v} + H_{v\to u,\,v\to u}\big)$, so Eq.* (6) *ranks edges by the cost of the operation actually performed.*

Theorem 1 (Ghost-grad stability)**.** *For an L-Lipschitz, ρ-smooth per-sample loss and the annealed mask of Eq.* (12) *with $\alpha > \eta\rho$, retraining for T steps is uniformly stable with parameter $\beta_{\mathrm{ghost}} \le 2\eta L^2/\big(n(\alpha - \eta\rho)\big)$, independent of T, whereas a hard cut at $t = T_w$ admits only $2\eta L^2 T\, e^{\eta\rho T}/n$.*

The ticket also inherits Lipschitz, Frobenius-complexity and PAC-Bayes generalization guarantees that hold for any binary weight mask and therefore under the global allocation of Eq. (10) (Propositions 3 to 5, Appendix A.4.4). They guarantee that the ticket is no less well behaved than the dense model, but they do not depend on which weights the mask keeps, so they do not by themselves separate curvature from any other criterion; that separation is the empirical subject of Section 6.

5 Experimental Setting

Datasets and attacks We evaluate on four citation graphs of increasing scale, Cora, Citeseer, PubMed and OGBN-ArXiv, and on one heterophilous graph, Chameleon, with the splits of ARGS and UGS. Cora and Citeseer are poisoned by three untargeted structure attacks, PGD (Xu et al., 2019), MetaAttack (Zügner & Günnemann, 2019) and Meta-PGD (Mujkanovic et al., 2022), at 5, 10, 15 and 20% of the edge count; PubMed and Chameleon by PGD, and OGBN-ArXiv by PR-BCD (Geisler et al., 2021) at 10 and 15%. The attacker's edges are recovered by differencing the clean and poisoned graphs, so we can measure how many of them each method removes. Splits, attack generation and dataset statistics are in Appendix A.7.

Baselines UGS (Chen et al., 2021), the standard graph lottery ticket, and ARGS (Chowdhury et al., 2023), the only robust one and the state of the art, are the published methods for tickets under poisoning; both are reported at every sparsity budget of the lattice. STRG (Li et al., 2023), the dense self-training defence ARGS uses as its reference, marks the accuracy of a defended model at no sparsity. A learned-magnitude arm, a UGS-style edge mask inside our own pipeline with every other component unchanged, isolates the edge criterion. Details are in Appendix A.7.

Backbone and protocol The backbones are a two-layer GCN with 512 hidden units, a two-layer GIN with sum aggregation and 512 hidden units, and on OGBN-ArXiv a 28-layer residual GCN, the same models the baselines use. Sparsity follows ARGS's lattice of twenty sparsity budgets, budget r removing $1 - 0.95^r$ of the edges and $1 - 0.8^r$ of the weights, $r = 1, \ldots, 20$. Every cell is the mean of three seeds. The hyper-parameters and the teacher budget are selected on validation only (Appendix A.7); their sensitivity is studied in Section 6.4.

Table 1: Accuracy at three sparsity budgets ARGS reports (graph / model sparsity), mean ± sd of three seeds, bold where CA-AWT exceeds ARGS. The remaining panels are in Appendix A.9.

	$r = 5$ (22.6% / 67.2%)			$r = 13$ (48.7% / 94.5%)			$r = 19$ (62.3% / 98.6%)		
Dataset / attack / rate	UGS	ARGS	CA-AWT	UGS	ARGS	CA-AWT	UGS	ARGS	CA-AWT
Cora / PGD / 5%	76.3	82.0	**84.1**±0.7	70.5	78.8	**81.5**±0.3	68.7	74.5	**77.1**±0.3
Cora / MetaAttack / 5%	77.2	81.7	**83.0**±0.4	71.8	76.7	**79.6**±0.5	68.8	71.8	**76.1**±0.9
Cora / Meta-PGD / 5%	71.6	79.5	**80.8**±1.7	70.4	76.3	76.0±1.3	69.4	72.0	**72.0**±0.7
Cora / PGD / 20%	59.0	78.5	**79.0**±2.4	50.5	77.9	77.5±0.6	47.9	75.1	74.1±0.8
Cora / MetaAttack / 20%	63.0	79.4	78.8±1.7	64.4	75.5	**76.7**±2.0	62.7	70.5	**74.5**±0.9
Cora / Meta-PGD / 20%	61.6	78.2	**80.2**±0.4	58.7	75.5	74.9±0.5	58.0	73.1	**73.2**±0.6
PubMed / PGD / 20%	53.9	80.1	**81.2**±0.4	54.5	83.2	82.7±0.5	52.7	80.4	80.3±0.9
OGBN-ArXiv / PR-BCD / 10%	60.5	66.7	65.5±0.6	57.6	61.8	**62.1**±0.6	52.6	56.7	**57.3**±0.4

6 Results and Analysis

6.1 Main results

Cora and Citeseer. Figure 3 shows the full lattice on Cora and Table 1 the three sparsity budgets ARGS reports; the other panels are in Appendix A.9.1. CA-AWT beats UGS in 98% of all cells, by 13 points on average, and is level with iterative ARGS, ahead in 44% of the cells and by 0.4 points on average, with one scoring pass and one teacher per graph. On Cora it is ahead of ARGS in 60% of the cells and by one point on average; on Citeseer the two are within 0.2 points. Under PGD and MetaAttack the gain over ARGS grows with sparsity, where the teacher keeps the student on track after the budget schedule switches on; under Meta-PGD, whose insertions are spread more widely, the gain sits at the small budgets and the two methods are level at high sparsity.

Generalization across datasets and backbones. On PubMed CA-AWT is above UGS everywhere and within 0.1 points of ARGS. On OGBN-ArXiv, with the 28-layer backbone, it is within 0.6 points of ARGS and level with it at high sparsity under 10% perturbation. With a GIN backbone (Figure 4) CA-AWT is level with ARGS, ahead in half of the cells and by 0.3 points on average, and above UGS in 99% of the cells.

6.2 Analysis of the edge criterion

Figure 5 compares the curvature criterion with a learned magnitude mask, everything else being equal. Curvature removes 88% of the attacker's edges on average and above 95% from the middle of the lattice on and over 98% at high sparsity, while the learned mask removes 30%, below random pruning at every budget. This is the mechanism of Figure 1 measured at scale.

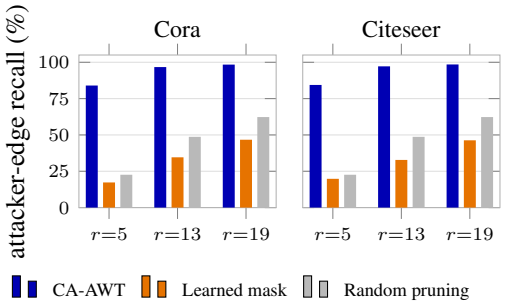

Figure 5: Attacker-edge recall at three sparsity budgets, mean over attacks, rates and seeds.

Random pruning is blind to the poison, so its recall equals the edge budget. The learned mask stays near 30% at every budget, so it removes fewer injected edges than a random cut. It is trained on the poisoned loss, the injected edges help to fit the poisoned labels, so the mask gives them large values and a prune-the-smallest rule keeps them. The gap in accuracy follows the gap in recall, 13 points on Cora and 10 on Citeseer over the lattice, and it grows with the attack, from 0.5 to 8 points at 5% perturbation to 6 to 29 points at 20% (Appendix A.9.2). Against ARGS the detector is the difference. On ARGS's own Cora graph at 20% perturbation after all of its rounds, ARGS removes 68, 47 and 14% of the injected edges between train and train, train and test, and test and test nodes, 51% in all; at the same budget on the same graph CA-AWT removes all of them in every group (Appendix A.9.3).

6.3 Ablation study

Table 2 removes one component at a time. The two largest effects are the edge criterion and the teacher's training graph. Training the teacher on the raw poisoned graph costs 8.6 points on Cora and 10.4 on Citeseer, at every budget, so the cleaned graph is not only the ticket's input but what makes distillation help at all; a stronger teacher trained on dirty structure transfers the poisoning.

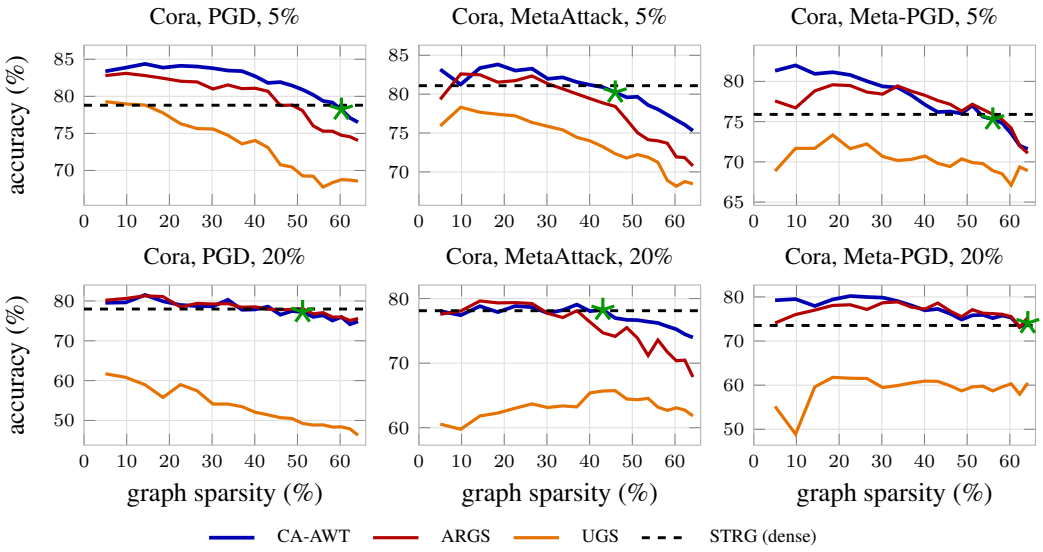

Figure 3: Accuracy against graph sparsity on Cora (GCN), mean of three seeds. Dashed line, STRG at no sparsity. Green star, the sparsest CA-AWT ticket within one point of STRG.

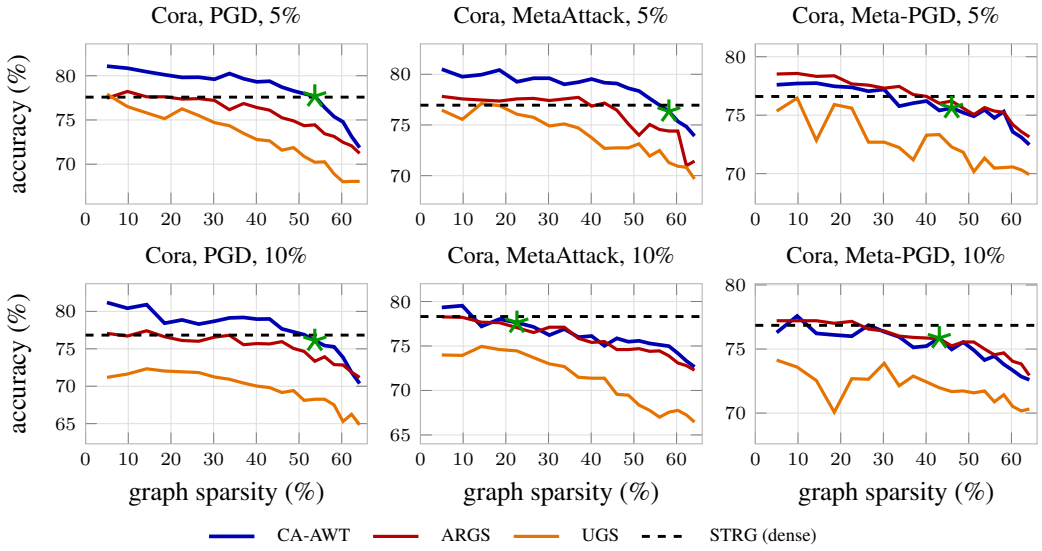

Figure 4: Accuracy against graph sparsity on Cora (GIN), same conventions as Figure 3; Citeseer is in Appendix A.9.1.

