DEEP MESSAGE-PASSING GNNS HAVE A BACKWARD PROBLEM: KRONECKER-ALIGNED FEEDBACK TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Efforts to train deep message-passing graph neural networks have largely focused on modifying the forward computation to mitigate oversmoothing. However, preserving informative hidden representations alone does not ensure effective learning: supervision can still attenuate through the backward chain of graph propagation, feature transformations, and activation gates. In plain ten-layer GCN and GraphSAGE models on three citation graphs, backpropagation (BP) produces hidden-layer gradients too small to change any FP32 parameter, while frozen probes on the same hidden states remain well above chance. This motivates treating backward credit assignment as an explicit design choice. We introduce Kronecker-Aligned Feedback Training (KAFT), which exploits the Kronecker structure of graph-convolution Jacobians to deliver graph-aware feedback directly to hidden layers, leaving the forward and inference computation unchanged. KAFT combines a capped operator on the forward graph with learned feature-feedback matrices, aligned to directional targets obtained by propagating and normalizing probes through downstream weights. We show that this target satisfies a positive-semidefinite alignment property and give sufficient conditions under which the KAFT update descends and converges to stationarity. Across four node-classification datasets, four backbones, two depths, and three learning rates, KAFT improves accuracy over BP by 3.1 percentage points on average and is significantly better in 76 of 96 matched settings and significantly worse in 4; gains shrink on backbones with built-in identity paths such as GIN. The benefit grows once depth exceeds the point where BP stops updating: on DBLP it rises from 4.3 points at 8 layers to 30.4 points at 20 layers. With BatchNorm or LayerNorm in the same forward model, KAFT is significantly better in four of six comparisons and never significantly worse. In preliminary three-seed experiments, KAFT raises Cora link-prediction ROC-AUC from 64.5% to 81.4% with a six-layer encoder and matches BP on mini-batch graph classification. These results suggest that the backward rule, not only the forward operator, is a useful design axis for deep message-passing GNNs.

1 INTRODUCTION

Graph neural networks have become the standard tool for learning on relational data, from citation graphs to molecular structures (Kipf & Welling, 2017; Hamilton et al., 2017). Their representational power is tied directly to depth: each message-passing layer expands a node's receptive field by one hop, so multi-hop reasoning, long-range interaction tasks, and any prediction that depends on more than a node's immediate neighborhood requires a deep stack (Lukovnikov & Fischer, 2021; Dwivedi et al., 2022). Yet deep message-passing GNNs do not train (Li et al., 2018; Oono & Suzuki, 2020; Lukovnikov & Fischer, 2021). Beyond $L \approx 4-6$ on standard backbones, accuracy collapses, variance explodes, and additional layers actively hurt the model (Li et al., 2018; Rong et al., 2020; Chen et al., 2020). This is not a niche failure mode: it is the central obstacle to scaling GNNs to the kind of long-range tasks the architecture was designed for (Dwivedi et al., 2022; Arroyo et al., 2025).

Why deep MP-GNNs fail is unsettled. The dominant explanation has been forward-side oversmoothing: repeated graph propagation drives node representations toward an indistinguishable subspace, so deep models lose useful task signal at the activation level (Li et al., 2018; Oono & Suzuki, 2020). Recent work has reframed this picture, showing theoretically that GNN-specific spectral contraction causes vanishing gradients independently of forward collapse (Arroyo et al., 2025; Park et al., 2025;

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.

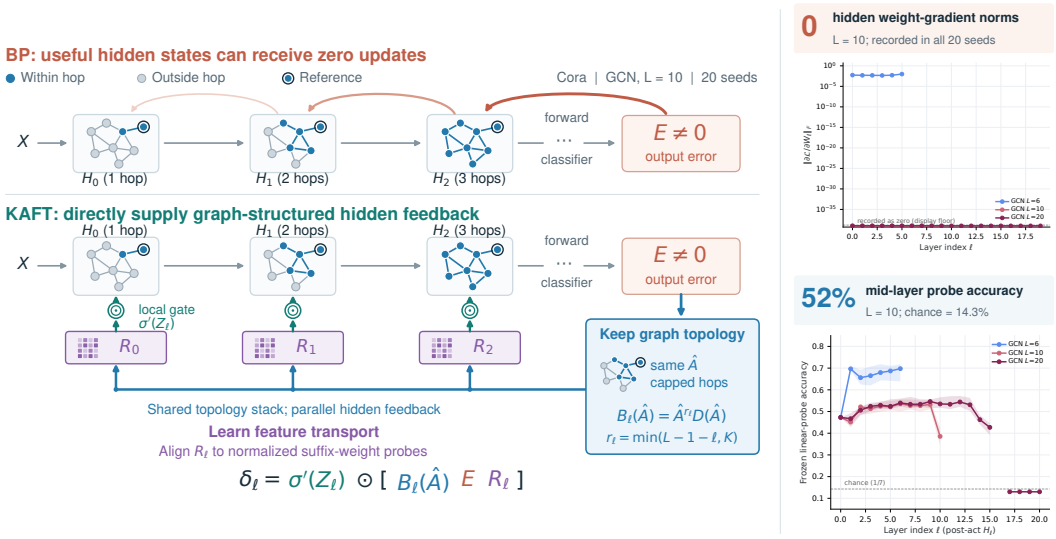


Figure 1: KAFT supplies graph-structured feedback to deep hidden layers. Left: BP transports the output error through the downstream chain; KAFT combines capped propagation on the forward graph with learned feature feedback and local activation gates. Right: plain GCNs trained on Cora at depths $L \in \{6, 10, 20\}$, over 20 seeds. Curves show medians and shaded bands show interquartile ranges. At $L = 10$ and 20, hidden weight-gradient norms are recorded as zero in FP32 and plotted at a display floor, while standardized hidden states retain above-chance probe accuracy in middle layers. The zero floor denotes the recorded FP32 norm.

Park & Kim, 2024; Peng et al., 2024; Tönshoff et al., 2023). However, no prior work has empirically separated backward-transport failure from forward-representational collapse on the same trained models. The diagnosis matters because it determines the kind of fix one should design. If the failure is forward-side, the appropriate response is to stabilize the forward computation (residual stacks, identity-mixing, normalization, edge dropout, jumping connections (Li et al., 2019; Chen et al., 2020; Zhao & Akoglu, 2020; Rong et al., 2020; Xu et al., 2018)); if it is backward-side, the appropriate response is to repair the backward rule itself.

Building on this theoretical reframing, we provide the first operational test that decouples the two failure modes. In a 10-layer GCN trained on Cora, FP32 norms of BP weight gradients at every hidden layer are recorded as zero across all 20 seeds, while the loss still produces output-side error and frozen linear probes on standardized hidden states stay well above chance. The hidden stack has not lost its class signal; BP has lost the ability to convert that signal into trainable updates. Once this diagnosis is accepted, the design question becomes structural rather than corrective: what geometry should a replacement backward rule respect?

Our answer comes from the message-passing Jacobian itself. A single GCN layer has pre-activation Jacobian $W_{\ell+1}^{\top} \otimes \hat{A}$, separating cleanly into a feature-side factor and a topology-side factor; this Kronecker structure is structural to message passing on graphs and absent from MLP feedback alignment. KAFT (Kronecker-Aligned Feedback Training) follows this structure directly: it preserves an explicit graph-side operator while learning a compact feature-side surrogate that is aligned to a chain-normalized probe estimator of the suffix weight chain. KAFT changes only the hidden backward rule; the forward message-passing computation is untouched, and the new rule is computed at constant parallel dependency depth in the network depth L.

Figure 1 brings together the backward-transport diagnosis and KAFT's graph-structured feedback rule.

Contributions.

- **Diagnosis** (Section 2). We provide the first empirical decoupling test confirming the backward-transport hypothesis (Arroyo et al., 2025): BP returns zero hidden updates while standardized hidden states retain probe-recoverable class signal, separating backward failure from forward collapse on the same trained models.
- **Method** (Section 3). We derive the Kronecker factorization of the GCN suffix Jacobian and propose KAFT, the first GNN training rule that learns a Jacobian-aligned feedback operator while explicitly preserving the graph-side transport factor.

- **Empirical results** (Section 4). Across 4 datasets $\times$ 4 backbones $\times$ 2 depths $\times$ 3 learning rates (96 paired comparisons), KAFT improves BP in 86 settings and is BH-significant at $q = 0.05$ in 76; it stacks additively with forward-side stabilizers (KAFT+JKNet, KAFT+ResGCN, KAFT+PairNorm achieve the best result on Cora, DBLP, and CiteSeer respectively); on four large real-world graphs at $L = 20$, KAFT is the unique top-performing method on every dataset (Appendix C).
- **Causal validation** (Section 4.6). A wrong-topology control replacing the backward graph with a rewired, permuted, or Erdős–Rényi alternative collapses KAFT by up to 45 percentage points ($p < 10^{-6}$ on every paired comparison), confirming that consistency between the forward and backward graph—not merely structured backward feedback—is what the topology factor enforces.

2 The Backward Gradient Transport Bottleneck

2.1 Depth Makes Message Passing a Transport Problem

A message-passing GNN expands a node's receptive field by one hop per layer. Depth is therefore the direct way for an MP-GNN to use information from longer graph paths. This need is explicit in prior work: patterns involving longer paths require deeper GNNs, and the Long Range Graph Benchmark (LRGB) was introduced to test tasks requiring long-range interactions between vertices (Lukovnikov & Fischer, 2021; Dwivedi et al., 2022).

The difficulty is that deeper MP-GNNs often fail before this extra receptive field becomes useful. Prior work reports performance drops with increasing depth, exponential loss of node-discriminating power in deep GNNs, and oversmoothing from repeated graph propagation (Lukovnikov & Fischer, 2021; Oono & Suzuki, 2020; Li et al., 2018). Forward-side remedies target different aspects of this collapse: DropEdge reduces effective graph connectivity (Rong et al., 2020), identity mappings preserve initial features (Chen et al., 2020), normalization controls representation scale (Zhao & Akoglu, 2020), and jumping connections retain earlier-layer signals (Xu et al., 2018).

We study sparse-to-moderate, homophilous node-classification graphs because they make the transport question well-isolated. Edges are label-relevant enough that message passing is meaningful, but neighborhoods are not so dense that supervision is effectively broadcast in a few steps. This is the regime where depth should help the forward model.

2.2 Hidden Information and Trainable Signal Can Decouple

When a deep MP-GNN fails, has depth destroyed the task-relevant information in its hidden states, or has backpropagation lost the ability to convert that information into weight updates? The standard explanation is forward-side collapse: repeated graph propagation makes node representations increasingly indistinguishable, so hidden states lose useful task information (Li et al., 2018; Oono & Suzuki, 2020). Recent work has challenged this account by identifying gradient flow as an independent failure mode: the sequential backward product through many layers can shrink gradients toward zero, preventing weight updates even when the forward representations remain informative (Arroyo et al., 2025; Park et al., 2025; Tönshoff et al., 2023). We therefore treat depth failure as a diagnostic question rather than a terminology debate.

A hidden state can contain class-relevant directions while the loss induces numerically ineffective updates to the weights that produced it. We therefore measure both sides on the same trained models. To assess backward-signal scale, we record hidden weight-gradient norms $\|\partial\mathcal{L}/\partial W_\ell\|_F$ and the output-side error. To test whether hidden states retain task information, we train frozen linear probes on standardized hidden states. Finite output error and above-chance hidden probes alongside vanishing recorded weight-gradient norms indicate an optimization bottleneck coexisting with recoverable hidden information.

2.3 Numerically Ineffective Updates to Useful Hidden Layers

We instantiate this test on plain BP-trained GCNs on Cora at depths $L \in \{6, 10, 20\}$, measuring all three quantities on the same trained checkpoints. Figure 1 shows the decoupling. At $L = 6$, BP remains trainable. At $L = 10$ and $L = 20$, every hidden-layer weight-gradient norm is recorded as zero in FP32 across all 20 seeds and test accuracy collapses, yet the loss still produces output-side error and frozen linear probes on standardized hidden states stay well above chance. FP32 norm reductions can underflow even when individual gradient entries remain nonzero, so the zero measurements characterize numerical signal scale.

