This MICCAI paper is the Open Access version, provided by the MICCAI Society. It is identical to the accepted version, except for the format and this watermark; the final published version is available on SpringerLink.

Supplementary Material

Towards Highly Heterogeneous Federated Learning: A Graph-Based Framework for Architectural and Data Heterogeneity

Furkan Pala and Islem Rekik

BASIRA Lab, Imperial-X (I-X) and Department of Computing, Imperial College London, London, United Kingdom
f.pala23@imperial.ac.uk, i.rekik@imperial.ac.uk

This document supports the main paper. Section S1 fixes the notation and states which parameters are updated at which stage of training. Section S2 specifies the architecture-to-graph conversion and shows that node features are specific to the block they describe. Section S3 defines the AH metric and validates it. Section S4 states the assumptions and gives the derivations behind the theoretical analysis. Section S5 reports the complete per-client results. Section S6 reports what we can and cannot measure of the two terms of the break-even condition. Section S7 gives the gradient-probe statistics with their p-values. Section S8 lists the training protocol, and Section S9 gives the full results for the three medical imaging types.

S1 Notation, and What Is Learned When

We consider a federation of m clients. Client k holds a private dataset of n_k samples ($n = \sum_k n_k$, $w_k = n_k/n$) and a private backbone f_{θ_k}.

The four symbols, and how they differ. Two of them belong to the shared graph neural network and two are private to the client. In particular, ϕ is a learned parameter, whereas γ and β are not:

- ϕ: the weights of the shared graph attention network g_ϕ (≈88k parameters). *This is the only object that is ever transmitted or averaged.* It is a parameter, updated by gradient descent at the client and by weighted averaging at the server.
- $(\gamma^{(v)}, \beta^{(v)})$: *not* parameters. They are *activations*: the output of a private 2-layer MLP applied to the node embedding $z^{(v)} \in \mathbb{R}^{64}$ that g_ϕ produces for node v. They are recomputed on every forward pass and never stored or shared. They are used only to modulate the backbone; they play no role in graph learning.
- ψ_k: the weights of that private MLP (the FiLM adapter), $\mathbb{R}^{64} \to \mathbb{R}^{128} \to \mathbb{R}^{2d_k}$. These weights are private and are never averaged.
- α_k: a private learned gate, initialized to 0, one scalar per hooked block. It scales how much of the adjustment the backbone actually receives.

For a hooked block v with hidden state h, the feature adjustment is

$$z^{(v)} = \big[g_\phi(G_k)\big]_v \in \mathbb{R}^{64}, \qquad (\gamma^{(v)}, \beta^{(v)}) = \mathrm{MLP}_{\psi_k}\big(z^{(v)}\big), \\ h \;\leftarrow\; h + \alpha_k^{(v)}\big(\gamma^{(v)} \odot h + \beta^{(v)}\big). \tag{S1}$$

At $\alpha_k = 0$ this is the identity, so training starts from the client's unmodified backbone. For a 4-D convolutional state (B, C, H, W), γ, β broadcast channel-wise; for a 3-D token state (B, N, D) they broadcast feature-wise. Because CNN stages differ in width, the MLP emits $2d_k$ values for the widest hooked block and the output is sliced to each block's own width.

Algorithm S1 GFL. Everything private to client k is marked (priv).

Input: backbones f_{θ_k}, graphs G_k, rounds T
server initializes ϕ^0; each client initializes θ_k (priv), ψ_k (priv), $\alpha_k \leftarrow 0$ (priv)
for $t = 0, \ldots, T-1$ **do**
  server broadcasts ϕ^t to all clients
  for each client k **in parallel do**
    $\phi_k \leftarrow \phi^t$
    for each local minibatch **do**
      refresh the dynamic node features of G_k from the live weights and activations of f_{θ_k}
      $Z_k \leftarrow g_{\phi_k}(G_k)$; forward f_{θ_k} with Eq. (S1) applied at each hooked block
      one gradient step on ϕ_k *and* on $\theta_k, \psi_k, \alpha_k$ (priv)
    upload ϕ_k *only*
  server: $\phi^{t+1} \leftarrow \sum_k w_k \, \phi_k$ ▷ no other tensor is aggregated

The three paradigms differ by one line of Algorithm S1. **Individual**: $\alpha_k \equiv 0$ and g_ϕ is switched off, so only θ_k trains. **GL**: identical to GFL, except the upload/average step is deleted, so each client keeps its own ϕ_k. **GFL**: as written. Nothing else changes; the same optimizer, schedule, seeds and data splits are used for all three.