The teacher's gain rises with attack strength, from $+4.4$ at 5% to $+7.8$ at 20% on Cora under PGD, and is U-shaped across budgets, smallest where the budget schedule takes over part of its work (Appendix A.9.6). Global allocation is worth 3.3 points on Cora and 3.0 on Citeseer, and 7.8 points at $r = 19$ alone, because a per-layer cut at 98.6% leaves the 3,584-weight classifier about fifty weights. Summing the two directed curvatures is worth 1.8 points on Cora and 1.1 on Citeseer and, more importantly, raises coverage from 54 to 70% and keeps every surviving edge bidirectional.

6.4 Hyper-parameter study

CA-AWT is insensitive to its hyperparameters within the ranges we swept (Appendix A.8). Ten of the twelve (β, τ) pairs are tied on validation; the selected pair is within 0.1 points of the best pair on test, and the rewind epoch, warm-up length, and probe count each move the mean by less than a point. The one sharp setting is the schedule threshold, and its value is measured rather than tuned; ramping helps above 92% model sparsity and hurts below.

Table 2: Component ablations on Cora and Citeseer, mean over sparsity budgets $r = 5$, 13 and 19, three attacks, 5 and 20% perturbation and three seeds. Δ is the change in accuracy relative to the full recipe. Per-budget numbers are in Appendix A.9.5.

	Cora		Citeseer	
Variant	acc (%)	Δ	acc (%)	Δ
CA-AWT (full recipe)	77.7	–	73.0	–
w/o teacher (no distillation)	72.8	−4.9	69.6	−3.4
w/o curvature edge score (learned magnitude mask)	64.0	−13.8	62.8	−10.3
w/o cleaned graph for the teacher (poisoned instead)	69.2	−8.6	62.6	−10.4
w/o global weight budget (per-layer budget)	74.4	−3.3	70.1	−3.0
w/o summed edge score (per directed entry)	75.9	−1.8	71.9	−1.1
w/o budget schedule above 92% model sparsity	75.9	−1.8	71.5	−1.5
w/o OBD saliency ($\lvert\theta H\rvert$ instead)	77.1	−0.6	72.1	−0.9

The teacher budget correlates with the poisoning level. The one setting CA-AWT chooses from the data is the teacher's cleaning budget, r_t, and it never needs the perturbation rate. Figure 6 shows the budget at which validation selects the teacher for each citation graph, attack, and rate. It rises with the rate in every column, from a 5% cut at 5% perturbation to a 37% cut at 20% on Cora under PGD, and the rank correlation between rate and selected budget is 0.65 over all cells. A defender who does not know the rate therefore still recovers a cut matched to it. The search costs one dense training per candidate budget, no more than a teacher retrained at every budget, and stopping it after five budgets without improvement keeps the same choice in 88% of cases at 63% of the cost (Appendix A.9.8).

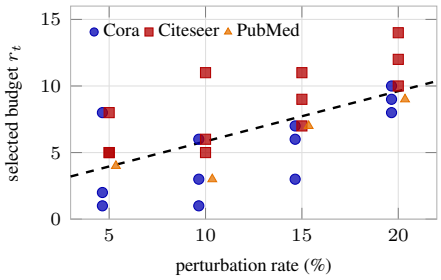

Figure 6: Teacher budget r_t against perturbation rate, one point per graph, attack and rate; dashed, least-squares fit; Spearman 0.65.

6.5 Beyond homophily

ARGS's edge-cleaning term assumes that neighbours have similar features, and its authors name non-homophilous graphs as an open limitation. CA-AWT uses no such term, so we test it on Chameleon, whose edge homophily is 0.23 against about 0.81 for Cora, poisoned with PGD under the same protocol (Appendix A.9.7). The detector transfers unchanged. Curvature removes 87 to 98%, while the learned mask is below random in every cell and varies by up to 13 points. Accuracy does not transfer as a gain. CA-AWT is within three points of the learned mask in every cell, ahead under 20% perturbation and behind in the two sparsest cells at 5%, because PGD is deletion-dominated on this dense graph and removing the most influential edges costs a GCN more than removing the injected ones recovers. The mechanism holds without homophily; the accuracy benefit depends on what the attack did to the graph.

7 Conclusion and Limitations

We introduced CA-AWT, a one-shot method that extracts a doubly-sparse, poisoning-robust graph lottery ticket from a single second-order reading of the loss. Because training on a poisoned graph coincides with adversarial training, one curvature measurement serves both objects. It ranks weights for removal by their Optimal-Brain-Damage saliency, and it localises the injected adversarial edges by their curvature, which is exactly where a learned mask protects them, on homophilous and heterophilous graphs alike and without any prior on what an attack looks like. A teacher trained on the curvature-cleaned graph, a global cross-layer budget, and a scheduled cut on the fixed ranking make extreme sparsity attainable while leaving the Lipschitz, Frobenius-complexity and PAC-Bayes guarantees intact. With these pieces, one-shot tickets outperform UGS in 98% of the cells and track an iterative method on citation graphs at a fraction of the scoring cost. Two directions follow naturally. Scoring the full objective, whose distillation term covers every node, would close the coverage gap without new machinery, and an attack-sized edge budget would address the two regimes where the method loses.

AI USE STATEMENT

The authors used GenAI tools for support, including improving the language, grammar, and readability of the manuscript and assisting with refactoring and organizing source code for final submission. We did not use GenAI tools to generate synthetic datasets, to formulate the mathematical claims or to write the proofs, and the remaining tasks in the disclosure list are not applicable to this work. The authors carefully reviewed, revised, and validated all AI-generated or AI-assisted content. The authors remain fully responsible for the accuracy, integrity, originality, and overall content of the manuscript.

ETHICS STATEMENT

This study did not involve human participants, animal subjects, or the collection of personal or identifiable data. The work studies a defence against poisoning of graph neural networks. It uses only public benchmark graphs and publicly released attack implementations, and it introduces no new attack. The authors conducted all data processing and computational analyses in compliance with relevant institutional regulations and ethical standards.

REPRODUCIBILITY STATEMENT

The code is available at `https://anonymous.4open.science/r/ca-awt-8CF2`. It contains the method, the sweeps that produce every table and figure, and the scripts that generate the poisoned graphs. Section 5 gives the datasets, attacks, backbones and the sparsity lattice. Appendix A.7 gives every hyper-parameter, the splits and the attack generation. Appendix A.4 contains the full proofs of Propositions 1 to 5 and Theorem 1, with the assumptions stated in the main text. Appendix A.8 reports the sensitivity of every hyper-parameter we swept. Every reported cell is the mean of three seeds.

REFERENCES

Peter Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In Advances in Neural Information Processing Systems, 2017.

Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. A unified lottery ticket hypothesis for graph neural networks. In International Conference on Machine Learning, 2021.

Subhajit Dutta Chowdhury, Zhiyu Ni, Qingyuan Peng, Souvik Kundu, and Pierluigi Nuzzo. Sparse but strong: Crafting adversarially robust graph lottery tickets. In NeurIPS Workshop: New Frontiers in Graph Learning (GLFrontiers), 2023.

Subhajit Dutta Chowdhury, Zhiyu Ni, Qingyuan Peng, Souvik Kundu, and Pierluigi Nuzzo. Unveiling adversarially robust graph lottery tickets. Transactions on Machine Learning Research, 2024.

Xin Dong, Shangyu Chen, and Sinno Jialin Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In Advances in Neural Information Processing Systems, 2017.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Uncertainty in Artificial Intelligence, 2017.

Negin Entezari, Saba A. Al-Sayouri, Amirali Darvishzadeh, and Evangelos E. Papalexakis. All you need is low (rank): Defending against adversarial attacks on graphs. In International Conference on Web Search and Data Mining (WSDM), 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In International Conference on Learning Representations, 2021.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In International Conference on Learning Representations, 2019.

Yonggan Fu, Qixuan Yu, Yang Zhang, Shang Wu, Xu Ouyang, David Cox, and Yingyan Lin. Drawing robust scratch tickets: Subnetworks with inborn robustness are found within randomly initialized networks. In Advances in Neural Information Processing Systems, 2021.

Simon Geisler, Tobias Schmidt, Hakan Şirin, Daniel Zügner, Aleksandar Bojchevski, and Stephan Günnemann. Robustness of graph neural networks at scale. In Advances in Neural Information Processing Systems, 2021.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Advances in Neural Information Processing Systems, 2017.

Babak Hassibi and David G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In Advances in Neural Information Processing Systems, 1993.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

Bo Hui, Da Yan, Xiaolong Ma, and Wei-Shinn Ku. Rethinking graph lottery tickets: Graph sparsity matters. In International Conference on Learning Representations, 2023.

Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. Communications in Statistics—Simulation and Computation, 1990.

Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In International Conference on Learning Representations, 2017.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations, 2017.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In Advances in Neural Information Processing Systems, 1990.

Kuan Li, Yang Liu, Xiang Ao, and Qing He. Revisiting graph adversarial attack and defense from a data distribution perspective. In International Conference on Learning Representations, 2023.

Yaxin Li, Wei Jin, Han Xu, and Jiliang Tang. Deeprobust: A pytorch library for adversarial attacks and defenses. arXiv preprint arXiv:2005.06149, 2020.

Felix Mujkanovic, Simon Geisler, Stephan Günnemann, and Aleksandar Bojchevski. Are defenses for graph neural networks robust? In Advances in Neural Information Processing Systems, 2022.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. In International Conference on Learning Representations, 2018.

Barak A. Pearlmutter. Fast exact multiplication by the Hessian. Neural Computation, 1994.

Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. In International Conference on Learning Representations (ICLR), 2020.

Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. Hydra: Pruning adversarially robust neural networks. In Advances in Neural Information Processing Systems, 2020.

Sidak Pal Singh and Dan Alistarh. Woodfisher: Efficient second-order approximation for neural network compression. In Advances in Neural Information Processing Systems, 2020.

Xianfeng Tang, Yandong Li, Yiwei Sun, Huaxiu Yao, Prasenjit Mitra, and Suhang Wang. Transferring robustness for graph neural network against poisoning attacks. In ACM International Conference on Web Search and Data Mining, 2020.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In International Conference on Learning Representations, 2022.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In International Conference on Learning Representations, 2018.