This is the backward gradient transport bottleneck: useful hidden directions survive while BP's sequential backward product becomes numerically ineffective for updating the parameters that would learn them.

3 KAFT: KRONECKER-ALIGNED FEEDBACK TRAINING

3.1 FROM INEFFECTIVE HIDDEN UPDATES TO STRUCTURED FEEDBACK

Given the diagnosis of Section 2, we change only the source of the hidden-layer feedback signal. Let $E := \partial\mathcal{L}/\partial Z_{L-1} \in \mathbb{R}^{N\times C}$ denote the output-side error. Given any hidden feedback signal δ_ℓ, the local weight update keeps the usual form $\nabla W_\ell = S_\ell^\top \delta_\ell$, where S_ℓ is the graph-aggregated input to layer ℓ:

$$S_\ell = \begin{cases} \hat{A}X, & \ell = 0, \\ \hat{A}H_{\ell-1}, & \ell > 0. \end{cases} \tag{1}$$

BP obtains δ_ℓ by sequentially transporting E through the suffix Jacobian from the output back to layer ℓ. Any replacement for this sequential transport should satisfy three requirements. First, it should produce trainable hidden-layer updates without relying on the same depth-L backward product that vanishes in Figure 1. Second, it should remain directionally related to the suffix Jacobian, rather than using fixed random feedback. Third, because the forward model is a message-passing network, the backward rule should respect the graph structure through which the hidden states were produced.

We propose KAFT (Kronecker-Aligned Feedback Training), which keeps the same forward computation and the same local update form, but replaces BP's sequential hidden feedback with a direct structured feedback operator satisfying all three requirements. The next subsection makes the structural observation that motivates KAFT's design: for a GCN layer, the pre-activation Jacobian factors as $W_{\ell+1}^\top \otimes \hat{A}$, separating feature-side and topology-side transport.

3.2 THE MESSAGE-PASSING JACOBIAN FACTORIZES

The structure missing from an arbitrary feedback rule is already present in the message-passing Jacobian. Consider one GCN transition $Z_{\ell+1} = \hat{A}H_\ell W_{\ell+1}$, where $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ is the symmetrically normalized adjacency matrix with self-loops. Using $\mathrm{vec}(AXB) = (B^\top \otimes A)\mathrm{vec}(X)$, its pre-activation Jacobian is

$$\frac{\partial \mathrm{vec}(Z_{\ell+1})}{\partial \mathrm{vec}(H_\ell)} = W_{\ell+1}^\top \otimes \hat{A}. \tag{2}$$

Thus one backward step separates into two factors: a feature-side transport $W_{\ell+1}^\top$ and a topology-side transport $\hat{A}$. The graph-side factor is the part absent from MLP feedback alignment (Lillicrap et al., 2016; Nøkland, 2016), where the corresponding node dimension is only an identity operator.

This factorization extends exactly through a linear GCN suffix. Let $\sigma = \mathrm{id}$ and define $Q_\ell := W_{L-1}^\top W_{L-2}^\top \cdots W_{\ell+1}^\top \in \mathbb{R}^{C\times d}$. Then the suffix from H_ℓ to the output pre-activation satisfies $Z_{L-1} = \hat{A}^{L-1-\ell} H_\ell Q_\ell^\top$, and therefore

$$\frac{\partial \mathrm{vec}(Z_{L-1})}{\partial \mathrm{vec}(H_\ell)} = Q_\ell \otimes \hat{A}^{L-1-\ell}. \tag{3}$$

The exact BP suffix Jacobian is therefore a Kronecker product of feature transport and graph transport in the linear case.

With nonlinearities, activation gates enter between adjacent Jacobian factors. Writing $D_k := \mathrm{Diag}(\mathrm{vec}(\sigma'(Z_k)))$, the suffix Jacobian becomes

$$\frac{\partial \mathrm{vec}(Z_{L-1})}{\partial \mathrm{vec}(H_\ell)} = (W_{L-1}^\top \otimes \hat{A}) D_{L-2} (W_{L-2}^\top \otimes \hat{A}) \cdots D_{\ell+1} (W_{\ell+1}^\top \otimes \hat{A}). \tag{4}$$

The gates destroy exact Kronecker separability, but they do not remove the repeated graph-side transport. KAFT therefore keeps an explicit topology-side operator and learns a compact feature-side surrogate for the suffix transport.

3.3 THE KAFT FEEDBACK RULE

KAFT replaces BP's hidden-layer feedback with a direct factorized operator. The classifier layer is trained with the true output-side error E. For hidden layers, KAFT first forms a graph-spread error $\bar{E} = D(\hat{A})E$, where $D(\hat{A})$ is a fixed diffusion operator. In semi-supervised node classification, E is nonzero only on labeled nodes, so $D(\hat{A})$ seeds the supervised error to nearby unlabeled nodes. In fully supervised settings, one can take $D(\hat{A}) = I$.

For each hidden layer $\ell \in \{0, \ldots, L-2\}$, KAFT uses a learned feature-side feedback matrix $R_\ell \in \mathbb{R}^{C\times d_\ell}$ and a layer-specific graph-side operator $P_\ell(\hat{A}) = \hat{A}^{r_\ell}$ with $r_\ell = \min(L-1-\ell, K)$.

The hop cap K prevents the backward graph signal from becoming an overly smoothed high-power diffusion, avoids recreating the full depth-L graph transport chain, and enables reuse of a fixed topology stack in the implementation.

Combining the two graph operators into $\mathcal{B}_\ell(\hat{A}) := P_\ell(\hat{A})D(\hat{A})$, the hidden feedback is

$$\delta_\ell^{\mathrm{KAFT}} = \sigma'(Z_\ell) \odot \left[\mathcal{B}_\ell(\hat{A})ER_\ell\right], \qquad \ell = 0, \ldots, L-2. \tag{5}$$

The local gate $\sigma'(Z_\ell)$ only masks inactive units at the target layer. KAFT does not multiply the output error through the full downstream chain of activation gates, which is the depth-dependent product that makes the BP suffix vanish. The hidden-layer update direction keeps the usual local form $U_\ell^{\mathrm{KAFT}} = S_\ell^\top \delta_\ell^{\mathrm{KAFT}}$, and the output layer remains standard. Thus KAFT leaves the forward message-passing computation unchanged and replaces only the source of hidden-layer feedback.

Ignoring the pointwise gate, this rule mirrors the linear BP suffix Jacobian $Q_\ell \otimes \hat{A}^{L-1-\ell}$ from Section 3.2: R_ℓ is a learned feature-side surrogate for Q_ℓ, and $\mathcal{B}_\ell(\hat{A})$ is a capped, semi-supervised graph-side surrogate for the full topology suffix.

3.4 Learning the Feature-Side Operator

KAFT uses R_ℓ as a compact feature-side surrogate for the ideal suffix transport Q_ℓ defined in Section 3.2. The Kronecker view is what makes this low-dimensional: graph-side transport is handled explicitly by $\mathcal{B}_\ell(\hat{A})$, so the learned object is only $C \times d_\ell$, not a full $NC \times Nd_\ell$ feedback matrix.

We do not use fixed random feedback. Instead, KAFT aligns R_ℓ to a chain-normalized probe target derived from the current suffix weights. At an alignment step, we draw m Gaussian probes $\Phi_\ell \in \mathbb{R}^{d_\ell \times m}$, and starting from $Y = \Phi_\ell$ propagate them through the feature-side suffix $Y \leftarrow W_k^\top Y$ for $k = \ell+1, \ldots, L-1$. After each multiplication, we normalize each probe column and rescale by the mean input-probe norm:

$$Y_{:j} \leftarrow \frac{Y_{:j}}{\|Y_{:j}\|_2 + \varepsilon} \cdot \frac{1}{m}\sum_{i=1}^{m} \|\Phi_{\ell,:i}\|_2.$$

The resulting target $\widetilde{Q}_\ell = \frac{1}{m} Y \Phi_\ell^\top$ is not an unbiased estimator of Q_ℓ. It is a scale-normalized directional target. This is intentional: long products of feature matrices can be ill-scaled at depth, while the feedback rule only needs a stable direction for the feature-side operator.

We update R_ℓ by increasing its Frobenius cosine with this target:

$$\mathcal{L}_{\mathrm{align},\ell} = -\frac{\langle R_\ell, \mathrm{sg}(\widetilde{Q}_\ell)\rangle_F}{\|R_\ell\|_F \, \|\widetilde{Q}_\ell\|_F + \varepsilon},$$

where $\mathrm{sg}(\cdot)$ stops gradients through the probe target. After the update, we column-normalize R_ℓ so that the alignment fixes the direction of the feature-side feedback while preventing its norm from becoming another unstable depth-dependent quantity.

3.5 Parallel-Depth Feedback Computation

The factorized form in Eq. (5) also gives an efficient implementation. Rather than backpropagating the output error through the hidden layers one at a time, KAFT precomputes a capped topology stack $T_k = \hat{A}^k \bar{E}$ for $k = 0, \ldots, K$ (only K SpMMs, independent of L), gathers one entry T_{r_ℓ} per layer, and applies all R_ℓ in a single batched multiplication to produce $\delta_\ell^{\mathrm{KAFT}} = \sigma'(Z_\ell) \odot T_{r_\ell} R_\ell$. The hidden updates U_ℓ^{KAFT} then follow as batched matrix products over the stored forward activations. Full pseudocode is in Appendix F.

This makes parallel dependency depth the salient distinction. BP must compute hidden feedback by traversing the suffix chain layer by layer, so the backward dependency length grows with L. KAFT instead builds a fixed-depth topology stack, gathers one entry per layer, and applies all feature-side feedback matrices in batch. The total arithmetic and storage still grow with the number of layers, but the dependency depth of hidden feedback computation is constant for fixed K. Table 1 summarizes the resulting backward-pass structure relative to BP.

The alignment of R_ℓ (Section 3.4) runs separately from this feedback path; the optimized implementation amortizes alignment over $T_{\mathrm{align}} = 10$ steps and is used for the wall-clock measurements in Appendix F.

KAFT makes three empirical predictions: it should (i) improve depth survival under the same forward architecture by bypassing BP's vanishing suffix chain, (ii) produce hidden feedback direc-

Table 1: Backward-pass structure under the same forward GCN. KAFT makes the hidden-feedback dependency length constant in L for a fixed hop cap K, while leaving the forward computation unchanged. Wall-clock measurements are reported in Appendix F.

Backward quantity	BP	KAFT (this work)
Hidden-feedback parallel depth	$\Theta(L)$	$\Theta(K+1)$
Hidden-feedback kernel launches	$\Theta(L)$ sequential	constant, batched
Per-layer gradient kernel launches	$\Theta(L)$ sequential	constant, batched

tionally aligned with BP gradients while those gradients remain measurable, and (iii) collapse under contradictory backward topology because the graph-side operator is tied to the forward graph.

4 Experiments: Depth Survival and Feedback Structure

We test the three predictions of Section 3: depth survival (Sections 4.2–4.3), per-layer cosine alignment with the BP gradient (Appendix E, all five hidden layers strictly positive on Cora GCN $L = 6$), and forward–backward graph consistency (Section 4.6).

4.1 Evaluation Protocol

We evaluate breadth on Cora, CiteSeer, PubMed, and DBLP at $L = 5, 6$ across GCN, GraphSAGE, APPNP, and GIN, and depth survival on larger graphs (CitationFull-CiteSeer/DBLP/PubMed, Coauthor-Physics) across $L \in \{3, \ldots, 20\}$. Unless stated otherwise, methods share the same forward backbone; only the hidden backward rule changes. Significance uses paired t-tests with Benjamini–Hochberg correction at $q = 0.05$; full hyperparameters, splits, and seed counts are in Appendix A.

4.2 Broad Gains at the Onset of Depth Failure

We first evaluate KAFT at $L = 5$ and $L = 6$ on plain message-passing backbones, the operating point where BP is already unstable but accuracy remains high enough for the backward-side intervention to be diagnostic. Table 2 reports the full learning-rate sweep; the largest gains appear at $L = 6$ on Cora and CiteSeer ($+8$ to $+12$ pp on GCN, SAGE, and APPNP).