S2 Architecture-to-Graph Conversion, and Why Node Features Are Block-Specific

Nodes. We traverse the backbone with the `named_modules()` call of PyTorch and keep every layer whose class appears in the twelve-entry vocabulary of Table S1. This retains both the container blocks, which carry the FiLM hooks, and the individual layers inside them. The nodes are kept in the order that this traversal returns.

Edges. Consecutive nodes are joined bidirectionally (a chain), and a self-loop is added at every node, so a graph with N nodes has $3N - 2$ directed edges. The topology is fixed at construction and never learned.

Node features. Each node carries a fixed-length descriptor, Table S2: 16 base dimensions plus, in the runs reported in the paper, three functional statistics (dims 16–18) refreshed during training, for 20 in total. Dimensions 0, 2, 3, 14 and 15 are exactly the capacity and shape of the block, so *two blocks of the same type but different width or kernel cannot receive the same feature vector.*

Table S1. Layer-type vocabulary. The identifier stored in node-feature dimension 0, and the vocabulary over which AH is computed (Sec. S3).

Block	Linear	LayerNorm	Embedding	MultiheadAttn	Conv2d	BatchNorm2d	GroupNorm	ConvBlock	BasicBlock	ViTBlock	MixerBlock	MLPBlock
id	1	2	3	4	5	6	7	8	9	10	11	12

Table S2. Node-feature schema. Dimensions 6–8 and 16–18 are dynamic and refreshed from the live backbone during training; the rest are fixed at construction. N is the node count of the graph.

Dim	Content	Role
0	layer-type identifier (Table S1)	what kind of block
1	parameter count / 10^6	capacity
2, 3	input, output width / 1024	shape
4	depth ratio, index/$(N-1)$	where in the stack
5	encoder flag (always 1 here)	n/a
6–8	weight mean, sd, $\|W\|_2/\sqrt{\|W\|}$	live state
9–13	is-attention, is-linear, is-norm, is-embedding, has-bias	type flags
14	parameters relative to the whole backbone	relative capacity
15	convolution kernel size (0 if not a conv)	receptive field
16–18	activation mean, sd, gradient RMS (EMA)	live functional state
19	reserved	n/a

Capacity is visible to the shared network. As a concrete check, ResNet18's 20 convolutional nodes carry 11 distinct (in, out, kernel) signatures, with parameter counts spanning three orders of magnitude (from 1.7×10^3 at the stem to 2.36×10^6 at the last 512×512 convolution) and relative-capacity values (dim 14) from 0.0002 to 0.21. Since g_ϕ is a graph attention network on these features, distinct inputs at distinct depths yield distinct node embeddings $z^{(v)}$ and hence distinct (γ, β). Table S3 confirms that no two nodes share a feature vector in any of the six backbones.

S3 The AH Metric: Definition and Validation

Let client i's backbone have layer-type histogram $c_i \in \mathbb{N}^{12}$ over the vocabulary of Table S1, and let $p_i = c_i/\|c_i\|_1$. With $\mathrm{JSD}(p\|q) = \frac{1}{2}D_{\mathrm{KL}}(p\|\bar{r}) + \frac{1}{2}D_{\mathrm{KL}}(q\|\bar{r})$,

Table S3. Graph sizes, and node-feature diversity. Every backbone produces a graph of a different size, and in every backbone *all* node feature vectors are pairwise distinct, so the count of distinct rows equals the node count. The last column counts distinct (layer type, input width, output width) signatures, that is, how many distinct block shapes the shared GNN must distinguish within one backbone.