Kun Wang, Yuxuan Liang, Pengkun Wang, Xu Wang, Pengfei Gu, Junfeng Fang, and Yang Wang. Searching lottery tickets in graph neural networks: A dual perspective. In International Conference on Learning Representations, 2023.

Huijun Wu, Chen Wang, Yuriy Tyshetskiy, Andrew Docherty, Kai Lu, and Liming Zhu. Adversarial examples for graph data: Deep insights into attack and defense. In International Joint Conference on Artificial Intelligence, 2019.

Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: An optimization perspective. In International Joint Conference on Artificial Intelligence, 2019.

Yiding Yang, Jiayan Qiu, Mingli Song, Dacheng Tao, and Xinchao Wang. Distilling knowledge from graph convolutional networks. In IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020.

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W. Mahoney. Pyhessian: Neural networks through the lens of the hessian. In IEEE International Conference on Big Data, 2020.

Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. In International Conference on Learning Representations, 2022.

Xiang Zhang and Marinka Zitnik. GNNGuard: Defending graph neural networks against adversarial attacks. In Advances in Neural Information Processing Systems, 2020.

Daniel Zügner and Stephan Günnemann. Adversarial attacks on graph neural networks via meta learning. In International Conference on Learning Representations, 2019.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2018.

A Appendix

A.1 Extended related work

Graph lottery tickets. The Lottery Ticket Hypothesis (Frankle & Carbin, 2019) spurred a broad effort to find sparse subnetworks that train to full accuracy. For graphs, Unified Graph Sparsification (UGS) (Chen et al., 2021) introduced the graph lottery ticket by jointly pruning the adjacency matrix and the GNN weights through differentiable masks, cutting the cost of message passing as well as of the weight multiplications. Subsequent work has scaled and refined graph sparsification along several axes, including inference efficiency, generalization, and training stability (Hui et al., 2023; Wang et al., 2023). These methods are designed for the clean setting and, as Chowdhury et al. (2023) show, they inherit and can even amplify the vulnerability of GNNs to structural attacks.

Adversarial attacks on graphs. Nettack (Zügner et al., 2018) introduced targeted attacks on node classification, and Metattack (Zügner & Günnemann, 2019) cast global poisoning as meta-learning over the graph structure, showing that a small number of edge flips made before training degrades accuracy far more than feature perturbations. PGD topology attacks (Xu et al., 2019), Meta-PGD (Mujkanovic et al., 2022) and the scalable PR-BCD (Geisler et al., 2021) optimise the discrete structure with first-order methods. These attacks are typically generated with DeepRobust (Li et al., 2020), and the perturbations they produce concentrate around the labelled nodes and connect nodes with dissimilar features (Wu et al., 2019; Li et al., 2023).

Robust GNNs against structure poisoning. Most defences against structural poisoning reshape the graph before or during training. Jaccard and SVD preprocessing remove edges between dissimilar nodes or project the adjacency onto a low-rank subspace (Wu et al., 2019; Entezari et al., 2020), ProGNN learns a cleaned graph under low-rank and feature-smoothness priors (Jin et al., 2020), GNNGuard reweights edges by feature similarity during aggregation (Zhang & Zitnik, 2020), and STRG trains with pseudo-labels on the less perturbed test nodes (Li et al., 2023). These methods restore accuracy on dense models with dense or nearly dense graphs, and most of them encode a homophily or low-rank prior on what a clean graph looks like. None yields a sparse ticket. ARGS (Chowdhury et al., 2023; 2024) is the only method that pursues robustness and double sparsity together. It adds a feature-smoothness loss and a cross-entropy term on MLP pseudo-labelled test nodes to the mask objective and re-learns both masks in every prune-and-retrain round; its journal version extends the evaluation to PR-BCD, GR-BCD and an adaptive attacker. Its authors name graphs where homophily does not hold as the open limitation.

Second-order pruning and knowledge distillation. Pruning by the Hessian dates to Optimal Brain Damage and Optimal Brain Surgeon (LeCun et al., 1990; Hassibi & Stork, 1993), which remove the weights whose deletion changes the loss least under a second-order expansion. The idea was revived for deep networks by layer-wise OBS (Dong et al., 2017) and by block-diagonal Fisher approximations (Singh & Alistarh, 2020), and Hutchinson's estimator (Hutchinson, 1990) together with Pearlmutter's Hessian-vector product (Pearlmutter, 1994) make the diagonal affordable at scale, as in PyHessian (Yao et al., 2020). Curvature has also been tied to generalization and robustness. Sharp minima generalize worse (Keskar et al., 2017), sharpness-aware minimisation seeks flat ones directly (Foret et al., 2021), HYDRA learns pruning scores under an adversarial objective for robust sparse image classifiers (Sehwag et al., 2020), and robust subnetworks have been shown to exist even in randomly initialised networks (Fu et al., 2021). On graphs, curvature has so far meant discrete Ricci curvature of the structure, used to rewire bottleneck edges (Topping et al., 2022), rather than the curvature of the training loss along an edge. Knowledge distillation (Hinton et al., 2015) has been used to compress GNNs into smaller GNNs or MLPs (Yang et al., 2020; Zhang et al., 2022) and to transfer robustness from a clean graph to a poisoned one (Tang et al., 2020).

Generalization guarantees for pruned and graph networks. Norm-based and PAC-Bayes analyses bound the generalization of neural networks through the spectral and Frobenius norms of their weights (Bartlett et al., 2017; Neyshabur et al., 2018) and have been made non-vacuous for trained networks (Dziugaite & Roy, 2017). Pruning reduces exactly the quantities these bounds depend on, which is why a sparse ticket can generalize at least as well as its dense parent.

A.2 Algorithm

Algorithm 1 lists the steps of CA-AWT described in Section 4.3.

Algorithm 1 Curvature-Aware Adversarial Winning Tickets

Require: poisoned graph $(\mathbf{A}', \mathbf{X})$, labels on $\mathcal{V}_{\text{tr}}$, sparsities p_θ, p_e, warm-up T_w, rewind epoch T_r, probes P, decay α, schedule $(p^\star, T_{\text{ramp}})$, distillation (β, τ), total epochs T

1: train $\boldsymbol{\theta}$ on $\mathcal{L}_{\text{adv}}$ for T_w epochs; save $\boldsymbol{\theta}_{T_r}$ at epoch T_r ▷ warm-up
2: estimate $\text{diag}(\mathbf{H})$ over $\boldsymbol{\theta}$ and over $\mathbf{a}$ at $\boldsymbol{\theta}_{T_w}$ with P probes ▷ scoring, Eqs. (8)–(9)
3: $s_j^{\boldsymbol{\theta}} \leftarrow \frac{1}{2} H_{jj} \theta_j^2$; $s_{uv}^{\mathbf{A}} \leftarrow s_{u\to v}^{\mathbf{a}} + s_{v\to u}^{\mathbf{a}}$ Eqs. (4), (6)
4: $\mathbf{m}_{\boldsymbol{\theta}} \leftarrow$ keep $\text{top}_{k_\theta}(\mathbf{s}^{\boldsymbol{\theta}})$ pooled over all layers; $\mathbf{m}_{\mathbf{A}} \leftarrow$ remove $\text{top}_{k_e}(\mathbf{s}^{\mathbf{A}})$ ▷ masks, Eq. (10)
5: $\boldsymbol{\theta} \leftarrow \mathbf{m}_{\boldsymbol{\theta}} \odot \boldsymbol{\theta}_{T_r}$ Eq. (11)
6: train one dense teacher on the graph cut at the budget r_t that maximizes validation accuracy; obtain $\mathbf{z}^t$ ▷ teacher
7: **for** $t = T_w, \ldots, T-1$ **do**
8: anneal $\mathbf{m}_{\boldsymbol{\theta}}(t), \mathbf{m}_{\mathbf{A}}(t)$ by Eq. (12); if $p_\theta \geq p^\star$, set the weight budget by Eq. (13)
9: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\, \mathbf{m}_{\boldsymbol{\theta}}(t) \odot \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{KD}}(\boldsymbol{\theta})$ Eq. (14)
10: **end for**
11: restore the best-validation checkpoint at sparsity p_θ; apply the hard masks

Ensure: ticket $f_{\mathbf{m}_{\boldsymbol{\theta}} \odot \boldsymbol{\theta}}$ on $\mathbf{m}_{\mathbf{A}} \odot \mathbf{A}'$

A.3 The edge-weight vector $\mathbf{a}$

Since the matrix $\mathbf{A}'$ is binary, the loss has no derivative with respect to an edge. We therefore assign each present edge e a weight a_e that multiplies its message in each layer, and collect these weights in $\mathbf{a}$. Absent edges have no entry. With $a_e = 1$, the message passes unchanged; with $a_e = 0$, the edge is removed. At $\mathbf{a} = \mathbf{1}$ the model is the ordinary GNN on $\mathbf{A}'$.

The matrix $\mathbf{a}$ serves one purpose. After warm-up, we take the second derivative of the loss with respect to each a_e at $\mathbf{a} = \mathbf{1}$, which gives the edge score in Eq. (5). No entry of $\mathbf{a}$ is ever moved by gradient descent. The warm-up phase only updates $\boldsymbol{\theta}$. In the retraining, the mask $\mathbf{m}_{\mathbf{A}}(t)$ of Eq. (12) multiplies the edges, but its values follow a fixed formula on a fixed ranking, and again only $\boldsymbol{\theta}$ receives gradients. In contrast, UGS and ARGS use gradient descent to update their edge masks in every round.

A.4 Proofs

Throughout, $\mathcal{L}$ denotes the supervised loss of Eq. (1) evaluated on the observed graph, that is $\mathcal{L}_{\text{adv}}$, and all derivatives with respect to edge weights are taken at $\mathbf{a} = \mathbf{1}$.

A.4.1 Proposition 1 (support of a label-driven criterion)

Proof of Proposition 1. Consider a message-passing layer of the form $\mathbf{h}_v^{(\ell)} = \phi\big(\mathbf{h}_v^{(\ell-1)},\ \{a_{u\to v}\, \psi(\mathbf{h}_u^{(\ell-1)})\}_{u \in \mathcal{N}(v)}\big)$, which covers the GCN, GAT and GraphSAGE layers used in this paper. The weight $a_{u\to v}$ multiplies the message from u and enters the representation of v only. By induction over layers, a change to $a_{u\to v}$ can alter $\mathbf{h}_w^{(\ell)}$ only for nodes w within $\ell - 1$ hops of v. The loss of Eq. (1) reads the layer-L outputs of the labelled nodes only. Hence, if every labelled node is more than $L - 1$ hops from v, the loss is constant in $a_{u\to v}$ on a neighbourhood of $\mathbf{a} = \mathbf{1}$, and all of its derivatives with respect to $a_{u\to v}$, of every order, vanish. This proves the first claim.