The structured exception is GIN. Its $(1 + \epsilon)I$ term creates an identity path through every layer, providing a skip-like route for both forward features and backward signal, so KAFT gives smaller gains or occasional losses on GIN. We treat this as an architectural control: when the forward architecture already supplies an identity path, the need for a backward-side repair is reduced.

4.3 Depth and Perturbation Stress Tests

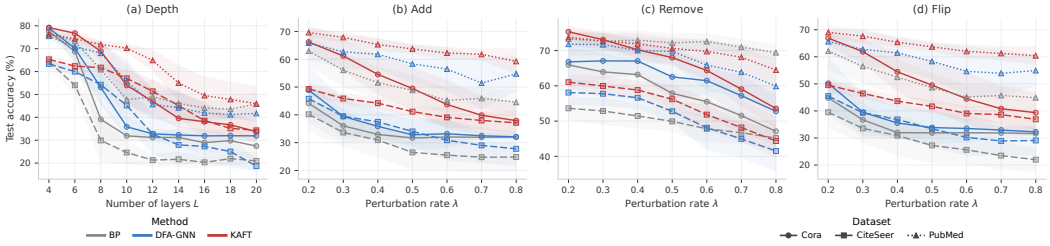


Figure 2: Depth and perturbation stress tests. (a) Depth sweep on standard citation benchmarks. All methods degrade at large depth, but KAFT delays collapse relative to BP and DFA-GNN under the same forward GCN backbone. (b–d) Perturbation sweeps under add, remove, and flip corruptions at rate λ. KAFT generally preserves a positive margin at moderate perturbation rates, while all methods degrade under severe corruption. Curves show mean accuracy with shaded uncertainty; full per-dataset results and paired tests are reported in Appendix D.

We stress the same forward GCN backbone along two axes: larger depths and corrupted graph inputs (Figure 2). KAFT does not eliminate degradation; it delays it. The depth sweep shows KAFT remaining above BP and DFA-GNN through $L = 8$–20, the region where the baselines collapse sharply; the same trend extends to four large real-world graphs at $L = 20$, where KAFT is the unique top-performing method on every dataset (Appendix C). The perturbation panels show KAFT preserving a positive margin at moderate rates under edge-addition and label-flip; at extreme corruption all methods degrade and BP can remain competitive in some removal settings. KAFT is therefore not a general-purpose adversarial defense, but its structured feedback stays stable across the same regimes used to evaluate graph feedback-alignment baselines.

Table 2: $\Delta(\mathrm{KAFT} - \mathrm{BP})$ test accuracy at the onset of depth failure. Each entry reports the accuracy difference in percentage points under the same dataset, backbone, depth, and learning rate. Positive values mean that changing only the hidden backward rule improves BP under the same forward backbone. KAFT improves BP in $86/96$ settings; cells are colored only where the paired t-test is significant after Benjamini–Hochberg correction at $q = 0.05$ (green: KAFT win, red: BP win), giving $76/96$ significant cells. Per-cell mean $\pm$ std and exact p-values are in Appendix B.

		$L = 5$			$L = 6$		
Dataset	Backbone	10^{-3}	$5{\times}10^{-3}$	10^{-2}	10^{-3}	$5{\times}10^{-3}$	10^{-2}
Cora	GCN	**+4.5**	**+5.3**	**+5.5**	**+9.0**	**+10.2**	**+8.0**
	SAGE	**+3.2**	**+4.3**	**+3.8**	**+8.1**	**+7.5**	**+6.5**
	GIN	**+1.5**	**+1.7**	**+0.7**	**+1.8**	**+1.0**	−0.1
	APPNP	**+4.3**	**+4.7**	**+4.4**	**+11.0**	**+10.3**	**+11.9**
CiteSeer	GCN	**+4.6**	**+4.8**	**+3.0**	**+10.4**	**+8.3**	**+7.9**
	SAGE	**+4.2**	**+4.0**	**+3.2**	**+9.2**	**+8.2**	**+5.9**
	GIN	+0.1	-1.4	−0.8	**+1.5**	−0.6	-2.0
	APPNP	**+3.8**	**+3.3**	**+3.2**	**+10.3**	**+10.1**	**+9.3**
PubMed	GCN	**+1.2**	+0.8	−0.4	**+2.6**	**+2.6**	+0.8
	SAGE	+0.8	+0.7	+0.0	**+1.9**	**+2.0**	+0.3
	GIN	**+1.1**	−0.5	+0.0	+0.5	+0.2	-2.0
	APPNP	**+2.2**	−0.4	-1.5	**+4.6**	**+2.2**	+2.1
DBLP	GCN	**+0.9**	**+1.0**	**+1.1**	**+1.6**	**+2.4**	**+1.8**
	SAGE	+0.2	+0.3	**+0.8**	**+0.7**	**+1.6**	**+1.1**
	GIN	**+0.7**	**+0.5**	**+0.8**	**+1.1**	**+0.6**	+0.2
	APPNP	**+1.5**	**+2.4**	**+2.5**	**+4.0**	**+4.9**	**+3.3**

86/96 positive settings; 76/96 significant after BH at $q = 0.05$

Table 3: Non-BP and backward-rule baselines under the same forward GCN at $L = 6$ with 20 seeds. Only the training rule changes. Green: best per column; underline: second best.

Method	Cora	CiteSeer	PubMed	DBLP
BP	68.8±4.6	54.0±4.1	73.2±3.0	80.5±1.0
DFA	70.4±6.8	60.2±2.4	72.4±2.0	81.5±1.2
DFA-GNN	70.1±6.1	60.0±1.8	70.8±2.0	81.0±1.1
PEPITA	31.9±0.0	18.2±0.3	41.6±2.6	47.7±5.5
FF+VN	25.5±8.8	19.0±2.0	39.7±5.0	44.7±0.0
KAFT	**76.7±1.8**	**62.4±1.9**	**74.4±1.6**	**82.1±0.4**

4.4 Non-BP and Backward-Rule Baselines

Table 3 compares KAFT with non-BP and feedback-rule baselines under the same GCN backbone at $L = 6$. PEPITA and FF+VN collapse to near random-class performance on these graphs. DFA and DFA-GNN are stronger, but remain below KAFT on all four datasets.

4.5 Stackability with Forward-Side Methods

Table 4: KAFT combined with forward-side stabilizers on GCN $L = 6$ with 20 seeds. KAFT changes only the hidden backward rule. Green: best per column; underline: second best.

Method	Cora	CiteSeer	DBLP
BP	68.8±4.6	54.0±4.1	80.5±1.0
BP + ResGCN	77.5±1.6	63.0±2.2	82.3±0.4
BP + JKNet	78.2±1.0	**64.4±1.2**	79.9±0.8
BP + PairNorm	69.0±3.2	55.4±3.4	79.0±0.8
BP + DropEdge	74.8±1.8	64.0±1.6	81.6±0.5
KAFT	76.7±1.8	62.4±1.9	82.1±0.4
KAFT + ResGCN	77.8±1.9	61.5±2.2	**82.7±0.6**
KAFT + JKNet	**78.3±1.6**	61.8±2.2	82.4±0.4
KAFT + PairNorm	75.8±1.5	64.3±2.0	80.7±0.6
KAFT + DropEdge	70.8±3.8	62.1±1.8	80.7±0.7

KAFT modifies only the hidden backward rule, so it can be combined with forward-side stabilizers. Table 4 shows the direct composition test on GCN $L = 6$. KAFT+JKNet gives the best Cora

result, KAFT+ResGCN gives the best DBLP result, and KAFT+PairNorm nearly matches the best CiteSeer result. The exception is DropEdge: stochastic edge removal changes the graph used by message passing and conflicts with KAFT's fixed backward graph operator, foreshadowing the graph-consistency test in Section 4.6.

4.6 Forward–Backward Topology Consistency

Section 3 predicts that supplying KAFT with a backward topology that disagrees with the forward graph should corrupt training, since $\mathcal{B}_\ell(\hat{A})$ is built from the same $\hat{A}$ that produced the forward representations. We test this directly. The forward GCN keeps the true graph $\hat{A}$; only the backward operator $P_\ell(\hat{A})$ is rebuilt from one of three alternatives: a degree-matched random rewiring $\tilde{A}$ generated by the configuration model, a node-index permutation $\Pi\hat{A}\Pi^\top$, and an Erdős–Rényi graph of matched edge density. All three preserve coarse statistics but break per-node correspondence with the forward graph. The results are shown in Table 5.

Table 5: Forward–backward topology consistency on GCN $L = 6$ with 20 seeds. The forward pass uses the true $\hat{A}$; only the backward operator $P_\ell(\hat{A})$ is replaced. Every contradictory backward graph collapses KAFT by up to 45 percentage points, with $p < 10^{-6}$ on every paired comparison against correct $\hat{A}$.

Backward graph	Cora	CiteSeer	DBLP
KAFT (correct $\hat{A}$)	$\mathbf{77.2}\pm\mathbf{1.3}$	$\mathbf{62.7}\pm\mathbf{1.6}$	$\mathbf{81.9}\pm\mathbf{0.8}$
Rewired ($\tilde{A}$)	32.3 ± 1.3	29.6 ± 8.0	46.1 ± 5.1
Permuted ($\Pi\hat{A}\Pi^\top$)	32.5 ± 2.0	48.1 ± 6.5	75.8 ± 3.9
Erdős–Rényi	31.9 ± 0.0	27.4 ± 5.8	44.8 ± 0.3

The collapse is uniform across datasets and graph variants, confirming the third prediction of Section 3: KAFT's gain depends on the backward operator being the same graph that produced the forward representations, not on simply having any structured backward operator. This consistency requirement also explains the single forward stack failure in Section 4.5: KAFT+DropEdge pairs a stochastic forward graph with a fixed backward $\hat{A}$, breaking the same correspondence shown to be necessary here.

5 Related Work

Deep MP-GNN failure and gradient-transport diagnosis. Multi-hop reasoning calls for deeper GNNs, and benchmarks such as LRGB make this need explicit (Lukovnikov & Fischer, 2021; Dwivedi et al., 2022). Yet depth is hard to train. The dominant explanation from 2018 onward has been forward-side oversmoothing: repeated graph propagation drives representations toward an indistinguishable subspace (Li et al., 2018; Oono & Suzuki, 2020). Gradient-aware architectural tricks appeared early—DeepGCNs (Li et al., 2019) borrowed residual connections from CNNs, and GraphCON (Rusch et al., 2022) proved gradient stability for oscillatory GNNs—but neither diagnosed vanishing gradients as a GNN-specific structural failure; they treated it as a known deep-learning problem requiring known solutions. Zhang et al. (2022) explicitly argued that gradient vanishing is *not* the leading cause of deep GNN failure, attributing performance degradation to model degradation from transformation depth.

The reframing arrived in 2024–2025. Park & Kim (2024) introduced "gradient oversmoothing" in the backward pass as an independent pathology distinct from forward collapse. Peng et al. (2024) argued that over-smoothing is not the dominant problem and that trainability is the core issue. Arroyo et al. (2025) unified over-smoothing and over-squashing as consequences of GNN-specific spectral contraction that also kills gradients, noting the backward problem had been "surprisingly mostly overlooked." Park et al. (2025) further argued that prior studies confused over-smoothing with vanishing gradients caused by transformation and activation rather than aggregation. Our work builds on this reframing by providing the first operational decoupling test—not theoretical bounds but zero-gradient observations alongside probe-recoverable hidden signal (Section 2)—and proposing KAFT (Section 3) as a structural backward-rule fix derived from the message-passing Jacobian factorization.

Forward-side remedies. Forward-side remedies change the message-passing computation: residual stacks (Li et al., 2019), identity-mixing (Chen et al., 2020), propagation-step normalization (Zhao & Akoglu, 2020), edge dropout (Rong et al., 2020), and jumping connections (Xu et al., 2018). KAFT is orthogonal: it leaves the forward backbone intact and replaces only the hidden backward rule, so its gains can compose with these forward fixes.