Backbone	Nodes	Edges	Distinct feature rows	Distinct (type, in, out)
SimpleCNN	10	28	10	10
VGG9	26	76	26	22
ResNet18	49	145	49	20
ViT-Tiny	87	259	87	8
MLP-Mixer-S	45	133	45	8
MLP	19	55	19	6

$\bar{r} = \frac{1}{2}(p+q)$, in base 2,

$$\mathrm{AH}(\{f_{\theta_i}\}_{i=1}^{m}) = \binom{m}{2}^{-1} \sum_{i<j} \mathrm{JSD}\big(p_i \,\|\, p_j\big). \tag{S2}$$

Base-2 JSD is symmetric, finite when supports differ, and bounded in $[0, 1]$; it is 0 iff all clients run the same layer-type mix and 1 for disjoint vocabularies. Data heterogeneity is measured on exactly the same scale, as the mean pairwise JSD between clients' *label* histograms, so the two axes are commensurable. Both are computable before training: AH from model definitions alone, DH from the partition.

Table S4 gives the histograms, Table S5 the pairwise divergences and Table S6 the resulting AH of each backbone combination. The metric behaves as intended: SimpleCNN and VGG9 are nearly indistinguishable (0.0012, both are plain conv–BN–block stacks), ResNet18 sits apart from them (≈ 0.26) because its residual `BasicBlock` replaces `ConvBlock`, and every convolutional–to–non-convolutional pair exceeds 0.73.

Table S4. Layer-type histograms c_i (block counts). Empty columns of the vocabulary (Embedding, GroupNorm) are omitted.

Backbone	Linear	LayerNorm	MultiheadAttn	Conv2d	BatchNorm2d	ConvBlock	BasicBlock	ViTBlock	MixerBlock	MLPBlock
SimpleCNN	1	0	0	3	3	3	0	0	0	0
VGG9	2	0	0	8	8	8	0	0	0	0
ResNet18	1	0	0	20	20	0	8	0	0	0
ViT-Tiny	37	25	12	1	0	0	0	12	0	0
MLP-Mixer-S	25	13	0	1	0	0	0	0	6	0
MLP	10	5	0	0	0	0	0	0	0	4

Limits of the metric. AH is deliberately coarse: it reads only the *multiset of layer types*, so it is blind to width, depth ordering and connectivity. Two backbones with identical type histograms but very different capacities score AH = 0,

Table S5. Pairwise Jensen–Shannon divergence between the layer-type distributions of Table S4 (base 2).

	SimpleCNN	VGG9	ResNet18	ViT-Tiny	Mixer-S	MLP
SimpleCNN	0.0000	0.0012	0.2643	0.7801	0.7397	0.8016
VGG9	0.0012	0.0000	0.2583	0.8092	0.7724	0.8339
ResNet18	0.2643	0.2583	0.0000	0.9022	0.8733	0.9372
ViT-Tiny	0.7801	0.8092	0.9022	0.0000	0.2121	0.2532
Mixer-S	0.7397	0.7724	0.8733	0.2121	0.0000	0.1838
MLP	0.8016	0.8339	0.9372	0.2532	0.1838	0.0000

Table S6. AH of the six backbone combinations of the main paper's Table 3, from Eq. (S2). Combinations were selected to step through the axis roughly evenly while adding one inductive-bias family at a time.

Backbone combination	Families	AH
5 × CNN	1	0.0000
4 × CNN + ResNet18	2	0.1057
4 × CNN + Mixer-S	2	0.2959
CNN + MLP + 3 × Mixer-S	3	0.3572
CNN + 2 × VGG9 + 2 × ViT	3	0.4799
CNN + 2 × ResNet18 + ViT + MLP	4	0.6042

and the near-zero SimpleCNN / VGG9 divergence (0.0012) is exactly this effect: VGG9 has thirteen times the parameters of SimpleCNN. AH is intended as a cheap, training-free *screening* statistic a consortium can compute before committing GPU time, not as a complete description of architectural distance. The mechanism-level counterpart, ζ_{arch} (Sec. S7), does see capacity, because it is measured from gradients.

S4 Theoretical Analysis in Full

This section states the assumptions and gives in full the two derivations that Sec. 3.2 of the main paper summarizes. Write $F_k(\phi, u_k)$ for client k's risk, where $u_k = (\theta_k, \psi_k, \alpha_k)$ collects the private state, and $\bar{F} = \sum_k w_k F_k$.