For the second claim, let $\mathbf{m}$ be a continuous edge mask trained by gradient descent on $\mathcal{L}(\boldsymbol{\theta}; \mathbf{m} \odot \mathbf{a}, \mathbf{X}) + \lambda \|\mathbf{m}\|_1$, as in UGS and ARGS. The gradient of the first term with respect to $m_{u\to v}$ is $a_{u\to v}\, \partial\mathcal{L}/\partial a_{u\to v}$, which is zero on exactly the entries above. The gradient of the second term is $\lambda\, \text{sign}(m_{u\to v})$, the same constant for every entry. On the blind entries the mask therefore moves in lockstep and the ranking among them is determined by initialisation alone. Any criterion built from the gradient or the curvature of $\mathcal{L}$ thus has the support $\mathcal{S}$ of Section 4.1, and the largest edge sparsity that such a criterion can rank is the coverage $c(\mathbf{A}')$ of Eq. (7). □

Empirical check. On a path graph $0-1-2-3-\cdots$ with only node 0 labelled and $L \in \{1, 2, 3, 4\}$ layers, the entries with non-zero gradient land on targets $\{0\}$, $\{0, 1\}$, $\{0, 1, 2\}$ and $\{0, 1, 2, 3\}$ respectively, as the rule predicts. On Cora under PGD at 20%, the fraction of directed entries whose

target lies within one hop of a labelled node, computed from the graph alone, is 0.5412; the fraction with a non-zero curvature score, measured from the scoring code, is 0.5412.

A.4.2 Proposition 2 (the symmetric sum is the second-order edge saliency)

Proof of Proposition 2. Let $g_{u\to v} = \partial\mathcal{L}/\partial a_{u\to v}$ and let $H_{e,e'}$ denote the entries of the Hessian of $\mathcal{L}$ with respect to the pair $(a_{u\to v}, a_{v\to u})$, all at $\mathbf{a} = \mathbf{1}$. Removing the undirected edge $\{u, v\}$ is the joint displacement $\Delta = (-1, -1)$ on this pair, so the second-order expansion gives

$$\Delta\mathcal{L} \;\approx\; -\big(g_{u\to v} + g_{v\to u}\big) + \tfrac{1}{2}\Big(H_{u\to v,\,u\to v} + H_{v\to u,\,v\to u} + 2\,H_{u\to v,\,v\to u}\Big). \tag{15}$$

Two approximations, both already made on the weight side, reduce this to the sum. At a converged warm-up the first-order terms are small, which is the assumption under which Optimal Brain Damage discards the gradient term for weights. The cross term $H_{u\to v,\,v\to u}$ is an off-diagonal entry of the edge Hessian, which the Hutchinson estimator of Eq. (8) does not return, and dropping it is the diagonal approximation used in Eq. (4). What remains is $\frac{1}{2}(H_{u\to v,u\to v} + H_{v\to u,v\to u})$, which is proportional to $s^{\mathbf{a}}_{u\to v} + s^{\mathbf{a}}_{v\to u}$ since both diagonal entries are non-negative at a local minimum and the absolute values in Eq. (5) are then immaterial. □

Why not the maximum or the minimum. The maximum of the two directed curvatures would be the correct statistic if a single directed entry were removed, which never happens, and it discards the second measurement whenever both directions carry signal ($2{,}042$ of Cora's $5{,}370$ edges under PGD 20%). The minimum returns zero for every edge with exactly one informative direction ($1{,}728$ edges, 32% of Cora) and thereby reproduces the blind spot that the aggregation is meant to reduce. The mean ranks identically to the sum.

A.4.3 Theorem 1 (ghost-grad stability)

Proof of Theorem 1. Let S and S' be two training sets of size n that differ in one example, and let $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}'_t$ be the parameters after t steps of masked gradient descent on each, with the same initialisation and the same annealed mask $\mathbf{m}(t)$ of Eq. (12). The updates are

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\,\mathbf{m}(t) \odot \nabla\mathcal{L}(\boldsymbol{\theta}_t; S), \qquad \boldsymbol{\theta}'_{t+1} = \boldsymbol{\theta}'_t - \eta\,\mathbf{m}(t) \odot \nabla\mathcal{L}(\boldsymbol{\theta}'_t; S'). \tag{16}$$

Write $\Delta_t = \|\boldsymbol{\theta}_t - \boldsymbol{\theta}'_t\|_2$ and decompose the gradient difference into a parameter-mismatch term and a sample-mismatch term,

$$\nabla\mathcal{L}(\boldsymbol{\theta}_t; S) - \nabla\mathcal{L}(\boldsymbol{\theta}'_t; S') = \big[\nabla\mathcal{L}(\boldsymbol{\theta}_t; S) - \nabla\mathcal{L}(\boldsymbol{\theta}'_t; S)\big] + \big[\nabla\mathcal{L}(\boldsymbol{\theta}'_t; S) - \nabla\mathcal{L}(\boldsymbol{\theta}'_t; S')\big]. \tag{17}$$

By ρ-smoothness the first bracket has norm at most $\rho\Delta_t$, and since $\|\mathbf{m}(t)\|_\infty \le 1$ the corresponding update contributes at most $(1+\eta\rho)\Delta_t$ to Δ_{t+1}. The second bracket involves only the single differing example, so by the L-Lipschitz property of the per-sample loss its norm is at most $2L/n$. On the coordinates that the final mask prunes, which are the only coordinates on which the two runs can be driven apart by the mask itself, the annealed mask satisfies $\|\mathbf{m}(t)\|_\infty \le e^{-\alpha t}$ by assumption, so this term contributes at most $(2\eta L/n)\,e^{-\alpha t}$. Together,

$$\Delta_{t+1} \;\le\; (1+\eta\rho)\,\Delta_t + \frac{2\eta L}{n}\,e^{-\alpha t}, \qquad \Delta_0 = 0. \tag{18}$$

Unrolling the recurrence and bounding the sum by an integral for small $\eta\rho$,

$$\Delta_T \;\le\; \frac{2\eta L}{n}\sum_{t=0}^{T-1}(1+\eta\rho)^{T-1-t}e^{-\alpha t} \;\approx\; \frac{2\eta L}{n}\int_0^T e^{\eta\rho(T-t)-\alpha t}\,dt \;=\; \frac{2\eta L}{n}\cdot\frac{e^{\eta\rho T}\big(1-e^{-(\eta\rho+\alpha)T}\big)}{\eta\rho+\alpha}. \tag{19}$$

For $\alpha > \eta\rho$ the right-hand side is bounded by $\frac{2\eta L}{n(\alpha-\eta\rho)}\big(1-e^{-(\alpha-\eta\rho)T}\big)$. Since the loss is L-Lipschitz in $\boldsymbol{\theta}$, the uniform stability parameter satisfies $\beta \le L\,\Delta_T$, which gives the stated bound. Without annealing, the sample-mismatch term is $2\eta L/n$ at every step and the same recurrence gives $\Delta_T \le \frac{2\eta L}{n}\cdot\frac{(1+\eta\rho)^T-1}{\eta\rho} \le \frac{2\eta LT}{n}e^{\eta\rho T}$, the bound for a hard cut at $t = T_w$. □

A.4.4 MASK-AGNOSTIC GUARANTEES

The three propositions below hold for any binary weight mask and are summarised in Section 4.4. They transfer unchanged to the global allocation of Eq. (10), but because they do not depend on which weights the mask keeps, they cannot by themselves justify one criterion over another. Let f be an L-layer network with 1-Lipschitz activations and weight matrices $\mathbf{W}_i$, let f_o be the dense model and f_s the ticket with $\mathbf{W}_i^s = \mathbf{M}_i \odot \mathbf{W}_i^o$, and let $\|\cdot\|_\sigma$, $\|\cdot\|_F$ and $\|\cdot\|_{2,1}$ denote the spectral, Frobenius and column-wise $\ell_{2,1}$ norms.

Proposition 3 (Lipschitz bound)**.** $\mathrm{Lip}(f_s) \le \prod_{i=1}^{L} \|\mathbf{W}_i^s\|_\sigma \le \prod_{i=1}^{L} \|\mathbf{W}_i^o\|_F$ *for any binary mask, so the Lipschitz certificate of the ticket,* γ/Lip *for a classification margin* γ*, never shrinks under pruning.*

Proof. The Lipschitz constant of a composition is at most the product of the layer constants, each layer $\mathbf{z} \mapsto \phi(\mathbf{W}_i\mathbf{z})$ has constant at most $\|\mathbf{W}_i\|_\sigma$ because ϕ is 1-Lipschitz, and $\|\mathbf{W}\|_\sigma \le \|\mathbf{W}\|_F$. Masking removes squared entries from $\|\mathbf{W}\|_F^2$, which cannot increase it. The certificate statement follows from $\epsilon_{\mathrm{cert}} = \gamma/\mathrm{Lip}$. □

Proposition 4 (Frobenius complexity)**.** *Define the Frobenius-normalised complexity* $\widetilde{M}(f) = \big(\prod_i \|\mathbf{W}_i\|_F\big)\big(\sum_i \|\mathbf{W}_i\|_{2,1}^{2/3}/\|\mathbf{W}_i\|_F^{2/3}\big)^{3/2}$*. If the mask removes at least one non-zero weight in every layer, then* $\widetilde{M}(f_s) < \widetilde{M}(f_o)$*, and with probability at least* $1-\delta$ *over a sample of size* n *with inputs bounded by* $\|\mathbf{x}\|_2 \le B$*,*

$$R(f_s) \;\le\; \widehat{R}_\gamma(f_s) + \widetilde{O}\Big(\frac{B\,\widetilde{M}(f_s)}{\gamma\sqrt{n}} + \sqrt{\frac{\ln(1/\delta)}{n}}\Big). \tag{20}$$

The bound needs no monotonicity of the per-layer spectral norms, and global allocation meets its condition whenever no layer is left intact.