Non-BP credit assignment and graph-aware feedback. Replacing or augmenting BP with alternative credit-assignment rules is a long-standing direction. Feedback alignment uses fixed random feedback matrices (Lillicrap et al., 2016); direct feedback alignment routes the output error to every hidden layer in one step (Nøkland, 2016); sign-concordant feedback removes the magnitude requirement (Xiao et al., 2019); difference target propagation routes targets rather than gradients (Lee et al., 2015); equilibrium propagation derives updates from energy minimization (Scellier & Bengio, 2017); synthetic gradients learn local predictors of the true gradient (Jaderberg et al., 2017); and recent forward-only or backward-pass-free schemes use input-modulation, contrastive-forward, or broadcast-error objectives (Hinton, 2022; Dellaferrera & Kreiman, 2022; Clark et al., 2021). DFA-GNN injects a graph-aware diffusion into the per-layer feedback while keeping the feature-side matrix random (Zhao et al., 2024). GrAPE uses BP anchors to refine random feedback matrices for better Jacobian alignment in MLPs (Caillon et al., 2026). KAFT instead derives the backward rule from the suffix Jacobian's structure: an explicit graph-side operator tied to the forward graph (Section 4.6 shows breaking this consistency collapses the method by up to 45 pp), a chain-normalized probe target that stabilizes alignment without requiring a BP anchor, and a parallel-depth implementation whose backward dependency is constant in L.

6 Discussion and Conclusion

We reframe deep MP-GNN failure as an operational backward-transport problem and follow the Kronecker factorization of the message-passing Jacobian to design a replacement backward rule. KAFT learns feature-side feedback while preserving graph-consistent backward transport, improves BP broadly across depth-stressed node-classification settings, beats random-feedback baselines, delays collapse under stress, and passes the alignment and topology-consistency checks.

Limitations. KAFT relies on edges being meaningful supervision routes. When edges connect nodes with incompatible labels (heterophily) or when neighborhoods are dense enough that supervision is broadcast in a few hops, the backward graph factor stops being the limiting failure mode and KAFT does not help. The current design also targets node-level prediction; graph-level pooling would change the structure of the suffix Jacobian and require a corresponding adjustment. Finally, KAFT addresses only the backward pass: forward over-smoothing and instability remain orthogonal problems for which the forward-side stabilizers we stack with continue to be necessary.

Broader perspective. The KAFT principle—preserve the part of the Jacobian that encodes the architectural prior, learn only the weight-dependent part—is not tied to GCNs or to feedback alignment specifically. It applies wherever the pre-activation Jacobian admits a structural factorization: convolutions with spatial transport, attention with mask-induced routing, and message-passing variants beyond GCN all share the same pattern. The same view also positions KAFT within the longer line of credit-assignment alternatives that abandon BP's exact gradient in favor of more locally implementable and biologically plausible rules—random and direct feedback alignment (Lillicrap et al., 2016; Nøkland, 2016), sign-concordant feedback (Xiao et al., 2019), equilibrium propagation (Scellier & Bengio, 2017), and predictive coding (Whittington & Bogacz, 2017)—by showing that, when the architecture provides exploitable structure, the surrogate feedback need not be random and need not require symmetric weight transport.

Outlook. Several follow-up directions are immediate. Heterophilous graphs invite signed or directional graph-side operators; dense graphs invite identity-mixed kernels $(1-\beta)\hat{A}^k+\beta I$ that interpolate between local routing and broadcasting; graph-level prediction invites pool-aware suffix Jacobians. Beyond GNNs, architectures whose Jacobians admit structural factorization—convolutional models on grid data, attention with structured masks—suggest analogous parallel-depth backward rules. Similar training approaches, such as energy-based GNNs, are also worth exploring. On the theoretical side, characterizing when chain-normalized multi-probe estimators are sharp surrogates for the suffix weight chain, and convergence guarantees for the amortized alignment loop, would tighten the design. The broader implication is that deep GNN design should not only stabilize the forward operator, but also choose a backward geometry that respects it.

AI use statement

In this work, we used generative AI tools (Claude Code and Codex, as coding agents) for writing and debugging experiment code, orchestrating and monitoring training runs. All experimental designs, claims and conclusions are the authors' own; every number reported was produced by the released code and checked by the authors. We have reviewed all AI-assisted work and take responsibility for

the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REFERENCES

Alvaro Arroyo, Alessio Gravina, Benjamin Gutteridge, Federico Barbero, Claudio Gallicchio, Xiaowen Dong, Michael M. Bronstein, and Pierre Vandergheynst. On vanishing gradients, over-smoothing, and over-squashing in GNNs: Bridging recurrent and graph learning. *arXiv preprint arXiv:2502.10818*, 2025.

Paul Caillon, Erwan Fagnou, Blaise Delattre, and Alexandre Allauzen. Scaling direct feedback learning with Jacobian alignment guarantees. In *International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=kasbbmwk3s.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, 2020.

David G. Clark, L. F. Abbott, and SueYeon Chung. Credit assignment through broadcasting a global error vector. In *Advances in Neural Information Processing Systems*, 2021.

Giorgia Dellaferrera and Gabriel Kreiman. Error-driven input modulation: Solving the credit assignment problem without a backward pass. In *International Conference on Machine Learning*, 2022.

Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Alipanah Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long range graph benchmark. In *Advances in Neural Information Processing Systems*, 2022. arXiv:2206.08164.

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

Geoffrey Hinton. The forward-forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

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*, 2020.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *International Conference on Machine Learning*, 2017.

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

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *ECML/PKDD*, 2015. doi: 10.1007/978-3-319-23528-8_31.

Guohao Li, Matthias Müller, Ali Thabet, and Bernard Ghanem. DeepGCNs: Can GCNs go as deep as CNNs? In *IEEE International Conference on Computer Vision*, 2019. doi: 10.1109/ICCV.2019.00936.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence*, 2018. doi: 10.1609/aaai.v32i1.11604.

Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7(13276), 2016. doi: 10.1038/ncomms13276.

Denis Lukovnikov and Asja Fischer. Improving breadth-wise backpropagation in graph neural networks helps learning long-range dependencies. In *International Conference on Machine Learning*, 2021.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *Advances in Neural Information Processing Systems*, 2016.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

MoonJeong Park and Dongwoo Kim. Taming gradient oversmoothing and expansion in graph neural networks. *arXiv preprint arXiv:2410.04824*, 2024. doi: 10.48550/arXiv.2410.04824.

MoonJeong Park, Sunghyun Choi, Jaeseung Heo, Eunhyeok Park, and Dongwoo Kim. The oversmoothing fallacy: A misguided narrative in GNN research. *arXiv preprint arXiv:2506.04653*, 2025.

Jie Peng, Runlin Lei, and Zhewei Wei. Beyond over-smoothing: Uncovering the trainability challenges in deep graph neural networks. In *International Conference on Information and Knowledge Management*, 2024. doi: 10.1145/3627673.3679776.

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*, 2023.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2020.

T. Konstantin Rusch, Benjamin Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-coupled oscillator networks. In *International Conference on Machine Learning*, 2022.

Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, 11, 2017. doi: 10.3389/fncom.2017.00024.

Ganlin Song, Ruitu Xu, and John Lafferty. Convergence and alignment of gradient descent with random backpropagation weights. In *Advances in Neural Information Processing Systems*, 2021.

Jan Tönshoff, Martin Ritzert, Eran Rosenbluth, and Martin Grohe. Where did the gap go? reassessing the long-range graph benchmark. *Transactions on Machine Learning Research*, 2023.

James C. R. Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural Computation*, 2017. doi: 10.1162/NECO_a_00949.

Will Xiao, Honglin Chen, Qianli Liao, and Tomaso Poggio. Biologically-plausible learning algorithms can scale to large datasets. In *International Conference on Learning Representations*, 2019.

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*, 2018.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. GraphSAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020.

Wentao Zhang, Zeang Sheng, Ziqi Yin, Yuezihan Jiang, Yikuan Xia, Jun Gao, Zhi Yang, and Bin Cui. Model degradation hinders deep graph neural networks. 2022. doi: 10.1145/3534678.3539374.

Gongpei Zhao, Tao Wang, Congyan Lang, Yi Jin, Yidong Li, and Haibin Ling. DFA-GNN: Forward learning of graph neural networks by direct feedback alignment. In *Advances in Neural Information Processing Systems*, 2024.

Lingxiao Zhao and Leman Akoglu. PairNorm: Tackling oversmoothing in GNNs. In *International Conference on Learning Representations*, 2020.

A Experimental Details

Datasets. We use four standard citation graphs (Cora, CiteSeer, PubMed, DBLP) for the main BP-vs-KAFT comparison and four larger graphs for the depth-survival experiments (CitationFull-CiteSeer, CitationFull-DBLP, CitationFull-PubMed, Coauthor-Physics). Sizes range from 2.7K to 34.5K nodes; average degree from 2.5 to 14.4; all are homophilous in the regime targeted by our diagnosis.

Splits. Planetoid-style 5% per-class semi-supervised split with the standard validation/test partition. The split is fixed across seeds; only model initialization and training shuffle vary.

Training. Adam optimizer with weight decay 5×10^{-4}, 200 epochs, no learning-rate scheduler, no residual connections, no BatchNorm, no Dropout. Hidden dimension 64. We deliberately stress the bare backbone to isolate the backward-transport effect; results with forward-side stabilizers are reported in the stackability table of Section 4.5. The learning rate is swept over $\{10^{-3}, 5 \times 10^{-3}, 10^{-2}\}$ per setting, and we report the best-LR cell in the main tables and all three LR cells in Appendix B.

KAFT hyperparameters. 64 Gaussian probes per alignment step, alignment performed every 10 training steps, hop cap $K = 3$. The diffusion operator $D(\hat{A})$ is 10 iterations of label spreading with $\alpha = 0.5$. Sensitivity to these choices is characterized in Appendix G; variation is $\leq 3\%$ across the tested ranges.

Statistical procedure. 20 random seeds (independent model initialization and training shuffle). We report mean ± standard deviation on test accuracy at the validation peak. Significance testing uses paired t-tests on seed-matched accuracy vectors. To control for multiple comparisons across 144 paired tests in the paper, we apply Benjamini–Hochberg correction at $q = 0.05$; 117 of 144 tests survive. Settings reported as significant survive both unadjusted ($p < 0.05$) and BH-adjusted thresholds.

B Full BP-vs-KAFT Sweep

Table 6 reports the unaggregated 32-cell BP-vs-KAFT comparison underlying Section 4.2: 4 datasets $\times$ 4 backbones $\times$ $\{L = 5, 6\}$, with 20 seeds per cell. Each row uses the best LR per (dataset, backbone, depth) configuration. Green entries mark the better method per row when the paired t-test is significant at $q = 0.05$ after BH correction; ties are unmarked.

KAFT improves BP in 86 of 96 paired settings across the full LR sweep, and is the better-or-tied method in 30 of 32 best-LR cells. The two cells where BP is better are GIN $L = 5$ on CiteSeer and GIN $L = 6$ on CiteSeer, consistent with the architectural control discussed in Section 4.2: GIN's $(1 + \epsilon)I$ identity already provides a residual gradient path.

Table 6: Full BP-vs-KAFT comparison at best LR per (dataset, backbone, depth). 20 seeds, paired t-test, BH corrected at $q = 0.05$. Green marks the better method per row when significant.