S4.1 No harm at initialization, and gated engagement

Because α_k enters Eq. (S1) multiplicatively, the shared gradient factorizes:

$$\nabla_\phi F_k(\phi, u_k) \;=\; \alpha_k \, G_k(\phi, u_k), \qquad G_k = \frac{\partial F_k}{\partial(\gamma, \beta)} \frac{\partial(\gamma, \beta)}{\partial \phi}. \tag{S3}$$

At $\alpha_k = 0$ the backbone's forward pass is exactly the Individual one and $\nabla_\phi F_k = 0$. **This is a statement about initialization only.** It guarantees that GFL

starts from Individual training and that the client, not the server, sets the rate at which shared signal is admitted. It does *not* guarantee a non-negative gain at convergence: once $\alpha_k > 0$, the client is optimizing against an averaged ϕ that need not suit it, and Sec. 4.2 of the main paper reports exactly that outcome under the strongest label skew ($\Delta < 0$). The correct reading is that the gate removes the *initialization* risk of joining, while the break-even condition below governs the *asymptotic* one.

S4.2 Two sources of gradient disagreement

Let $F_{P,A}$ be the risk with data distribution P and model pathway A (architecture graph, backbone family, private state), P_k client k's distribution and $\bar{P} = \sum_k w_k P_k$ the mixture. Define

$$\zeta_{\text{data}} = \max_k \sup_{\phi,u} \left\| \nabla_\phi F_{P_k,A_k} - \nabla_\phi F_{\bar{P},A_k} \right\|, \tag{S4}$$

$$\zeta_{\text{arch}} = \max_k \sup_{\phi,u} \left\| \nabla_\phi F_{\bar{P},A_k} - \textstyle\sum_j w_j \nabla_\phi F_{\bar{P},A_j} \right\|. \tag{S5}$$

The quantity ζ_{data} swaps the data at a fixed pathway, whereas ζ_{arch} swaps the pathway at fixed mixture data. ζ_{arch} vanishes when every client runs the same backbone.

Proposition 1 (Heterogeneity decomposition). *For every k, $\|\nabla_\phi F_k - \nabla_\phi \bar{F}\| \le 2\zeta_{\text{data}} + \zeta_{\text{arch}}$, and hence $\delta^2 \le 8\zeta_{\text{data}}^2 + 2\zeta_{\text{arch}}^2$ for the gradient diversity constant δ.*

Proof. Insert two counterfactual midpoints and apply the triangle inequality:

$$\begin{aligned} \nabla_\phi F_k - \nabla_\phi \bar{F} = {} & \underbrace{\left[\nabla_\phi F_{P_k,A_k} - \nabla_\phi F_{\bar{P},A_k}\right]}_{\text{own data swap} \le \zeta_{\text{data}}} \\ & + \underbrace{\left[\nabla_\phi F_{\bar{P},A_k} - \textstyle\sum_j w_j \nabla_\phi F_{\bar{P},A_j}\right]}_{\text{pathway mismatch at common data} \le \zeta_{\text{arch}}} \\ & + \underbrace{\textstyle\sum_j w_j \left[\nabla_\phi F_{\bar{P},A_j} - \nabla_\phi F_{P_j,A_j}\right]}_{\text{peers' data swaps} \le \sum_j w_j \zeta_{\text{data}} = \zeta_{\text{data}}} . \end{aligned}$$

The second display follows from $(a+b)^2 \le 2a^2 + 2b^2$ with $a = 2\zeta_{\text{data}}$, $b = \zeta_{\text{arch}}$.

The partial-personalization convergence result of Pillutla et al. assumes L-smoothness of each F_k in (ϕ, u_k), bounded stochastic-gradient variance, and bounded gradient diversity δ. Under these assumptions it gives an $\mathcal{O}(1/\sqrt{T})$ rate to a stationary point, with a drift penalty $\mathcal{O}(\tau\delta^2/T)$ for τ local steps. Substituting Proposition 1 into that result, the drift penalty becomes $\mathcal{O}\big(\tau(\zeta_{\text{data}}^2 + \zeta_{\text{arch}}^2)/T\big)$. Data skew and architectural spread therefore degrade convergence *additively*: removing one leaves the other's penalty intact. This is the formal content of the claim that the two axes are separable, and it is what licenses the factorial grid of Sec. 4.2 of the main paper.