Proof. Let $r_i = \|\mathbf{W}_i^s\|_F/\|\mathbf{W}_i^o\|_F$ and $q_i = \|\mathbf{W}_i^s\|_{2,1}/\|\mathbf{W}_i^o\|_{2,1}$. Removing a non-zero entry strictly decreases the Frobenius norm and the ℓ_2 norm of the affected column, so $r_i < 1$ and $q_i < 1$ for every layer. Writing $w_i = (\|\mathbf{W}_i^o\|_{2,1}/\|\mathbf{W}_i^o\|_F)^{2/3} > 0$, a direct computation gives

$$\Big(\frac{\widetilde{M}(f_s)}{\widetilde{M}(f_o)}\Big)^{2/3} = \frac{\sum_i w_i\, q_i^{2/3} \prod_{j\neq i} r_j^{2/3}}{\sum_i w_i} < 1, \tag{21}$$

because every term in the numerator is strictly smaller than the corresponding w_i. For the bound, the class $\{f : \prod_i \|\mathbf{W}_i\|_F \le s,\ \sum_i \|\mathbf{W}_i\|_{2,1}^{2/3}/\|\mathbf{W}_i\|_F^{2/3} \le r\}$ is contained in the spectrally-normalised class of Bartlett et al. (2017) with the same radii since $\|\cdot\|_\sigma \le \|\cdot\|_F$, so its covering numbers, and by Dudley's entropy integral its Rademacher complexity, are bounded by $\widetilde{O}(sr/\sqrt{n})$; the margin bound follows. The reference term of Bartlett et al. (2017) is $\|\mathbf{W}_i - \mathbf{W}_i^{(0)}\|_{2,1}$, and the statement above takes $\mathbf{W}_i^{(0)} = \mathbf{0}$. □

Proposition 5 (PAC-Bayes generalization)**.** *Let* f_s *keep exactly* k *of the* d *weights of* f_o*, each quantised to one of* 2^b *values, and let the loss be bounded in* $[0,1]$*. With probability at least* $1-\delta$ *over a sample of size* n*,*

$$R(f_s) \;\le\; \widehat{R}(f_s) + \sqrt{\frac{kb + k\ln(ed/k) + \ln(1/\delta)}{2n}}, \tag{22}$$

a description length stated over a global selection of surviving positions, which is the regime of Eq. (10)*.*

Proof. The set $\mathcal{H}_{k,b}$ of networks obtainable by choosing k of d positions and one of 2^b values for each has cardinality $\binom{d}{k}2^{kb} \le (ed/k)^k 2^{kb}$. Hoeffding's inequality for a fixed hypothesis and a union bound over $\mathcal{H}_{k,b}$ give the result with $\ln|\mathcal{H}_{k,b}| \le k\ln(ed/k) + kb$, absorbing $\ln 2$ into the bit convention. This is the PAC-Bayes compression bound of Dziugaite & Roy (2017) with a uniform prior on $\mathcal{H}_{k,b}$ and a point-mass posterior at f_s; it is stated over global selections of k positions, which is the regime of Eq. (10). □

A.5 A WORKED EXAMPLE OF CURVATURE-BASED EDGE PRUNING

This appendix carries one small graph through scoring, ranking and cutting. The score values are illustrative and chosen to show the mechanism; every number attributed to Cora or Citeseer elsewhere in the paper is measured.

The graph. Eight nodes in two communities, $A = \{0, 1, 2, 3\}$ and $B = \{4, 5, 6, 7\}$, with clean edges $0{-}1$, $0{-}2$, $1{-}2$, $2{-}3$, $4{-}5$, $4{-}6$, $5{-}6$, $6{-}7$ and one injected edge $0{-}7$. The labelled nodes are $\{0, 1\}$ and the model is a two-layer GCN, so a directed entry can move the loss only if its target lies within one hop of a labelled node, that is in $\{0, 1, 2, 7\}$. Node 7 is reachable only through the injected edge.

Which entries the score can see. Of the 18 directed entries, the 10 whose target is in $\{0, 1, 2, 7\}$ receive a non-zero score and the remaining 8, seven inside community B and the entry $2 \to 3$, receive exactly zero (Table 3). This is the support of Proposition 1; on the real Cora graph the corresponding fractions are 51.0% and 49.0%.

Table 3: Directed entries of the toy graph ranked by curvature. Values are illustrative.

rank	entry	$s_e^{\mathbf{a}}$	comment
1	$0 \to 7$	0.91	injected, into a labelled node
2	$7 \to 0$	0.88	injected, mirror
3	$0 \to 1$	0.42	clean, between two labelled nodes
4	$1 \to 0$	0.40	clean, mirror
5–8	$2 \to 0,\ 0 \to 2,\ 1 \to 2,\ 2 \to 1$	0.21–0.13	clean
9–10	$3 \to 2,\ 6 \to 7$	0.05, 0.04	clean, periphery
11–18	all entries with target in $\{3, 4, 5, 6\}$	0.00	outside the support, tied

Cutting at three budgets. The method scores undirected edges by the sum of the two directed curvatures (Eq. (6)) and removes $k_e = \lfloor p_e |\mathcal{E}| \rfloor$ of them by rank. The summed scores are 1.79 for the injected edge $\{0, 7\}$, 0.82 for $\{0, 1\}$, about 0.3 for $\{0, 2\}$ and $\{1, 2\}$, 0.05 for $\{2, 3\}$, 0.04 for $\{6, 7\}$ and exactly zero for the three edges inside community B, so the coverage of Eq. (7) is six of nine edges. At $p_e = 22.6\%$, sparsity budget $r = 5$ of the lattice, $k_e = 2$ and the cut removes the injected edge and the clean edge $0{-}1$, which is influential because it joins two labelled nodes; high curvature means influential, not malicious, and the criterion cannot tell the two apart. At $p_e = 48.7\%$, $r = 13$, $k_e = 4$ and every removal is still ranked by a real curvature while community B is untouched. At $p_e = 62.3\%$, $r = 19$, $k_e = 5$ removes the last edge with a non-zero score, and one budget further the removal must come from the three-way tie at zero, where no tie-break carries information. This is the regime $p_e > c(\mathbf{A}')$ of Section 4.1. Summing the two directions scores the edge $\{2, 3\}$ once, from its informative direction $3 \to 2$ whose target is one hop from node 1, instead of recording one useful and one worthless entry; on Cora this raises the coverage from 54% to 70% of edges.

A.6 SUPPORT AND COVERAGE

Table 4: Coverage of the edge score, per directed entry and per undirected edge after summing (Eq. (7)), and the first sparsity budget r at which the edge budget exceeds it. Heavier attacks widen the support because injected edges attach to labelled nodes.

dataset / attack / rate	per entry	per edge	budget exceeds coverage at
Cora / PGD / 5%	51.0%	69.6%	$r = 24$
Cora / PGD / 20%	54.1%	70.2%	$r = 24$
Cora / MetaAttack / 5%	55.6%	74.5%	$r = 27$
Cora / MetaAttack / 20%	67.6%	83.1%	$r = 35$
Citeseer / PGD / 5%	51.7%	67.5%	$r = 22$
Citeseer / PGD / 20%	57.9%	72.3%	$r = 26$
Citeseer / MetaAttack / 5%	55.3%	72.3%	$r = 26$
Citeseer / MetaAttack / 20%	66.0%	81.4%	$r = 33$
PubMed / PGD / 20%	77.2%	–	–

The gradient of the supervised loss with respect to the edge weights has exactly the same support as the curvature on every graph we measured (Cora PGD 5%, Cora MetaAttack 20%, Citeseer PGD 5%; 0 of 5,018, 3,931 and 3,649 blind entries are rescued by a first-order term), so no combination of gradient and curvature extends the coverage. On Cora under PGD at 20%, per-entry scoring

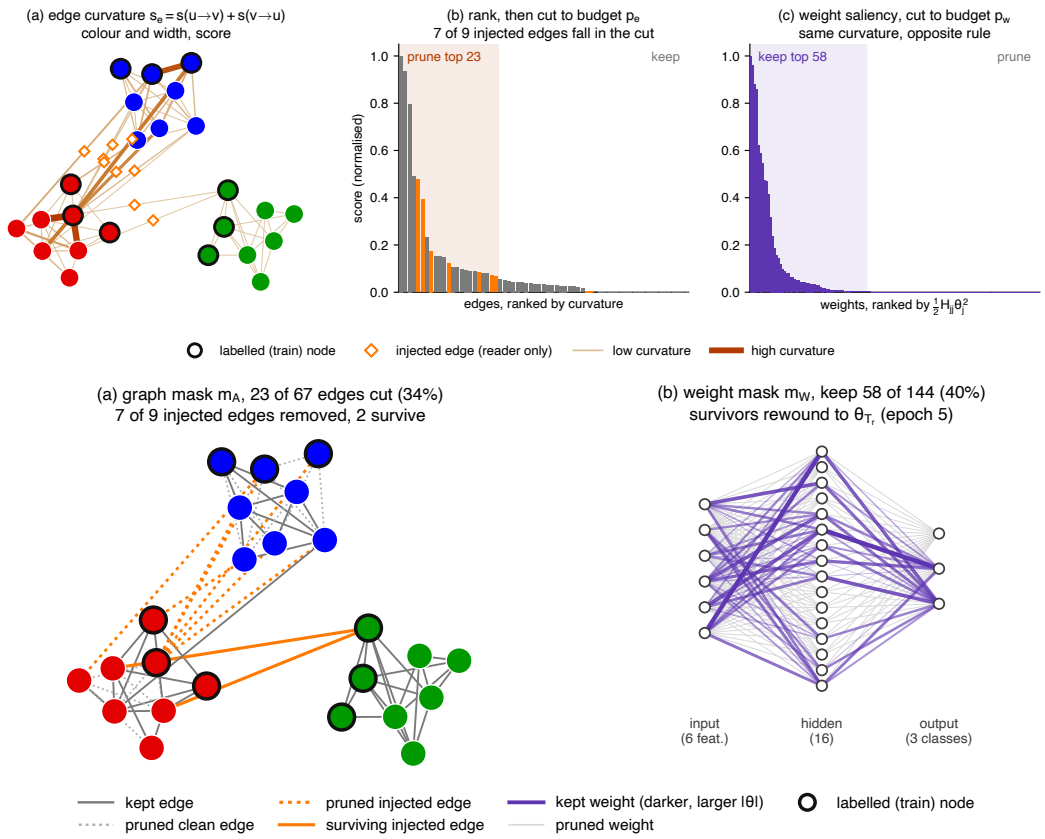

Figure 7: Steps 2 and 3 of CA-AWT run on a second toy graph, 24 nodes in three classes with nine labelled nodes, 67 edges of which nine were injected by a loss-raising attacker, and a two-layer GCN with 144 weights. Top, edge curvature (colour and width), the ranked edge scores with the cut at the budget, and the ranked weight saliencies with the opposite rule. Bottom, the resulting graph mask and weight mask, with survivors rewound to $\boldsymbol{\theta}_{T_r}$.

exhausts its information beyond budget $r = 15$ and summed scoring only at $r = 24$, past the end of the lattice, so at every budget of the lattice every removal is informed.

Selection by rank. On Cora under PGD at 20%, 4,928 of 10,740 directed entries score exactly zero. A rule that selects by comparing scores to a threshold value fails once the budget exceeds the non-zero fraction, because the threshold itself becomes zero and the test keeps nothing, which on this graph would delete the whole graph at every budget above 54%. Eq. (10) selects by rank, removes exactly k_e edges, and breaks ties by index.

Symmetric versus per-entry pruning. At budget $r = 13$ on Cora, per-entry pruning keeps 3,623 edges of which 1,733 (47.8%) survive in only one direction, so the pruned graph is no longer undirected and the reported edge count has three different values (5,513 directed entries, 3,623 edges with at least one entry, 1,890 with both). Summed scoring keeps 2,757 whole edges and leaves no one-directional survivor.

A.7 Experimental details

Datasets and splits. Table 5 lists the graphs. The three citation graphs split at random into 10% training, 10% validation and 80% test nodes, exactly as in ARGS and UGS, so that every cell of our sweep is comparable. Chameleon, a Wikipedia page network with edge homophily 0.23, follows the same protocol and is used only for the heterophily probe. OGBN-ArXiv uses the standard OGB split. For the GCN backbone the node features are the raw bag-of-words vectors; for the GIN backbone they are ℓ_1 row-normalised.