Dataset	Setting	BP	KAFT	Δ	p
Cora	GCN $L = 5$	74.3±2.5	78.8±1.0	+4.5	<0.001
	GCN $L = 6$	69.4±5.7	78.2±1.1	+8.7	0.002
	SAGE $L = 5$	74.4±2.8	77.9±0.9	+3.5	<0.001
	SAGE $L = 6$	69.5±4.9	78.4±0.9	+8.9	<0.001
	APPNP $L = 5$	74.8±2.7	79.1±1.1	+4.3	<0.001
	APPNP $L = 6$	66.4±5.0	77.8±2.9	+11.4	<0.001
	GIN $L = 5$	78.5±1.3	80.1±1.0	+1.6	<0.001
	GIN $L = 6$	77.8±1.5	77.8±1.5	+0.0	ns
CiteSeer	GCN $L = 5$	60.6±3.1	63.7±1.8	+3.1	0.002
	GCN $L = 6$	55.7±3.6	63.5±2.2	+7.7	<0.001
	SAGE $L = 5$	61.2±3.2	63.9±1.8	+2.8	0.005
	SAGE $L = 6$	55.8±4.8	62.0±2.1	+6.2	0.007
	APPNP $L = 5$	61.3±2.7	64.6±1.6	+3.2	<0.001
	APPNP $L = 6$	53.3±5.4	64.7±1.7	+11.4	<0.001
	GIN $L = 5$	66.7±1.3	65.2±1.3	−1.5	<0.001
	GIN $L = 6$	65.1±1.7	63.1±2.3	−2.1	0.004
PubMed	GCN $L = 5$	75.8±2.1	76.9±0.7	+1.2	0.032
	GCN $L = 6$	73.2±2.7	75.8±1.1	+2.6	<0.001
	SAGE $L = 5$	75.8±1.8	76.6±0.4	+0.8	ns
	SAGE $L = 6$	74.5±1.8	76.5±1.0	+2.0	0.001
	APPNP $L = 5$	76.9±1.8	79.1±0.4	+2.2	<0.001
	APPNP $L = 6$	73.7±3.7	78.3±0.9	+4.6	<0.001
	GIN $L = 5$	76.6±0.7	77.7±0.6	+1.1	<0.001
	GIN $L = 6$	76.4±1.3	76.9±1.0	+0.5	ns
DBLP	GCN $L = 5$	82.1±0.4	83.1±0.3	+0.9	<0.001
	GCN $L = 6$	81.3±0.5	82.9±0.3	+1.5	<0.001
	SAGE $L = 5$	82.4±0.3	82.5±0.4	+0.2	ns
	SAGE $L = 6$	81.7±0.5	82.5±0.3	+0.8	0.002
	APPNP $L = 5$	81.6±0.4	83.1±0.4	+1.5	<0.001
	APPNP $L = 6$	79.6±1.2	83.2±0.4	+3.6	<0.001
	GIN $L = 5$	81.8±0.4	82.3±0.4	+0.5	0.001
	GIN $L = 6$	81.6±0.6	82.2±0.5	+0.6	0.004

C Real-World Deep Graphs

The main-text experiments stress the standard Planetoid-style benchmarks. Here we extend the depth comparison to four larger real-world graphs that exercise the same regime at scale: CitationFull-CiteSeer (4.2K nodes, 6 classes), CitationFull-DBLP (17.7K, 4 classes), CitationFull-PubMed (19.7K biomedical, 3 classes), and Coauthor-Physics (34.5K, 5 classes). All four are sparse-to-moderate homophilous node-classification graphs with average degree 2.5–14.4.

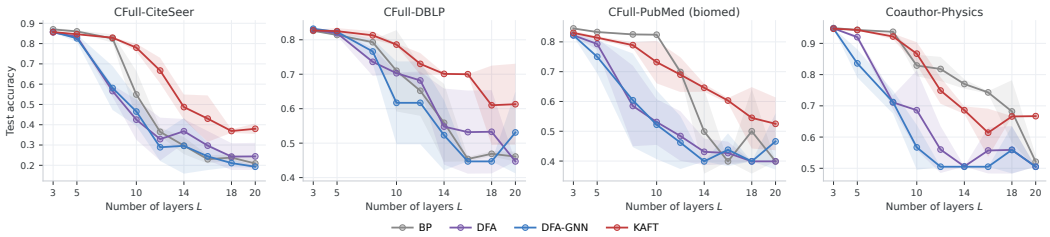


Figure 3: Depth survival on four large real-world graphs. $L \in \{3, 5, 8, 10, 12, 14, 16, 18, 20\}$, 3 seeds per cell. At shallow depth ($L = 3$) all methods are tied above 0.83. From $L \geq 10$, BP, DFA, and DFA-GNN collapse on every dataset; KAFT descends gracefully and remains the leading method at every deeper depth.

To pin down the deep-network gap with full statistical resolution, Table 7 reports a 20-seed comparison at $L = 20$. KAFT is the unique top-performing method on every dataset, with statistically significant gains on CitationFull-CiteSeer ($p = 5 \times 10^{-6}$) and CitationFull-PubMed ($p = 0.002$). The DFA and DFA-GNN baselines exhibit zero or near-zero variance on three datasets, indicating deterministic majority-class collapse: their backward signal is too weak to escape the trivial classifier at this depth.

Table 7: Real-world depth-survival hero at $L = 20$, 20 seeds, paired t-test vs. BP. Bold marks the unique top method per row.

Dataset	BP	DFA	DFA-GNN	KAFT	p (vs. BP)
CFull-CiteSeer	25.3±1.3	21.2±4.2	19.6±0.4	**37.1±8.1**	5×10^{-6}
CFull-DBLP	54.6±2.8	44.7±0.0	44.7±0.0	**57.3±12.0**	0.34
CFull-PubMed (biomed)	41.9±1.3	40.0±0.3	39.9±0.0	**49.9±9.6**	0.002
Coauthor-Physics	58.5±15.5	50.6±0.2	50.5±0.0	**65.4±5.1**	0.07

D Stress Tests

D.1 Perturbation Sweep

The four-panel figure in Section 4.3 (Figure 2) summarizes a perturbation sweep over three attacks (edge add/remove, label flip), three datasets (Cora, CiteSeer, PubMed), three methods (BP, DFA-GNN, KAFT), and seven rates $\lambda \in \{0, 0.1, 0.2, 0.3, 0.5, 0.7\}$, with 20 seeds per cell. KAFT preserves a positive margin over BP across the moderate-rate region ($\lambda \leq 0.2$) for all three attack types. As a representative paired t-test slice on CiteSeer edge-rewire: at $\lambda = 0$, BP 53.8 vs. KAFT 62.6 ($p = 2.6 \times 10^{-8}$); at $\lambda = 0.1$, 36.4 vs. 42.7 ($p = 10^{-4}$). The seed-matched gain at moderate perturbation is therefore not a single-seed artifact.

D.2 BP-vs-KAFT Depth Stress

To complement the depth-sweep figure in Section 4.3, Table 8 reports paired BP-vs-KAFT cells on Cora and DBLP at depths $L \in \{6, 8, 10, 12, 16, 20\}$ under the bare GCN backbone (3 seeds per cell, no forward-side stabilizers).

Table 8: Bare-GCN depth stress, BP vs. KAFT. KAFT delays collapse on every depth across both datasets; the largest absolute gains appear in the $L = 8$–16 range where BP has begun to fail but the task has not yet become unrecoverable.

Dataset	L	BP	KAFT	Δ
Cora	6	71.4±1.1	76.4±2.1	+5.0
	8	39.7±5.3	63.8±5.0	+24.1
	10	35.1±4.4	54.5±4.7	+19.4
	12	32.8±1.9	45.7±1.8	+12.9
	16	29.3±2.2	35.4±2.6	+6.1
	20	24.3±6.7	38.3±5.0	+14.0
DBLP	6	79.9±0.9	82.6±0.5	+2.7
	8	78.8±1.0	82.2±0.4	+3.4
	10	71.1±11.9	78.1±1.0	+7.0
	12	66.8±6.4	73.4±3.2	+6.6
	16	45.4±0.7	69.9±0.1	+24.5
	20	46.1±1.4	61.8±11.0	+15.7

E Alignment Diagnostics

Section 3 predicts that KAFT's learned feature-side feedback should be directionally aligned with the true BP gradient when BP gradients are still measurable. We measure this alignment directly on a trained GCN with $L = 6$ on Cora, recording the per-layer cosine $\cos(\delta_\ell^{\mathrm{KAFT}}, \nabla_\ell^{\mathrm{BP}})$ over 200 epochs and 20 seeds.

Table 9: Per-layer cosine alignment between the KAFT hidden feedback and the BP hidden gradient on a trained Cora GCN $L = 6$, 20 seeds. All five hidden layers show strictly positive alignment with 95% CI above zero throughout training.

Layer	$\ell = 0$	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$
$\cos(\delta^{\mathrm{KAFT}}, \nabla^{\mathrm{BP}})$	0.33 ± 0.12	0.36 ± 0.15	0.39 ± 0.16	0.42 ± 0.16	0.59 ± 0.19

Cosine is highest near the loss ($\ell = 4$) and decays smoothly toward the input, consistent with the multi-probe estimator becoming noisier as more weight matrices chain into Q_ℓ. Alignment improves over training as R_ℓ converges, showing that the feedback is a meaningful BP-direction approximation rather than an arbitrarily-shaped vector.

Gradient reach at depth. The same diagnostic at $L = 10$ makes the underlying transport problem visible. Across all 20 seeds and all hidden layers, BP gradient norms $\|\partial\mathcal{L}/\partial Z_\ell\|_F < 10^{-38}$ (i.e., underflow IEEE-754 single precision), while the forward representations remain $\Theta(1)$-bounded. KAFT's hidden feedback at the same depth has $\|\delta_\ell^{\mathrm{KAFT}}\|_F \approx 0.7$–$1.2$ across all layers with tight 95% CI. The accuracy gap at $L = 10$ is correspondingly large: GCN $\Delta = +16.3\%$ ($p = 4\times10^{-4}$) and APPNP $\Delta = +10.8\%$ ($p = 8\times10^{-3}$). At $L = 6$, BP norms are ~ 0.02 while KAFT feedback is ~ 0.17 (about $8\times$ larger), consistent with BP being marginally functional rather than completely broken at this depth.

F Efficiency and Implementation Parity

F.1 Wall-Clock per Training Step

Table 10 compares the wall-clock per training step of BP, BP+ResGCN, and the optimized KAFT implementation (Section 3.5) on Cora and DBLP at $L = 6$ and $L = 10$. All timings are run on a single NVIDIA A6000 (48 GB), reported as the median of 5 timing runs in milliseconds.

Table 10: Wall-clock per training step (ms; median of 5 runs). Speedup is BP/KAFT-Opt. Green marks the fastest method per row.

Dataset	L	BP	BP+ResGCN	KAFT-Opt	Speedup vs. BP
Cora	6	4.16	4.80	2.62	$1.59\times$
Cora	10	7.03	6.40	4.07	$1.73\times$
DBLP	6	5.51	5.35	5.34	$1.03\times$
DBLP	10	7.13	7.42	7.33	$0.97\times$

The Cora speedup is driven by avoiding autograd overhead and replacing the L-step sequential backward chain with a constant number of batched kernel launches. On DBLP the speedup vanishes because individual SpMMs are large enough to saturate the GPU and amortize the autograd overhead. Memory overhead is modest (1.2–$1.4\times$ peak), driven by the topology stack $\{T_k\}_{k=0}^K$ and the per-layer feedback matrices $\{R_\ell\}$.

F.2 Full Parallel-Depth Implementation

The compressed description in Section 3.5 is unpacked here. KAFT computes all hidden feedback signals from a shared topology stack and a batched feature-side multiplication.

First, after computing the output-side error E, we form the graph-spread error

$$\bar{E} = D(\hat{A})E.$$

We then build a capped topology stack

$$T_k = \hat{A}^k \bar{E}, \qquad k = 0, \ldots, K.$$

Since K is fixed, this requires only K sparse matrix multiplications, independent of the network depth L. For each hidden layer ℓ, KAFT selects T_{r_ℓ} with $r_\ell = \min(L-1-\ell, K)$, which corresponds to applying the graph-side operator $P_\ell(\hat{A})$ to $\bar{E}$.

Second, all layerwise feature feedbacks are computed as a batched multiplication $G_\ell = T_{r_\ell} R_\ell$ for $\ell = 0, \ldots, L-2$. Stacking the G_ℓ's gives a tensor in $\mathbb{R}^{(L-1)\times N\times d}$. The local activation gates are then

applied layerwise, $\delta_\ell^{\mathrm{KAFT}} = \sigma'(Z_\ell) \odot G_\ell$. Finally, the hidden update directions $U_\ell^{\mathrm{KAFT}} = S_\ell^\top \delta_\ell^{\mathrm{KAFT}}$ are computed with batched dense multiplications and sparse graph aggregations over the stored forward activations.