S4.3 The gradient probe

Eqs. (S4)–(S5) are statements about ∇_ϕ at a *fixed* ϕ, not about optimizer trajectories, so we estimate them directly. Given a finished run: (i) form $\hat{\phi} = \sum_k w_k \phi_k$ from the clients' final GAT states; (ii) for each client, backpropagate its own (backbone, adapter, graph) at $\hat{\phi}$ on B fixed minibatches from the mixture distribution, recording $\hat{g}_k^{\text{mix}}$, and likewise $\hat{g}_k^{\text{own}}$ on its own shard. Then

$$\hat{\zeta}^2_{\text{arch}} = \sum_k w_k \big\| \hat{g}_k^{\text{mix}} - \textstyle\sum_j w_j \hat{g}_j^{\text{mix}} \big\|^2, \qquad \hat{\zeta}_{\text{data}}(k) = \big\| \hat{g}_k^{\text{own}} - \hat{g}_k^{\text{mix}} \big\|. \tag{S6}$$

No learning rate, local-step count or optimizer enters this estimate. We report the trajectory-free probe rather than the dispersion of the clients' updates because the latter is only proportional to δ^2 under plain SGD; under Adam the step is not gradient-proportional and the statistic reads the optimizer noise floor instead (2.24 even for five identical backbones on near-IID data, against a probe reading of 0.135 on the same run).

S4.4 The break-even condition

For non-convex training there is no general criterion that decides whether federating helps or harms a given client. Following Donahue and Kleinberg, we therefore answer the question exactly within a surrogate that retains the structure of the mechanism. Linearizing around the engaged regime, client k's excess risk in the shared block is quadratic, $R_k(\hat{\phi}) = \frac{a_k}{2}\,\mathbb{E}\|\hat{\phi} - \phi_k^\star\|^2$ with curvature $a_k \propto \alpha_k^2$. Decompose the spread of the client optima through the same two counterfactuals,

$$\phi_k^\star = \bar{\phi}^\star + b_k^{\text{data}} + b_k^{\text{arch}},$$
$$b_k^{\text{data}} = \phi^\star(P_k, A_k) - \phi^\star(\bar{P}, A_k), \qquad b_k^{\text{arch}} = \phi^\star(\bar{P}, A_k) - \bar{\phi}^\star,$$

with $\bar{\phi}^\star = \sum_j w_j \phi^\star(\bar{P}, A_j)$ and $b_k = b_k^{\text{data}} + b_k^{\text{arch}}$, so that $\sum_j w_j b_j = 0$. Client k's local estimate is $\hat{\phi}_k^{\text{loc}} = \phi_k^\star + \varepsilon_k$ with $\mathbb{E}\|\varepsilon_k\|^2 = \sigma^2/n_k$, independent across clients, and the federated estimate averages them.

Proposition 2 (Helps/hurts threshold). *With $w_j = n_j/n$,*

$$\text{gain}_k \;=\; R_k(\hat{\phi}_k^{\text{loc}}) - R_k(\hat{\phi}^{\text{fed}}) \;=\; \frac{a_k}{2}\Big[\; \underbrace{\sigma^2\big(\tfrac{1}{n_k} - \tfrac{1}{n}\big)}_{\textit{variance removed}} \;-\; \underbrace{\|b_k\|^2}_{\textit{heterogeneity bias}} \;\Big], \tag{S7}$$

so federation helps client k iff $\sigma^2(1/n_k - 1/n) > \|b_k\|^2$. Holding $\|b_k\|$ fixed, gain_k *strictly decreases in n_k.*

Proof. Bias–variance expansion of both risks. Local: $\mathbb{E}\|\hat{\phi}_k^{\text{loc}} - \phi_k^\star\|^2 = \sigma^2/n_k$. Federated: the bias is $\|\bar{\phi}^\star + \sum_j w_j b_j - \phi_k^\star\| = \|b_k\|$ using $\sum_j w_j b_j = 0$, and the variance is $\sum_j w_j^2 \sigma^2/n_j = \sigma^2 \sum_j n_j/n^2 = \sigma^2/n$. Subtracting gives Eq. (S7); differentiating in n_k gives the monotonicity.