Models. The backbone on the citation graphs and on Chameleon is the two-layer GCN of Kipf & Welling (2017) with 512 hidden units, the architecture ARGS and UGS report on, with symmetric

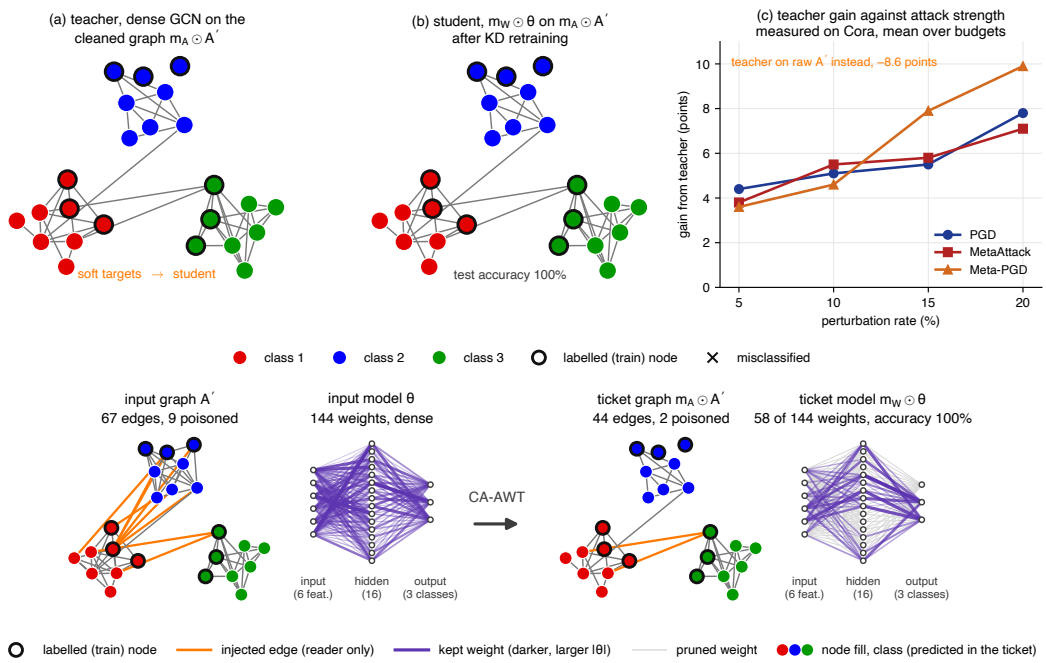

Figure 8: Step 4 and the output on the same toy graph. Top, the dense teacher on the cleaned graph, the sparse student after distillation, and the measured gain from the teacher against attack strength on Cora (Table 12). Bottom, the input graph and dense model against the final doubly-sparse ticket.

Table 5: Datasets. Node and edge counts are for the largest connected component, as used by ARGS; homophily is the edge homophily ratio, the fraction of edges joining nodes of the same class.

dataset	nodes	edges	classes	features	homophily	attacks evaluated
Cora	2,485	5,069	7	1,433	0.81	PGD, MetaAttack, Meta-PGD
Citeseer	2,110	3,668	6	3,703	0.74	PGD, MetaAttack, Meta-PGD
PubMed	19,717	44,338	3	500	0.80	PGD
Chameleon	2,277	31,371	5	2,325	0.23	PGD
OGBN-ArXiv	169,343	1,166,243	40	128	0.65	PR-BCD

normalisation $\mathbf{D}^{-1/2}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-1/2}$ and dropout 0.5; the layer aggregates first and transforms second, following ARGS's implementation, and accepts a per-entry edge weight so that the relaxation of Section 3 applies without any change to the model. The input layer holds $733{,}696$ weights on Cora and the classifier $3{,}584$, which is the disparity that makes the global allocation of Eq. (10) matter. On OGBN-ArXiv the backbone is the 28-layer residual GCN with width 128 that ARGS reports on; its convolution is a GCN layer rather than GENConv because GENConv accepts no per-edge weight and therefore cannot carry an edge mask. The GIN backbone is a two-layer graph isomorphism network with 512 hidden units and dropout 0.5; each layer sums the neighbour features weighted by the edge weights, adds the node's own features, and applies one linear map followed by a ReLU, with $\epsilon = 0$. It is evaluated on Cora at 5 and 10% and Citeseer at 15 and 20% under the three attacks, with ℓ_1 row-normalised features. The teacher has the same architecture as the student and is trained dense.

Baselines. UGS (Chen et al., 2021) learns a differentiable mask on the adjacency and on the weights jointly with the model under an ℓ_1 penalty, prunes 5% of the edges and 20% of the weights per round, rewinds and repeats. ARGS (Chowdhury et al., 2023; 2024) keeps this loop and adds a feature-smoothness loss and a cross-entropy term on MLP pseudo-labelled test nodes to the mask objective. Both are evaluated at the same sparsity budgets as CA-AWT. STRG (Li et al., 2023) trains a dense GNN with self-training on pseudo-labels for the test nodes and has no sparsity. The learned-magnitude arm of Section 6.2 is our own re-implementation of the UGS mask inside the CA-AWT pipeline, a per-edge mask trained jointly with the student across all retraining epochs with the same ℓ_1 subgradient and cut once at the end to the same budget, so that the only difference from CA-AWT is the edge criterion.

Attacks. Cora and Citeseer PGD graphs are the files shipped with ARGS. MetaAttack and Meta-PGD graphs on Cora and Citeseer, and PGD graphs on PubMed and Chameleon, were generated with DeepRobust at perturbation rates of 5, 10, 15 and 20% of the edge count, with flip counts verified against the budget. The attacker's edges are recovered by differencing the clean and poisoned graphs; the symmetric difference matches the budget within 3% on all 28 dataset–attack–rate combinations. PubMed is reported under PGD only.

Backbone and training. Two-layer GCN with 512 hidden units and dropout 0.5, Adam with learning rate 0.01 and weight decay $5 \cdot 10^{-4}$, $T_w = 30$, $T_r = 5$, $T = 300$, $P = 20$, $\alpha = 0.1$, teacher trained for 200 epochs with best-validation checkpointing, $\beta = 0.7$, $\tau = 1$, schedule threshold $p^\star = 0.92$ with T_{ramp} equal to 60% of the retraining span. The teacher budget r_t is selected on validation over $r_t = 1, \ldots, 20$ once per graph, each candidate a dense training of 200 epochs; on OGBN-ArXiv the sweep stops three budgets after the best one. Every cell is the mean of three seeds. The achieved edge and weight sparsity is recorded for every run and never deviates from the target.

A.8 Hyper-parameter robustness

Table 6: Distillation weight and temperature, Cora under PGD, sparsity budgets $r = 5$, 13 and 19, six seeds, selection on validation only. Ten of twelve pairs are tied on validation within two standard errors.

β	τ	validation	test
0.3	1 / 2 / 4	81.59 / 82.27 / 81.56	67.67 / 68.70 / 68.09
0.5	1 / 2 / 4	82.32 / 82.25 / 81.24	68.25 / 68.98 / 68.03
0.7	**1** / 2 / 4	**82.39** / 81.76 / 81.00	68.91 / 68.83 / 67.85
0.9	1 / 2 / 4	81.85 / 81.29 / 80.88	68.93 / 68.39 / 67.63

The validation-selected pair ($\beta = 0.7$, $\tau = 1$) scores 68.91 on test against 68.98 for the best-on-test pair, a difference of 0.07 points. Validation identifies only the bad corner, high temperature with high β. The rewind epoch $T_r \in \{1, 5, 15, 25\}$ changes the mean over 36 cells by at most 0.7 points and, once the budget schedule is active, by less than 0.05. The warm-up length $T_w \in \{30, 100, 200\}$ at a fixed 300-epoch budget spans 0.75 points. Raising the probe count from $P = 20$ to 100 changes accuracy by 0.19 points and leaves 87 to 98% of the pruned set unchanged. The ratio $\lambda_{\max}(\mathbf{H}) / \max_j |H_{jj}|$, estimated by Lanczos on Hessian-vector products, is about 5 with adequate probes and warm-up, so the diagonal is a ranking signal and not an eigenvalue estimate.

A.9 Additional results

A.9.1 Accuracy against graph sparsity on every panel

Figures 9 to 12 extend Figure 3 to every graph, attack, rate and backbone, with the same conventions.

Table 7: CA-AWT accuracy (mean ± sd of three seeds) and attacker-edge recall in percent at sparsity budgets $r = 5$, 13 and 19, Citeseer at 10 and 15% and PubMed at 5 and 10% perturbation, GCN.

	$r = 5$		$r = 13$		$r = 19$	
Dataset / attack / rate	acc.	recall	acc.	recall	acc.	recall
Citeseer / PGD / 10%	74.0±0.8	96	74.1±1.0	100	74.0±0.7	100
Citeseer / MetaAttack / 10%	73.0±1.0	88	72.6±0.7	100	72.3±0.1	100
Citeseer / Meta-PGD / 10%	73.1±0.3	80	72.2±0.2	95	71.7±0.3	97
Citeseer / PGD / 15%	75.3±0.3	94	75.3±0.5	100	75.0±0.7	100
Citeseer / MetaAttack / 15%	74.8±0.3	82	73.1±0.1	99	72.3±0.4	100
Citeseer / Meta-PGD / 15%	74.0±2.6	68	72.3±0.8	90	71.9±0.5	95
PubMed / PGD / 5%	84.6±0.5	95	83.7±0.3	100	82.0±0.2	100
PubMed / PGD / 10%	83.4±0.5	90	82.3±0.6	99	80.0±0.7	99

A.9.2 Edge criterion, full comparison

Figure 13 shows the recall of the two criteria at every sparsity budget of the lattice and Table 8 gives accuracy and recall on every attack and rate. The gap in accuracy follows the gap in recall on every panel and grows with the perturbation rate under all three attacks.

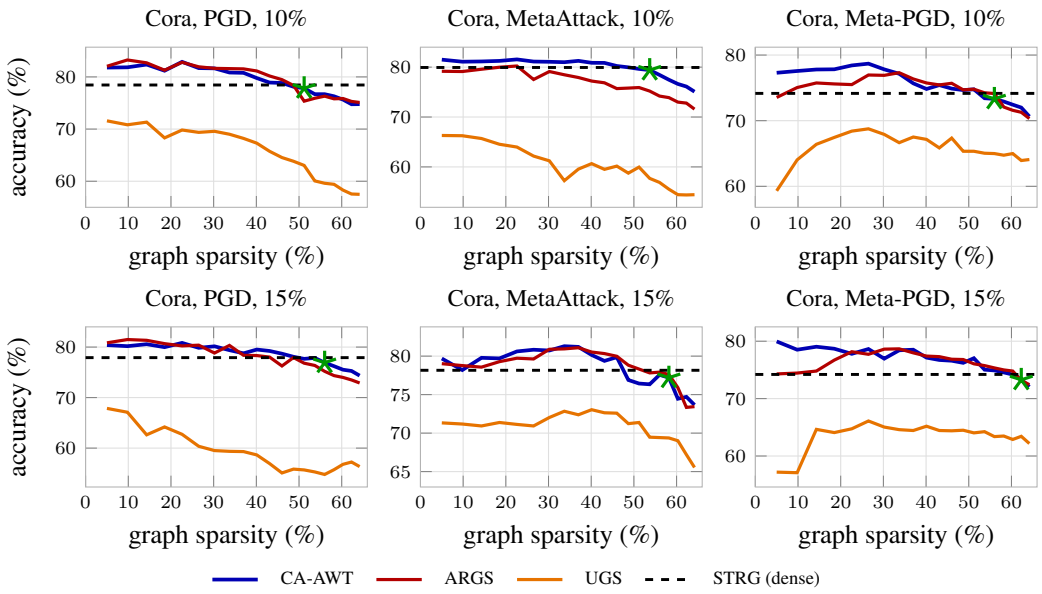

Figure 9: Accuracy against graph sparsity as in Figure 3, Cora at 10 and 15% perturbation, GCN.