The total number of GPU kernel launches in the backward phase is therefore constant in L, in contrast to BP's L-step sequential dependency. Total arithmetic still grows with L inside the wide batched multiplication, but the GPU executes it as a small number of fused operations.

F.3 Reference vs. Optimized Implementation

The optimized KAFT implementation amortizes the alignment step every 10 training steps (instead of every step) and reuses a precomputed topology stack. Table 11 confirms that this amortization preserves accuracy.

Table 11: Optimized KAFT accuracy at $L = 6$, 5 seeds. All nine settings are within $\pm 2\%$ of the reference implementation; no setting is significantly different at $p < 0.05$.

Backbone	Cora	CiteSeer	DBLP
GCN	76.9±2.2	61.6±2.7	82.5±0.3
SAGE	75.6±1.1	61.5±2.1	82.2±0.4
APPNP	76.1±1.7	59.4±1.7	82.8±0.3

G Hyperparameter Sensitivity

KAFT introduces three method-specific hyperparameters: the number of Gaussian probes m used to estimate the alignment target, the alignment frequency T_{align}, and the hop cap K on the backward graph operator. Table 12 sweeps each axis with the others held at their default values on Cora GCN $L = 6$ (3 seeds; defaults bold).

Table 12: Hyperparameter sensitivity on Cora GCN $L = 6$, 3 seeds. Variation is $\leq 3\%$ across each tested range; defaults sit at or near the optimum on every axis.

(a) Probe count m (alignment every 10 steps, $K = 3$)					
m	16	32	**64**	128	256
Acc (%)	74.6±0.8	76.1±1.1	**77.5±1.6**	77.5±1.2	77.1±3.5
(b) Alignment frequency T_{align} ($m = 64$, $K = 3$)					
T_{align}	1	5	**10**	20	50
Acc (%)	77.1±1.8	76.9±0.3	**78.2±0.9**	71.9±4.9	73.0±3.2
(c) Hop cap K ($m = 64$, alignment every 10 steps)					
K	1	2	**3**	5	—
Acc (%)	77.1±1.9	76.0±0.9	**78.3±0.6**	78.2±0.7	—

The alignment-frequency axis is the only one with a clear cliff: pushing $T_{\mathrm{align}} \geq 20$ lets R_ℓ drift too far from the chain-normalized target between updates. The probe count and hop cap are both flat over the practical range, suggesting that the method is not finely tuned to either choice.

H Evaluation Protocols for the Additional Experiments

The additional experiments in Appendices I–P use separate selection and evaluation seeds, so their uncertainty estimates should not be compared across tables as if they came from one experiment. Table 13 lists the protocol of each study. Fixed-hyperparameter diagnostics and ablations use 20 seeds. Held-out learning-rate (LR) studies reserve seed 0 for validation-only LR selection, separately for each method, and report paired seeds 1–19. The deep large-graph study uses the same held-out design with paired seeds 1–9 because of its cost. Three-seed results are preliminary and are not used as core accuracy evidence. Unless stated otherwise, BP and KAFT share the identical bias-free forward model, split, width 64, optimizer, training budget, and paired initialization.

I Precision Audit of the BP Update-Null Diagnostic

Section 2 reports that hidden-layer weight-gradient norms of a 10-layer GCN on Cora are recorded as zero. This appendix makes precise what that measurement means, extends it to further datasets and backbones, and locates the depth at which the same endpoint appears on larger graphs.

FP32 norms versus elementwise gradients. The recorded quantity is the Frobenius norm $\|\partial \mathcal{L}/\partial W_\ell\|_F$ computed in native FP32. When gradient entries are of order **??**, their squares underflow during the FP32 reduction even though the entries themselves remain representable and nonzero. We therefore distinguish three quantities: the native FP32 norm, the FP64-accumulated norm, and the

Table 13: Protocol legend for the additional experiments. "Sel." is the seed used only for LR selection; "Eval." are the seeds that produce every reported mean, standard deviation, win count, and p-value.

Study	Sel.	Eval.	LR	Appendix
Update-null precision audit	—	0–19	fixed	I
BP transport horizon (larger graphs)	—	3–5 seeds	fixed	I
Deep crossover and training dynamics (DBLP, Physics)	0	1–9	per method	J
Forward normalization (BN/LN)	0	1–19	per method	K
Identity-rich forwards (GIN, GCNII-style)	0	1–19	per method	L
$D(\hat{A})/P_\ell(\hat{A})$ factorial ablation	—	0–19	fixed (10^{-2})	M
Minesweeper (heterophily)	split 0	splits 1–9	per method	N
Peak training memory	—	0–2	—	O
Link prediction, graph classification	—	0–2	fixed	P

number of FP32 parameter elements that actually change when the supervision gradient is applied. Re-running the Cora GCN $L = 10$, seed-0 configuration gives, for the output layer, nonzero elements in $74{,}740/173{,}312$ entries of its input H_{L-2}, $79{,}753/173{,}312$ entries of the graph-propagated input S_{L-1}, and $343/448$ entries of the weight gradient, while all three native FP32 norms are 0. Recomputing the output-layer gradient manually as $S_{L-1}^\top E$ matches autograd to within **??**. The gradient is therefore not elementwise zero, but it is numerically incapable of updating the parameters: applying it changes zero FP32 parameter elements. The precise diagnostic claim is:

> *Class-relative hidden structure remains recoverable, while ordinary BP produces supervision gradients that are numerically incapable of updating the FP32 parameters.*

The output-adjacent weight gradient is small partly because the forward activation magnitude is strongly attenuated near the output. The backward preactivation error is, however, also attenuated: on the same run its Frobenius norm decays from 7.8×10^{-2} at the output layer to 2.6×10^{-20} at layer 2 and underflows to zero below it, more than 18 orders of magnitude through the backward chain. Near the output the small activation controls the weight-gradient scale; toward earlier layers the sequentially attenuated backward error is decisive. Both factors are present in the deep network, and the standardized hidden probe shows that the forward attenuation has not removed class-relative structure.

Breadth across datasets and backbones. Table 14 repeats the audit for plain 10-layer GCN and GraphSAGE on Cora, CiteSeer, and PubMed (100 epochs, 20 seeds). In all 120 runs every native FP32 weight-gradient norm is zero and the supervision update changes 0% of FP32 parameter elements, including after an Adam-style first-step preconditioning; FP64 accumulation recovers residual entries of order **??**. The output-adjacent preactivation error remains finite, and standardized penultimate-layer probes are far above the seven-, six-, and three-class chance levels.

Table 14: Update-null audit at $L = 10$, 20 seeds per row. "Update-null" counts seeds in which all ten native FP32 weight-gradient norms are zero and the supervision update changes no FP32 parameter element. The probe is a frozen linear probe on the standardized penultimate hidden state.

Dataset	Backbone	Update-null	Max $\lvert\text{grad}\rvert$ (FP64)	Output-adj. error	Probe acc. (%)
Cora	GCN	20/20	**??**	**??**	53.23±3.71
Cora	GraphSAGE	20/20	**??**	**??**	56.54±3.73
CiteSeer	GCN	20/20	**??**	**??**	36.88±2.15
CiteSeer	GraphSAGE	20/20	**??**	**??**	40.47±2.90
PubMed	GCN	20/20	**??**	**??**	61.55±3.48
PubMed	GraphSAGE	20/20	**??**	**??**	63.61±3.58

BP transport horizon on larger graphs. The depth at which BP becomes update-null is graph-dependent rather than a universal ten-layer threshold: the graph operator, weights, gates, features, labels, and normalization all matter. Table 15 reports the observed boundary on four larger graphs, including the 169K-node ogbn-arxiv (Hu et al., 2020) and Flickr (Zeng et al., 2020). Larger and better-connected graphs retain BP longer, and KAFT's advantage correspondingly appears at greater depth (Appendix J). The ogbn-arxiv result is a mechanism diagnostic, not a leaderboard claim.

Reproduction check. The released CPU-default notebook and CLI fix seeds, train the identical bias-free BP and KAFT GCN on Cora, and assert the diagnostic endpoints. Over three seeds it

Table 15: Observed BP transport boundary on larger graphs (bare GCN). "Norm-zero" means all hidden-layer native FP32 weight-gradient norms are zero.

Dataset	Observed BP transport boundary
DBLP	norm-nonzero at $L =$ **??**; norm-zero in 4/5 seeds at $L =$ **??** and 5/5 at $L =$ **??**
Flickr	norm-nonzero at $L =$ **??** (shallow/deep norm ratio **??**); norm-zero in 3/3 seeds at $L =$ **??**
ogbn-arxiv	norm-nonzero at $L =$ **??**; norm-zero in 3/3 seeds at $L =$ **??**
Coauthor-Physics	stable through approximately $L =$ **??**; increasingly unstable around $L =$ **??**–32

gives six-layer test accuracy 68.87 ± 1.20 (BP) and 78.53 ± 1.33 (KAFT); at $L = 10$, all native FP32 BP norms are zero in 3/3 seeds, the output-adjacent error is finite and nonzero, and standardized penultimate probes reach 51.4–57.8% against 14.3% chance.

J Deep Crossover and Training Dynamics on Larger Graphs

Architectural choices and graph connectivity shift the BP transport horizon (Appendix I). To test whether KAFT's benefit tracks this horizon, we compare BP and KAFT around each transition with the identical bias-free GCN, split, paired initialization, and validation protocol (seed 0 selects each method's LR; seeds 1–9 report). Table 16 shows the within-graph crossover on DBLP: the KAFT gain rises from $+4.31$ points at $L = 8$ to $+30.42$ at $L = 20$ as BP's hidden updates become null. Coauthor-Physics transitions later; BP becomes initialization-sensitive around $L = 31$–32, where KAFT gains 25–38 points. KAFT wins every paired held-out seed in all six cells.

Table 16: BP–KAFT crossover around the BP transport horizon, bias-free GCN. Seed 0 selects each method's LR; paired held-out seeds 1–9.

Dataset	L	BP	KAFT	Δ	Wins	p
DBLP	8	77.80 ± 2.47	82.11 ± 0.63	$+4.31$	9/9	3.50×10^{-4}
	12	73.25 ± 2.46	81.52 ± 0.64	$+8.26$	9/9	1.33×10^{-5}
	16	51.34 ± 8.43	80.53 ± 1.11	$+29.19$	9/9	8.41×10^{-6}
	20	46.14 ± 1.71	76.56 ± 3.74	$+30.42$	9/9	5.64×10^{-8}
Coauthor-Physics	31	46.78 ± 11.24	72.20 ± 3.50	$+25.42$	9/9	2.43×10^{-4}
	32	33.00 ± 18.33	71.41 ± 4.79	$+38.41$	9/9	1.57×10^{-4}

Training loss and training accuracy. If KAFT restores supervision transport, the effect should appear in the training dynamics and not only in test accuracy. Table 17 reports the DBLP runs of Table 16 at $L = 16$ and $L = 20$. BP stays at the uniform-posterior four-class loss $\log 4 \approx 1.386$, whereas KAFT reaches 0.629 and 0.790, and training accuracy at the validation-selected checkpoint confirms that KAFT fits the training labels under identical checkpointing. Very deep unnormalized KAFT can show late-training instability; checkpoint selection by validation accuracy is applied identically to both methods.

Table 17: Training dynamics on DBLP, bias-free GCN, seeds 1–9. "Min. loss" is the minimum of the epoch-wise median training-loss curve; "Train acc." is the median training accuracy at each run's validation-selected checkpoint; test accuracy is mean $\pm$ std over the same checkpoints.

L	Method	Min. loss	Train acc. (%)	Test acc. (%)
16	BP	$\log 4$	46.55	51.34 ± 8.43
	KAFT	0.629	82.15	$\mathbf{80.53\pm1.11}$
20	BP	$\log 4$	11.86	46.14 ± 1.71
	KAFT	0.790	78.42	$\mathbf{76.56\pm3.74}$

Uniform-posterior loss versus above-chance accuracy. The three quantities in Table 17 are aggregated differently and should not be read as one uniform model at one checkpoint. Moreover, the fixed DBLP split is class-imbalanced (train-class proportions **??**; validation and test are nearly identical). In a reproduced collapsed BP run the FP64 cross-entropy is exactly $\log 4$ and 99.435% of nodes have four bitwise-equal logits; the deterministic argmax breaks ties toward class 0, the majority class, yielding 44.63% training accuracy. Uniform posteriors therefore do not imply 25% accuracy on this split. The nine-seed BP mean test accuracy at $L = 16$ is further raised by one seed that escapes collapse and reaches approximately 71%. We accordingly describe the BP endpoint as a uniform-posterior loss with majority-class tie-breaking rather than chance accuracy.