Note that σ^2 is the *per-sample noise variance* of a client's estimate of the shared parameters, not the dispersion of the client optima; that dispersion is carried by $\{b_k\}$. The first bracket in Eq. (S7) is independent of label skew; the second grows with it through b_k^{data}, and grows with the diversity of the backbone combination through b_k^{arch}. Hence the ordering the main paper tests: Δ is largest at comparable data, falls as distributions diverge, and changes sign at a break-even point, and at fixed data falls as AH grows.

Remark 1 (Where the surrogate stops applying). The surrogate treats a client's deficit as *estimation error* (σ^2/n_k), which sharing can reduce. Under extreme skew a client may hold *zero* samples of some classes; its deficit is then *representational*, no shared conditioner can supply features for classes never observed, and $\|b_k^{\text{data}}\|$ is effectively unbounded. Proposition 2 still predicts no gain, but for a reason outside its own model. This is the regime of the $\alpha_{\text{Dir}} = 0.1$ column, and it is why we do not extrapolate the break-even curve past it.

S5 Per-Client Results

The main paper reports client-averaged accuracy. Because the framework claims a benefit *for each participating site*, we resolve every reported cell to the client level here. The procedure is the one used in the main paper. Each client's own model is scored on the pooled test distribution, at that client's validation-selected round, and the result is averaged over the three paired seeds.

Table S7 gives the six backbone combinations at comparable data. GFL raises accuracy at 25 of the 30 clients and in 63 of the 90 (client, seed) pairs. The five clients that do not improve are spread over four different combinations, and include both the largest backbone in its combination (ResNet18) and the smallest (MLP), so the failures do not concentrate on one family or one capacity.

Table S7. Per-client $\Delta = GFL-$ Individual, pooled-test accuracy, mean over three seeds, for the six backbone combinations of the main paper's Table 3 (PathMNIST, comparable client data). Clients that do not improve are in **bold**.

Backbone combination	AH	client 1	client 2	client 3	client 4	client 5	clients > 0	(client, seed) > 0
5 × CNN	0.00	+.009	+.003	+.004	+.024	+.014	5/5	13/15
4 × CNN + ResNet18	0.11	+.024	+.013	+.004	+.015	**−.018**	4/5	12/15
4 × CNN + Mixer-S	0.30	+.003	+.005	+.013	+.017	+.017	5/5	11/15
CNN + MLP + 3 × Mixer-S	0.36	+.018	**−.011**	+.005	**−.010**	+.034	3/5	9/15
CNN + 2 × VGG9 + 2 × ViT	0.48	+.003	+.001	+.021	+.020	+.003	5/5	8/15
CNN + 2 × ResNet18 + ViT + MLP	0.60	+.006	**−.002**	**−.006**	+.014	+.014	3/5	10/15
Total							**25/30**	**63/90**

Table S8 resolves the break-even grid. The count of clients that benefit rises monotonically with data agreement at *all three* levels of architectural heterogeneity, which is the per-client form of the ordering the main paper reports for the mean, and a stronger test of it: the mean could rise because one client improves a great deal, whereas the count cannot.

Table S8. Break-even grid, per client. Number of clients (of five) whose accuracy GFL raises above Individual training, and the mean Δ, over the AH × DH grid. Lower Dirichlet α means stronger label skew.

		clients improved			mean Δ		
Backbone combination	AH	α=0.1	α=1	α=1000	α=0.1	α=1	α=1000
5 × CNN	0.00	1/5	3/5	5/5	−.011	+.003	+.011
CNN + MLP + 3 × Mixer-S	0.36	0/5	2/5	3/5	−.018	−.004	+.007
CNN + 2 × ResNet18 + ViT + MLP	0.60	1/5	2/5	3/5	−.009	−.008	+.005

S6 The Two Terms of the Break-Even Condition: What We Can and Cannot Measure

Proposition 2 is stated in terms of σ^2 and $\|b_k\|^2$. We report here an attempt to estimate them separately from the trained networks, and explain why that attempt does not succeed at our scale. We include it because the negative result bounds what our evidence supports.