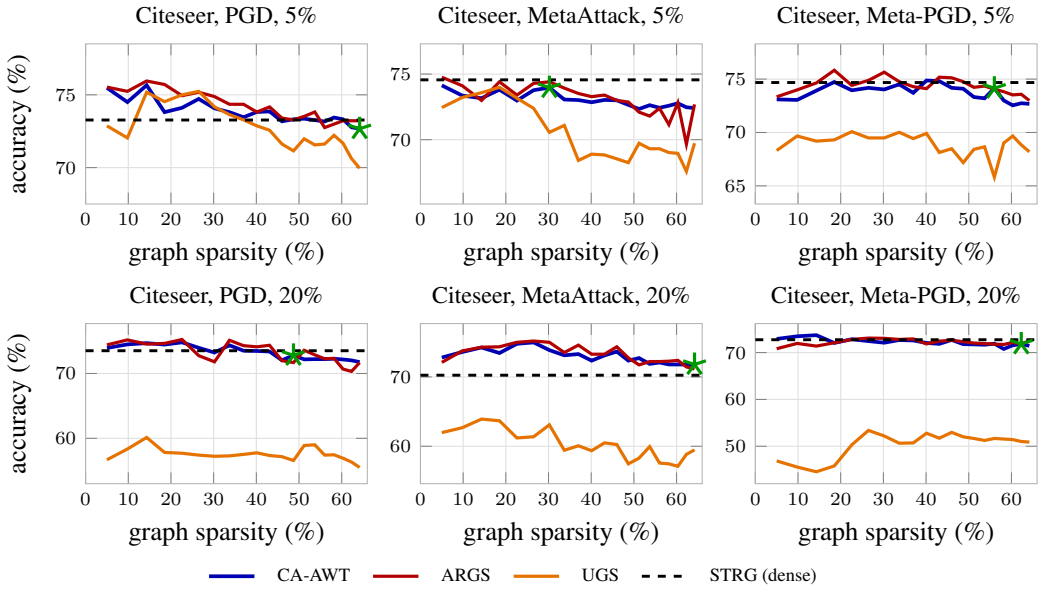

Figure 10: Accuracy against graph sparsity as in Figure 3, Citeseer at 5 and 20% perturbation, GCN.

A.9.3 ATTACKER-EDGE REMOVAL BY NODE CATEGORY

ARGS reports its removal of adversarial edges by the category of the two endpoints on Cora under PGD at 20% after all of its rounds. Table 9 places CA-AWT at the same budget on the same graph. The 651 injected edges present in that graph split into 120 train–train, 464 train–test and 8 test–test edges, with the remaining 59 touching a validation node.

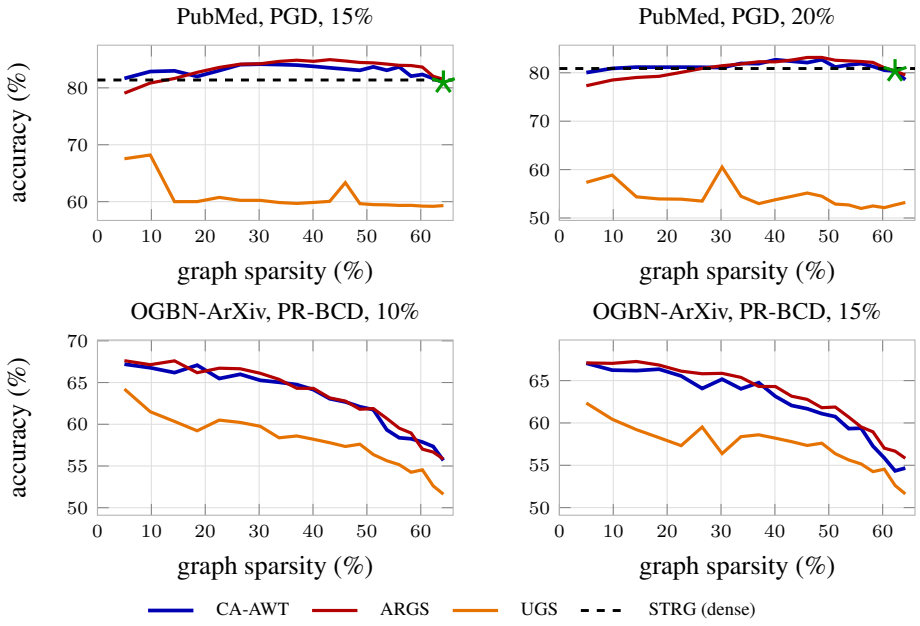

Figure 11: Accuracy against graph sparsity as in Figure 3, PubMed under PGD at 15 and 20% and OGBN-ArXiv under PR-BCD at 10 and 15% perturbation, GCN.

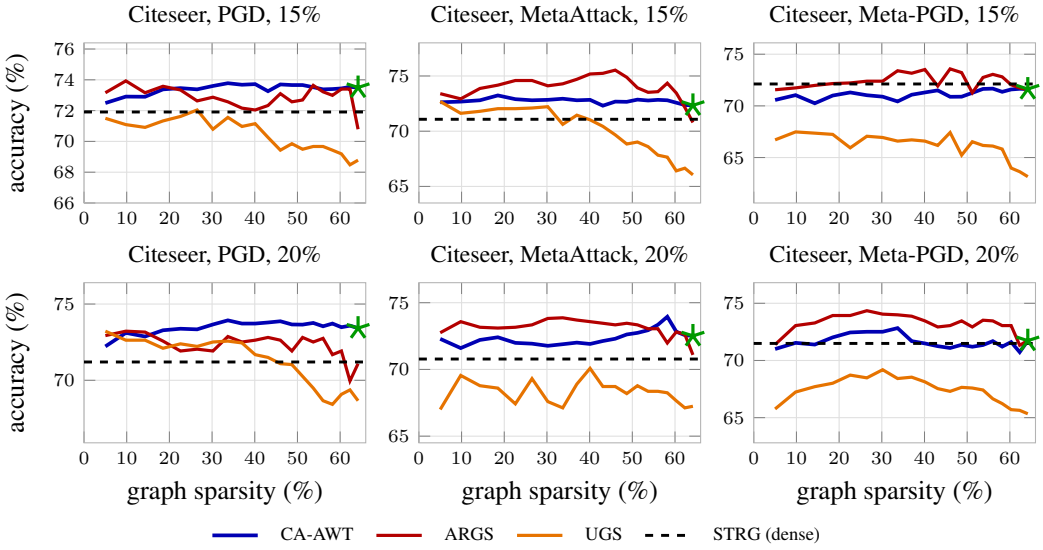

Figure 12: Accuracy against graph sparsity as in Figure 3, Citeseer at 15 and 20% perturbation, GIN.

A.9.4 GIN BACKBONE

Table 10 gives the GIN backbone at the three sparsity budgets; the full curves are Figure 4 for Cora and Figure 12 for Citeseer.

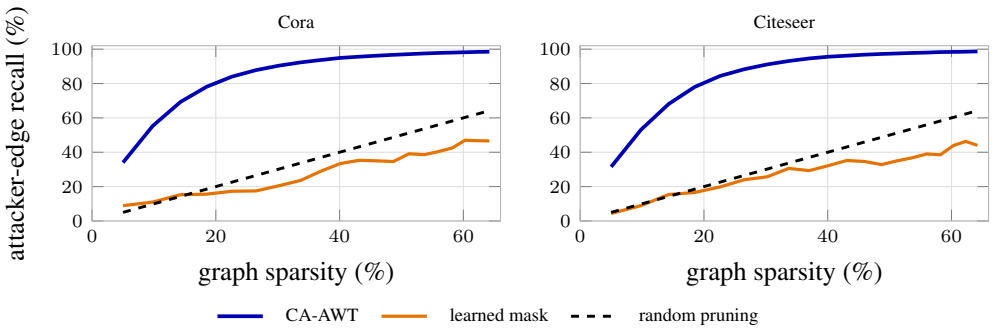

Figure 13: Attacker-edge recall of CA-AWT and of the learned magnitude mask at every sparsity budget $r = 1$ to 20, mean over attacks, rates and seeds; dashed, random pruning.

Table 8: The edge criterion on every panel, accuracy and attacker-edge recall, mean over sparsity budgets $r = 1$ to 20 and three seeds.

			accuracy (%)			recall (%)	
dataset	attack	rate	CA-AWT	mask	Δ	CA-AWT	mask
Cora	PGD	5%	81.7	75.9	+5.8	98	20
Cora	PGD	10%	79.2	72.5	+6.6	95	25
Cora	PGD	15%	78.5	67.3	+11.1	92	29
Cora	PGD	20%	77.8	66.0	+11.8	89	26
Cora	MetaAttack	5%	80.4	73.7	+6.6	96	30
Cora	MetaAttack	10%	79.8	70.1	+9.7	92	30
Cora	MetaAttack	15%	78.4	64.2	+14.3	89	30
Cora	MetaAttack	20%	77.2	56.6	+20.5	85	23
Cora	Meta-PGD	5%	77.4	71.3	+6.1	81	34
Cora	Meta-PGD	10%	75.4	59.8	+15.6	83	33
Cora	Meta-PGD	15%	76.7	52.9	+23.8	80	30
Cora	Meta-PGD	20%	77.2	51.3	+25.9	73	34
Citeseer	PGD	5%	73.8	72.8	+1.0	96	12
Citeseer	PGD	10%	74.2	71.4	+2.8	94	35
Citeseer	PGD	15%	75.1	68.0	+7.1	92	24
Citeseer	PGD	20%	73.2	66.9	+6.3	87	34
Citeseer	MetaAttack	5%	73.0	72.5	+0.5	96	31
Citeseer	MetaAttack	10%	72.9	68.8	+4.0	91	28
Citeseer	MetaAttack	15%	73.2	66.9	+6.4	88	34
Citeseer	MetaAttack	20%	73.0	57.7	+15.2	86	27
Citeseer	Meta-PGD	5%	73.7	65.8	+8.0	88	32
Citeseer	Meta-PGD	10%	72.8	53.9	+18.9	84	21
Citeseer	Meta-PGD	15%	72.4	49.5	+22.9	77	32
Citeseer	Meta-PGD	20%	72.2	43.6	+28.6	74	30

Table 10: CA-AWT and ARGS accuracy at sparsity budgets $r = 5$, 13 and 19, mean ± sd of three seeds, Cora at 5 and 10% and Citeseer at 15 and 20% perturbation, GIN; bold where CA-AWT exceeds ARGS.