K Comparison with Forward Normalization

BatchNorm and LayerNorm are standard forward-side remedies for unstable gradient flow. We add each to the same bias-free GCN and compare BP and KAFT under the identical normalized forward model at $L = 10$ on Cora, CiteSeer, and PubMed (width 64, 200 epochs, Adam with weight decay; seed 0 selects each method's LR from the grid **??** using validation accuracy; paired held-out seeds 1–19). BatchNorm uses running statistics at validation and test time, and all BP baselines fit the training labels.

Table 18: BP vs. KAFT under the same normalized forward GCN at $L = 10$, paired held-out seeds 1–19. Green marks the better method when the paired t-test is significant at $p < 0.05$.

Dataset	Norm	BP	KAFT	Δ	Wins	p
Cora	BN	65.88±3.29	73.18±2.06	+7.30	19/19	??
	LN	75.92±1.27	78.27±1.59	+2.35	16/19	??
CiteSeer	BN	49.05±3.52	48.71±3.19	−0.35	9/19	0.765
	LN	60.46±2.43	62.45±1.27	+1.99	16/19	0.00214
PubMed	BN	68.01±1.94	78.28±1.14	+10.27	19/19	??
	LN	74.97±1.40	74.81±1.57	−0.16	7/19	0.789

KAFT is significantly better in four of the six matched comparisons and statistically indistinguishable in the other two; none is significantly negative. On Cora the paired-difference 95% confidence intervals are $[5.54, 9.06]$ (BN) and $[1.35, 3.35]$ (LN). Forward normalization therefore strengthens BP substantially but does not make the backward intervention redundant: normalization rescales the forward Jacobians, whereas KAFT changes how supervision is transported to hidden layers. The two near-zero differences show that the combination is not universally additive.

At the shallower depth $L = 6$ with LayerNorm, BP and KAFT give 77.41 ± 1.46 vs. 79.16 ± 1.57 on Cora ($+1.75$, $p =$ **??**), 64.28 ± 1.97 vs. 63.87 ± 1.40 on CiteSeer (-0.42, n.s.), and 75.44 ± 1.89 vs. 75.30 ± 1.43 on PubMed (-0.14, n.s.), consistent with KAFT's benefit growing as BP approaches its transport horizon.

L Identity-Rich Forward Architectures

GIN's $(1 + \epsilon)I$ term and GCNII-style initial-residual/identity mixing give forward features and BP errors a skip-like route (Section 4.2). Table 19 re-runs this control under method-specific held-out LR selection (seed 0 selects each method's LR from three values; seeds 1–19 report). Identity paths reduce, but do not eliminate, KAFT's benefit on Cora. On CiteSeer, $L = 5$ remains a small significant negative and $L = 6$ becomes a non-significant tie.

Table 19: Identity-rich forward backbones with held-out, method-specific LR selection (paired seeds 1–19). Green marks the better method when the paired t-test is significant at $p < 0.05$.

Dataset / forward	L	BP	KAFT	Δ	Wins	p
Cora / GIN	6	77.76±1.59	78.92±1.30	+1.16	14/19	0.0195
Cora / GIN	10	74.92±2.41	76.55±1.78	+1.63	13/19	0.0222
Cora / GCNII-style	6	78.03±1.27	78.86±1.77	+0.83	13/19	0.0366
CiteSeer / GIN	5	66.85±1.10	65.91±1.21	−0.94	??	0.0241
CiteSeer / GIN	6	64.98±1.67	64.26±1.61	−0.73	??	0.162

Two protocols for CiteSeer GIN. The CiteSeer GIN $L = 6$ numbers in Table 6 and Table 19 answer different questions. Table 6 uses the submitted common-LR protocol (20 seeds): BP 65.12, KAFT 63.05, $\Delta = -2.07$, $p = 0.00419$. Table 19 is a sensitivity analysis asking whether the degradation persists after validation-only, method-specific LR selection: the gap shrinks to -0.73 and becomes non-significant. When BP is already healthy through an identity path, KAFT trades topology-approximation bias against a transport benefit that is no longer needed; we retain CiteSeer GIN $L = 5$ as a real, small negative boundary rather than a conclusion about GIN in general.

When to prefer KAFT. Combining Appendices J–L: KAFT is useful when the desired depth exceeds the graph- and architecture-specific BP transport horizon. Identity paths, normalization, and connectivity postpone that horizon, after which the gain expands to 25–38 points. KAFT leaves the forward and inference graph unchanged, adds no inference-time module, and composes with forward remedies (Table 4, Table 18).

M Disentangling Output-Error Diffusion and the Topology Factor

KAFT's graph side contains two operators: the diffusion $D(\hat{A})$, which spreads the sparse labeled-node error to unlabeled nodes, and the layer-dependent topology factor $P_\ell(\hat{A}) = \hat{A}^{r_\ell}$, which mirrors the graph factor of the suffix Jacobian. The wrong-topology control of Section 4.6 tests whether any structured operator suffices but does not separate these two roles. Table 20 reports the full 2×2 factorial on DBLP GCN $L =$ **??** with fixed LR 10^{-2}, identical hyperparameters, and 20 paired initializations.

Table 20: 2×2 ablation of the graph-side operators on DBLP GCN, 20 paired seeds. "Label spreading" is the submitted $D(\hat{A})$ (10 iterations, $\alpha = 0.5$).

Diffusion $D(\hat{A})$	Topology $P_\ell(\hat{A})$	Test acc. (%)	Median final train loss
I	I	80.95±0.62	2.82
I	$\hat{A}^{r_\ell}$	82.31±0.32	0.0136
label spreading	I	81.99±0.65	0.159
label spreading	$\hat{A}^{r_\ell}$	**82.41±0.66**	0.276

Table 21: Paired comparisons from Table 20.

Comparison	Gain	Wins	p
Add P_ℓ when $D = I$	+1.36	20/20	**??**
Add D when $P_\ell = I$	+1.04	19/20	**??**
Full method vs. $D = P_\ell = I$	+1.46	18/20	**??**
Add D when $P_\ell = \hat{A}^{r_\ell}$	+0.10	11/20	0.571

Each operator helps significantly on its own, so the topology factor is not reducible to label spreading. Beyond the mean, P_ℓ reduces the seed standard deviation from 0.62 to 0.32, raises the worst seed by 1.83 points, and reduces the number of runs with final training loss above 1 from 17/20 to 0/20. Near the BP transition it gains +1.52 (9/9 wins) at $L =$ **??** and +2.24 (8/9) at $L =$ **??**, reducing the standard deviation from 0.85 to 0.37 and from 2.80 to 1.73. The two gains are not additive: because both operators move graph-side error, their benefits overlap and largely saturate on DBLP, and the small, non-significant increment of D on top of P_ℓ reflects this saturation rather than an identical mathematical role. We keep both, since the joint configuration attains the highest mean accuracy.

N Heterophilous Graphs

Heterophily is not an assumption of KAFT's factorization. For any forward layer $Z_{\ell+1} = MH_\ell W_{\ell+1}$ with a fixed propagation operator M, the preactivation Jacobian is $W_{\ell+1}^\top \otimes M$, irrespective of labels or homophily. The forward backbone chooses M; KAFT aligns its backward topology with it. The limitation discussed in Section 6 concerns the unsigned-GCN instantiation studied in the main text, not the rule itself.

We test KAFT on the heterophilous Minesweeper benchmark (Platonov et al., 2023) using an unchanged six-layer GCN with BatchNorm. Official split 0 selected LR 10^{-2} for both BP and KAFT; configurations were then frozen for official splits 1–9.

Table 22: Minesweeper, six-layer GCN+BN, ROC-AUC (%) on official splits 1–9.

KAFT variant	BP+BN	KAFT+BN	Δ	Wins	p
Full submitted method	66.06±0.68	67.50±0.73	+1.43	9/9	**??**
$D(\hat{A}) = I$ control	66.06±0.68	68.12±0.47	+2.06	9/9	**??**

KAFT improves every held-out split. Removing the label-spreading diffusion increases the gain to +2.06 AUC, which rules out homophilous label spreading as the source of the improvement. We do not claim that one unsigned operator is optimal for all heterophilous graphs; the separated P_ℓ and learned R_ℓ can equally accommodate signed, directional, or relation-specific forward operators.

O Training Memory on Larger and Deeper Graphs

We measure peak CUDA allocation with the same bias-free GCN at width 64. Each method, depth, and seed runs in a fresh process: one warm-up step, an allocator reset, and then the maximum over three steady-state steps. We use the reference KAFT with per-step feature alignment, not the amortized implementation. Seeds 0–2 give identical peaks since shapes determine allocation.

Table 23: Peak training memory (MiB), bias-free GCN, width 64, reference KAFT.

Dataset	L	BP	KAFT	KAFT/BP
DBLP	6	176.6	216.3	1.225×
DBLP	20	237.8	399.1	1.678×
Coauthor-Physics	20	1371.9	1684.8	1.228×
Coauthor-Physics	32	1473.5	1988.9	1.350×

On the 34,493-node Coauthor-Physics graph, where graph, features, activations, and optimizer state dominate allocation, the ratio is $1.23\times$ at $L = 20$ and $1.35\times$ at $L = 32$. On the smaller DBLP graph it rises from $1.23\times$ to $1.68\times$ because per-layer state is a larger share of the total; the overhead is therefore not depth-independent. For fixed K, the shared topology stack holds at most $K+1$ error tensors independent of L, while the compact $C \times d_\ell$ matrices R_ℓ, the layerwise updates, and the per-step alignment state scale with L. All of this state is training-only; the unchanged forward GNN has no deployment-memory overhead.

Role of the hop cap. The cap K bounds only the graph-side backward surrogate $P_\ell(\hat{A}) = \hat{A}^{\min(L-1-\ell,K)}$; it does not alter the L-layer forward network or its L-hop receptive field. Since the full graph-side feedback is $P_\ell(\hat{A})D(\hat{A})$ and $D(\hat{A})$ already performs 10 diffusion iterations, $K = 3$ is not a three-hop supervision limit. Copying the exact $\hat{A}^{L-1-\ell}$ would recreate a depth-growing path that repeatedly suppresses non-principal graph modes; capping trades this bias for a fixed critical path and less error oversmoothing. Table 12(c) shows accuracy is flat for $K \in \{1, 2, 3, 5\}$, with no benefit at $K = 5$; applications that need longer backward range can validate K against accuracy, propagation time, and memory. Throughout, "parallel" refers to the dependency depth of hidden feedback, which is independent of L for fixed K, not to constant total backward time (Table 10).

P Preliminary Extensions Beyond Node Classification

KAFT only requires the exact task-loss derivative at the encoder output; structured feedback replaces BP only inside the hidden encoder layers and uses the forward encoder's graph operator. We implemented three preliminary extensions (three seeds each), summarized in Table 24.

Table 24: Preliminary extensions, three seeds. Link prediction reports ROC-AUC (%); graph classification reports accuracy (%).

Task	Protocol	BP	KAFT
Link prediction	Cora, six-layer GCN encoder, inner-product decoder	64.51±2.44	**81.39±2.44**
Edge-varying link prediction	As above; 20% of training edges resampled/dropped per epoch	61.37±2.98	**83.61±2.19**
Graph classification	ENZYMES, six-layer encoder, batch size 64	39.44±8.39	40.00±10.14

For link prediction, both methods use the same GCN encoder, inner-product decoder, and training-edge-only adjacency, and both obtain the exact decoder derivative; KAFT then transports it through the hidden layers. The gain therefore extends beyond node-label cross-entropy. The edge-varying variant resamples 20% of training edges each epoch and uses the same adjacency for the forward pass, the diffusion, and the KAFT topology; it tests rebuilding aligned feedback under changing topology and is an edge-resampling stress test rather than a temporal-graph benchmark. ENZYMES uses block-diagonal sparse mini-batches; the graph-level readout is trained by exact BP in both methods and KAFT transports its derivative through mean pooling into the hidden encoder. The tie establishes end-to-end mini-batch and graph-level feasibility, not an accuracy gain. Neighbor-sampled large-scale training remains open, since forward/backward sampling consistency and the global $D(\hat{A})$ require care.