	$r = 5$		$r = 13$		$r = 19$	
Dataset / attack / rate	ARGS	CA-AWT	ARGS	CA-AWT	ARGS	CA-AWT
Cora / PGD / 5%	77.4	**79.8**±0.8	74.9	**78.3**±0.6	72.1	**73.1**±2.8
Cora / PGD / 10%	76.1	**78.9**±1.2	75.0	**77.3**±0.8	71.9	71.8±0.5
Cora / MetaAttack / 5%	77.6	**79.3**±1.1	75.1	**78.6**±1.0	71.0	**74.8**±1.6
Cora / MetaAttack / 10%	77.1	**77.6**±0.9	74.6	**75.5**±0.3	72.8	**73.3**±0.2
Cora / Meta-PGD / 5%	77.7	77.4±1.4	75.6	75.2±1.8	73.6	73.1±1.2
Cora / Meta-PGD / 10%	77.2	76.0±1.8	75.5	75.5±0.3	73.8	72.8±1.1
Citeseer / PGD / 15%	73.3	**73.5**±0.4	72.6	**73.7**±0.2	73.4	**73.5**±0.7
Citeseer / PGD / 20%	71.9	**73.4**±0.5	71.9	**73.7**±0.9	70.0	**73.6**±0.7
Citeseer / MetaAttack / 15%	74.6	72.9±0.5	74.9	72.7±0.3	72.2	**72.4**±0.6
Citeseer / MetaAttack / 20%	73.2	72.0±1.4	73.5	72.6±0.5	72.6	72.6±0.7
Citeseer / Meta-PGD / 15%	72.2	71.3±1.9	73.2	70.9±1.1	71.7	71.7±0.2
Citeseer / Meta-PGD / 20%	73.9	72.5±1.2	73.5	71.4±0.8	71.3	70.7±0.4

Table 9: Share of injected edges removed by endpoint category, Cora, ARGS's PGD 20% graph, $r = 20$ (graph sparsity 64.2%, model sparsity 98.9%).

	train–train	train–test	test–test
ARGS (homophily prior, iterative)	68.1%	47.3%	14.3%
CA-AWT (curvature, one shot)	100.0%	100.0%	100.0%

A.9.5 Component ablations by sparsity budget

Table 11 gives the accuracy behind Table 2 at each of the three sparsity budgets. Every entry is the mean over three attacks, 5 and 20% perturbation and three seeds.

Table 11: Component ablations by sparsity budget, accuracy in percent, mean over three attacks, two perturbation rates and three seeds, GCN.

	Cora			Citeseer		
Variant	$r = 5$	$r = 13$	$r = 19$	$r = 5$	$r = 13$	$r = 19$
CA-AWT (full recipe)	81.0	77.7	74.5	73.9	72.8	72.3
w/o teacher	73.4	74.8	70.3	66.8	71.6	70.4
w/o curvature edge score	68.9	63.8	59.2	63.8	61.7	62.8
w/o cleaned graph for the teacher	72.1	69.5	65.9	62.6	62.7	62.6
w/o global weight budget	80.2	76.4	66.7	73.0	72.2	65.0
w/o summed edge score	80.0	75.2	72.5	72.0	71.8	72.0
w/o budget schedule	80.5	76.7	70.5	72.7	72.1	69.8
w/o OBD saliency	80.8	76.8	73.8	72.5	72.0	71.8

Two components are budget-dependent. The global weight budget and the budget schedule cost almost nothing at $r = 5$ and $r = 13$ and then cost 7.8 and 4.0 points on Cora at $r = 19$, which is where model sparsity reaches 98.6%. The teacher's training graph costs between 8 and 11 points at every budget on both datasets, and the edge criterion costs more as the cut deepens.

A.9.6 Teacher gain by sparsity budget and by attack strength

Table 12 gives the gain from the teacher by sparsity budget and by attack strength. It rises with attack strength on both datasets, from between $+1.3$ and $+4.4$ points at 5% to between $+7$ and $+10$ points at 20%. Across budgets it is U-shaped. It is largest at $r = 1$, falls to its minimum of $+2.8$ on Cora at $r = 13$ and $+1.1$ on Citeseer at $r = 15$, which is where the budget schedule of Eq. (13) switches on and takes over part of the teacher's work, and rises again afterwards. Three modifications of the teacher were tested and none paid, hidden-layer feature distillation (-0.12), an ensemble of three teachers (0.00) and 600 instead of 200 teacher epochs ($+0.07$). Training the teacher on the raw poisoned graph costs 8.6 points on Cora and 10.4 on Citeseer (Table 11).

Table 12: Gain from the teacher of Eq. (14) over the same recipe without it, in accuracy points, sparsity budgets $r = 1$ to 20, three attacks, four rates, three seeds. Top, gain by budget, averaged over attacks and rates. Bottom, gain by attack and perturbation rate, averaged over budgets.

budget r	1	2	5	7	10	13	16	20
Cora	+14.0	+11.8	+6.8	+4.7	+3.4	+2.8	+3.7	+5.0
Citeseer	+17.5	+13.9	+6.2	+3.7	+1.9	+1.3	+1.4	+2.4

dataset	attack	5%	10%	15%	20%
Cora	PGD	+4.4	+5.1	+5.5	+7.8
	MetaAttack	+3.8	+5.5	+5.8	+7.1
	Meta-PGD	+3.6	+4.6	+7.9	+9.9
Citeseer	PGD	+1.6	+4.4	+6.8	+8.5
	MetaAttack	+1.3	+2.7	+5.1	+7.6
	Meta-PGD	+3.4	+4.3	+6.0	+9.5

A.9.7 HETEROPHILY PROBE ON CHAMELEON

Table 13: Chameleon (edge homophily 0.23), DeepRobust PGD, three seeds, mean ± sd. The dense GCN reference is 54.2 on the clean graph and 41.1 on the graph poisoned at 5%. PGD is deletion-heavy on this graph (at 20% it removes 2,971 clean edges and adds 3,292).

			CA-AWT		learned magnitude mask		random
rate	r	edges / weights cut	acc	recall	acc	recall	recall
5%	5	23% / 67%	39.6 ± 0.7	94.3 ± 1.0	38.1 ± 0.8	16.6 ± 3.6	22.6
5%	13	49% / 95%	36.4 ± 0.7	97.5 ± 0.7	36.8 ± 2.4	11.8 ± 7.8	48.7
5%	19	62% / 99%	34.8 ± 0.0	97.9 ± 0.5	37.5 ± 1.3	19.2 ± 6.7	62.3
20%	5	23% / 67%	33.8 ± 1.6	87.0 ± 0.4	32.3 ± 1.6	19.4 ± 12.7	22.6
20%	13	49% / 95%	32.9 ± 2.6	96.6 ± 0.1	30.3 ± 2.5	24.0 ± 8.2	48.7
20%	19	62% / 99%	28.2 ± 1.4	98.2 ± 0.1	27.9 ± 0.3	26.1 ± 3.7	62.3

The detector transfers unchanged, with recall between 87 and 98% and a seed standard deviation under 1.2 points, while the learned magnitude mask stays below random in every cell and varies by up to 13 points across seeds, consistent with ranking blind entries by initialisation. On accuracy CA-AWT is ahead of the learned mask by 1.5, 2.6 and 0.3 points under 20% perturbation and by 1.5 points at 5% in the densest cell, and behind by 0.4 and 2.7 points in the two sparsest cells at 5%, where removing the most influential edges costs a GCN more than removing the injected ones recovers.

A.9.8 TEACHER BUDGET SELECTION

Table 14: Budget selected for the teacher, three seeds. The injected share is the fraction of edges in the observed graph that the attacker added; the cut and the recall are those of the curvature ranking at the selected budget.

dataset	attack	rate	injected share (%)	selected r_t	cut (%)	recall (%)	teacher acc.
Cora	PGD	5%	4.8	1	5	73	84.3
Cora	PGD	10%	9.1	1	5	46	80.9
Cora	PGD	15%	13.0	3	14	79	79.3
Cora	PGD	20%	16.7	9	37	98	76.5
Cora	MetaAttack	5%	4.8	2	10	81	81.6
Cora	MetaAttack	10%	9.1	3	14	78	80.4
Cora	MetaAttack	15%	13.0	6	26	90	76.8
Cora	MetaAttack	20%	16.7	8	34	91	76.8
Cora	Meta-PGD	5%	4.8	8	34	85	73.1
Cora	Meta-PGD	10%	9.1	6	26	80	73.6
Cora	Meta-PGD	15%	13.0	7	30	80	72.3
Cora	Meta-PGD	20%	16.7	10	40	81	70.7
Citeseer	PGD	5%	4.8	5	23	99	73.0
Citeseer	PGD	10%	9.1	5	23	96	71.6
Citeseer	PGD	15%	13.0	9	37	100	73.6
Citeseer	PGD	20%	16.7	10	40	97	70.9
Citeseer	MetaAttack	5%	4.8	5	23	98	71.4
Citeseer	MetaAttack	10%	9.1	6	26	93	71.8
Citeseer	MetaAttack	15%	13.0	11	43	99	71.8
Citeseer	MetaAttack	20%	16.7	12	46	98	71.0
Citeseer	Meta-PGD	5%	4.8	8	34	93	72.4
Citeseer	Meta-PGD	10%	9.1	11	43	94	71.5
Citeseer	Meta-PGD	15%	13.0	7	30	78	68.7
Citeseer	Meta-PGD	20%	16.7	14	51	89	69.7
PubMed	PGD	5%	4.8	4	19	91	84.5
PubMed	PGD	10%	9.1	3	14	74	83.3
PubMed	PGD	15%	13.0	7	30	89	82.5
PubMed	PGD	20%	16.7	9	37	89	81.5
Chameleon	PGD	5%	4.8	1	5	79	40.5
Chameleon	PGD	20%	16.7	1	5	41	31.4

Table 14 gives the selection for every panel with the teacher's accuracy. On Cora under PGD the selected cut grows from 5% at 5% perturbation to 37% at 20%, where the curvature ranking has

removed 73 and 98% of the injected edges. The selected budget rises with the perturbation rate in every column up to a wobble of one or two budgets, on PubMed and on Cora under Meta-PGD between 5 and 10% and on Citeseer under Meta-PGD between 10 and 15%. The selected cut is larger than the injected share on all but two panels, by a factor of one to seven, because recall at the cut is below one and some clean edges leave with the poison. The budget is therefore correlated with how heavily the graph is poisoned, and is not an estimate of the injected share itself.

Early stopping of the sweep. On every candidate curve of this paper, stopping the sweep once validation accuracy has not improved for k consecutive budgets selects the same teacher as the full sweep in 29, 52, 70, 88 and 94% of cases for $k = 1, 2, 3, 5, 8$, and trains on average 3.8, 6.8, 9.4, 12.6 and 15.5 of the 20 candidates. Stopping at the first decrease is unsafe, because the curves dip before their peak under heavy attack; $k = 5$ keeps the choice in 88% of cases at 63% of the cost. Every number in the paper uses the full sweep.