Q Convergence Analysis

KAFT's hidden update need not be a conservative vector field, so in general no fixed scalar objective has every KAFT update as its gradient. We instead ask when the KAFT direction is a descent direction for the task loss. The analysis below separates an exact algebraic property of the probe target (Lemma 2), an exact kernel identity (Lemma 1), and standard sufficient conditions for descent (Theorem 1); nonlinear gates, the hop cap, diffusion, and tracking error of R_ℓ enter as explicit perturbations rather than being assumed away.

Setup. Let $\theta = (W_0, \dots, W_{L-1})$, let $f(\theta) \in \mathbb{R}^{NC}$ be the vectorized training logits, and $\mathcal{L}(\theta) = \ell(f(\theta))$ with $g := \nabla_f \ell$. Let $J_\ell = \partial f / \partial \mathrm{vec}(W_\ell)$ be the actual (gated, nonlinear) Jacobians, so the BP gradient has blocks $J_\ell^\top g$. KAFT uses the exact block for the output layer and $\widetilde{J}_\ell^\top g = \mathrm{vec}(S_\ell^\top \delta_\ell^{\mathrm{KAFT}})$ for hidden layers, which defines surrogate Jacobians $\widetilde{J}_\ell$ ($\widetilde{J}_{L-1} = J_{L-1}$). Write u for the stacked KAFT direction.

Lemma 1 (Exact alignment identity)**.** $\langle \nabla\mathcal{L}(\theta), u\rangle = g^\top \mathcal{K} g$ *with* $\mathcal{K} := \sum_{\ell=0}^{L-1} J_\ell \widetilde{J}_\ell^\top$.

Proof. $\langle \nabla\mathcal{L}, u\rangle = \sum_\ell \langle J_\ell^\top g, \widetilde{J}_\ell^\top g\rangle = \sum_\ell g^\top J_\ell \widetilde{J}_\ell^\top g$. □

The identity uses the actual nonlinear Jacobians and does not require the nonlinear suffix to be Kronecker-separable. For softmax cross-entropy, each node's logit gradient sums to zero, so g lies in the class-centered subspace; positivity of $\mathcal{K}$ is only needed there.

Lemma 2 (PSD property of the chain-normalized target)**.** *Let $Q_\ell \in \mathbb{R}^{C\times d_\ell}$ be the feature suffix of Section 3.2 and $\widetilde{Q}_\ell = \frac{1}{m} Y \Phi_\ell^\top$ the probe target of Section 3.4, for any rank of Q_ℓ. Then*

$$\widetilde{Q}_\ell Q_\ell^\top = \frac{1}{m}\sum_{j=1}^{m} c_j\,(Q_\ell\phi_j)(Q_\ell\phi_j)^\top \succeq 0, \qquad c_j > 0,$$

and $\ker(\widetilde{Q}_\ell Q_\ell^\top) = \mathrm{span}\{Q_\ell\phi_j\}_{j=1}^m{}^\perp$. *If the probes are Gaussian and $m \geq \mathrm{rank}(Q_\ell)$, then $\widetilde{Q}_\ell Q_\ell^\top$ is almost surely strictly positive on* $\mathrm{range}(Q_\ell)$.

Proof. Every intermediate normalization (including the ε term) multiplies a probe column by a positive scalar, and positive scalars commute with the linear maps $W_k^\top$. Hence the final column is $Y_{:j} = c_j Q_\ell \phi_j$ with $c_j > 0$, and $\widetilde{Q}_\ell = \frac{1}{m}\sum_j c_j Q_\ell \phi_j \phi_j^\top$; multiplying by $Q_\ell^\top$ gives the displayed sum of rank-one PSD terms. For $v \in \mathbb{R}^C$, $v^\top \widetilde{Q}_\ell Q_\ell^\top v = \frac{1}{m}\sum_j c_j (v^\top Q_\ell \phi_j)^2$, which vanishes iff $v \perp Q_\ell\phi_j$ for all j. For Gaussian probes with $m \geq \mathrm{rank}(Q_\ell)$, the images $Q_\ell\phi_j$ span $\mathrm{range}(Q_\ell)$ almost surely. □

This is a finite-probe identity, not an unbiasedness claim: chain normalization preserves the probe directions and the PSD sign while discarding the ill-conditioned product scale. Small $\|Q_\ell\|$ does not by itself make the factor degenerate; only directions outside $\mathrm{range}(Q_\ell)$ are lost, and full row rank is needed only for a single layer to cover all of $\mathbb{R}^C$.

Corollary 1 (Matched linear GCN)**.** *For a linear GCN ($\sigma = \mathrm{id}$) with $D(\hat{A}) = I$, no hop cap ($r_\ell = L-1-\ell$), and $R_\ell = \widetilde{Q}_\ell$, each hidden-layer cross-kernel is*

$$J_\ell \widetilde{J}_\ell^\top = (Q_\ell \widetilde{Q}_\ell^\top) \otimes \big(\hat{A}^{L-1-\ell} S_\ell S_\ell^\top \hat{A}^{L-1-\ell}\big) \succeq 0,$$

so $\mathcal{K} = J_{L-1}J_{L-1}^\top + \sum_{\ell<L-1} J_\ell \widetilde{J}_\ell^\top$ is PSD at arbitrary depth, rank, and differentiable loss.

Proof. With $p = L-1-\ell$, $f = \mathrm{vec}(\hat{A}^p S_\ell W_\ell Q_\ell^\top)$, and the KAFT block is $S_\ell^\top \hat{A}^p E \widetilde{Q}_\ell$ with E the matrix form of g ($\hat{A}$ is symmetric). Composing, $J_\ell \widetilde{J}_\ell^\top$ maps $E \mapsto \hat{A}^p S_\ell S_\ell^\top \hat{A}^p\, E\, \widetilde{Q}_\ell Q_\ell^\top$, whose vectorization is the stated Kronecker product. Both factors are PSD (the first by Lemma 2), hence so is their Kronecker product, and a sum of PSD matrices is PSD. □

The topology factor is what makes the node-side factor PSD: with a mismatched backward graph $\hat{A}'$, the node factor becomes $\hat{A}^p S_\ell S_\ell^\top \hat{A}'^p$, which is not PSD in general—a formal counterpart of the topology-consistency result in Section 4.6. Collective coverage also suffices: for PSD matrices, $\ker(\sum_\ell K_\ell) = \bigcap_\ell \ker K_\ell$, so the output layer and different hidden layers can cover different task directions.

Deviations from the matched case. In the actual method, write $\mathcal{K} = \mathcal{K}^\star + \Delta$, where $\mathcal{K}^\star$ is the matched PSD kernel of Corollary 1 evaluated along the trajectory and Δ collects (i) the suffix activation gates, whose operator error is bounded by the standard telescoping product bound, (ii) the hop cap and the diffusion $D(\hat{A})$, and (iii) the tracking error $R_\ell - \widetilde{Q}_\ell$. By Weyl's inequality, on any subspace $\mathcal{S}$ containing g,

$$\lambda_{\min}\big(\mathrm{sym}(\mathcal{K})|_{\mathcal{S}}\big) \;\geq\; \lambda_{\min}\big(\mathcal{K}^\star|_{\mathcal{S}}\big) - \|\Delta\|_2,$$

so the positive margin is preserved whenever the matched margin dominates these deviations. For tracking, with U an orthonormal basis of $\mathcal{S}$, it suffices that $\|U^\top(R_\ell - \widetilde{Q}_\ell)Q_\ell^\top U\|_2$ is below the restricted eigenvalue of $\widetilde{Q}_\ell Q_\ell^\top$; positive average cosine alone is not such a certificate.

Theorem 1 (Descent under positive alignment)**.** *Let $\mathcal{L}$ be bounded below by $\mathcal{L}^\star$ with β-Lipschitz gradient, and let $\theta_{t+1} = \theta_t - \eta u_t$ where, along the trajectory, $\langle \nabla\mathcal{L}(\theta_t), u_t\rangle \geq \mu\|\nabla\mathcal{L}(\theta_t)\|^2$ and $\|u_t\| \leq M\|\nabla\mathcal{L}(\theta_t)\|$ for some $\mu, M > 0$. If $0 < \eta \leq \mu/(\beta M^2)$, then*

$$\mathcal{L}(\theta_{t+1}) \leq \mathcal{L}(\theta_t) - \tfrac{\eta\mu}{2}\|\nabla\mathcal{L}(\theta_t)\|^2, \qquad \min_{t<T}\|\nabla\mathcal{L}(\theta_t)\|^2 \leq \frac{2(\mathcal{L}(\theta_0) - \mathcal{L}^\star)}{\eta\mu T}.$$

If in addition $\mathcal{L}$ satisfies the Polyak–Łojasiewicz condition $\frac{1}{2}\|\nabla\mathcal{L}\|^2 \geq \rho(\mathcal{L} - \mathcal{L}^\star)$, then $\mathcal{L}(\theta_t) - \mathcal{L}^\star \leq (1 - \eta\mu\rho)^t(\mathcal{L}(\theta_0) - \mathcal{L}^\star)$.

Proof. The descent lemma gives $\mathcal{L}(\theta_{t+1}) \leq \mathcal{L}(\theta_t) - \eta\langle\nabla\mathcal{L}, u_t\rangle + \frac{\beta\eta^2}{2}\|u_t\|^2 \leq \mathcal{L}(\theta_t) - \eta(\mu - \frac{\beta\eta M^2}{2})\|\nabla\mathcal{L}\|^2$, and $\eta \leq \mu/(\beta M^2)$ bounds the bracket below by $\mu/2$. Summing over $t < T$ and using $\mathcal{L} \geq \mathcal{L}^\star$ gives the stationarity bound; combining the one-step bound with the PL inequality gives the linear rate. □

By Lemma 1, the alignment condition holds whenever $\lambda_{\min}(\mathrm{sym}(\mathcal{K})|_S)\,\|g\|^2 \geq \mu\|\nabla\mathcal{L}\|^2$, which is implied by a positive kernel margin together with a bounded Jacobian norm. This is the direction-alignment argument of Caillon et al. (2026); with diminishing Robbins–Monro steps it also yields convergence to stationarity in the stochastic setting. Softmax cross-entropy is smooth in logit space, and the transductive node-classification protocol is full-batch.

Squared-loss residual contraction. As a rate-bearing special case following Song et al. (2021), let $r = f(\theta) - y$ under squared loss and gradient flow $\dot\theta = -u$. Then $\dot r = -\mathcal{K}r$ and $\frac{d}{dt}\|r\|^2 = -2r^\top\mathcal{K}r$. If $\lambda_{\min}(\mathrm{sym}(\mathcal{K})) \geq \lambda > 0$ along the trajectory, then $\|r(t)\| \leq e^{-\lambda t}\|r(0)\|$: the squared residual is a Lyapunov function even though no global parameter-space potential exists. The two-layer random-feedback theorem of Song et al. (2021) does not itself cover KAFT; the result here is conditional on the kernel margin.

Finite steps and optimizers. For a finite ReLU update, gate crossings add an exact remainder to the linearization, which must also be dominated by the margin. With Adam, the sufficient condition of Theorem 1 must be imposed on the actual preconditioned moment direction; raw KAFT alignment does not by itself imply an unconditional Adam guarantee. These conditions are checkable along training, but they are sufficient conditions rather than an unconditional theorem for deep ReLU networks trained with cross-entropy and Adam.

Empirical alignment at depth. At $L = 6$, where BP gradients are measurable, KAFT and BP hidden directions have positive cosine at every hidden layer (Table 9). At $L = 10$ the BP update is FP32-null (Appendix I), so a cosine against it is undefined; there we instead verify the target, the nonzero KAFT feedback, the training curves (Table 17), and recovery after the BP stall.