The estimator. Within one run every client starts the shared GAT from the same initialization, so the GL per-client final states ϕ_k are directly comparable. Writing $d_k = \phi_k - \sum_j w_j \phi_j$, the surrogate gives $\mathbb{E}\|d_k\|^2 = \|b_k\|^2 + \sigma^2(1/n_k - 1/n)$, which is exactly the sum of the two quantities whose *difference* Eq. (S7) requires. We calibrate σ^2 on the homogeneous, near-IID reference combination of five identical CNNs at α=1000, where $b_k \approx 0$ by construction. Subtracting the calibrated term then yields $\|\hat{b}_k\|^2$ in every other cell.

The outcome. The calibration is degenerate. The fitted $\hat{\sigma}^2(1/n_k - 1/n)$ exceeds the residual $\|\hat{b}_k\|^2$ by roughly an order of magnitude in *every* cell, so the estimated threshold predicts a positive gain for essentially every client, including the cells where Δ is measurably negative. Its sign agrees with the observed per-client Δ in 134 of 255 client–seed pairs, which is 52.5% and therefore no better than chance, and at the cell level the estimated and observed quantities are *anti*-correlated (Spearman −0.52 over 17 cells). The estimator is therefore uninformative at this scale, for two reasons that we can identify. First, the assumption $b_k \approx 0$ at the reference combination absorbs into $\hat{\sigma}^2$ whatever residual spread the identical-backbone combination has, which inflates that estimate. Second, Euclidean distance in the parameter space of the graph attention network is a poor proxy for excess risk, because the network has permutation and rescaling symmetries that move ϕ without changing the function it computes.

The related equity prediction of Proposition 2, that at fixed $\|b_k\|$ smaller clients gain more, is likewise not resolved. Across the 14 experimental cells in which client sample counts differ by more than 50%, the rank correlation between n_k and Δ_k averages −0.14 and is negative in 8 of 14: the predicted direction, but far from significant, and confounded because $\|b_k\|$ is not held fixed across clients in any of our runs.

Consequences for the claims of the paper. We therefore do not claim to have measured σ^2 and $\|b_k\|$ separately. What we test, and what Secs. 4.2–

4.3 of the main paper support, are the two *orderings* the proposition implies and that are robust to the unknown constants: Δ decreases with label skew at every AH level (Table S8), and the architecture term ζ_{arch} increases with AH (Sec. S7). A direct measurement would require a distance defined in function space rather than in parameter space, and more than three seeds. We regard this as the natural next step.

S7 Gradient-Probe Statistics

Table S9 lists the probe readings behind Fig. 2 (right) of the main paper, and Table S10 lists every correlation coefficient together with an exact permutation p-value. These p-values are obtained by enumerating all $6! = 720$ and all $5! = 120$ permutations respectively, so they are exact rather than asymptotic.

Table S9. Gradient probe over the architecture axis. $\hat{\zeta}_{\text{arch}}$ is the spread of the clients' raw ϕ-gradients at one shared $\hat{\phi}$ on a common data mixture; $\|\bar{g}\|$ is the mean gradient norm.

Backbone combination	AH	$\hat{\zeta}_{\text{arch}}$	$\hat{\zeta}_{\text{arch}}/\|\bar{g}\|$
5 × CNN	0.00	0.050	0.90
4 × CNN + ResNet18	0.11	0.047	0.92
4 × CNN + Mixer-S	0.30	0.060	0.96
CNN + MLP + 3 × Mixer-S	0.36	0.044	1.80
CNN + 2 × VGG9 + 2 × ViT	0.48	0.080	0.98
CNN + 2 × ResNet18 + ViT + MLP	0.60	0.101	1.15

Table S10. AH versus gradient spread: every coefficient. Exact two-sided permutation p-values. "−outlier" excludes the CNN + MLP + 3 × Mixer-S combination, whose mean gradient norm is the smallest of the six and which therefore inflates any ratio.

Statistic	n	Spearman ρ	p	Pearson r	p
$\hat{\zeta}_{\text{arch}}$ vs. AH	6	+0.60	0.24	+0.81	0.053
$\hat{\zeta}_{\text{arch}}/\|\bar{g}\|$ vs. AH	6	+0.83	0.058	+0.33	0.56
$\hat{\zeta}_{\text{arch}}$ vs. AH (−outlier)	5	+0.90	0.083	+0.95	0.025
$\hat{\zeta}_{\text{arch}}/\|\bar{g}\|$ vs. AH (−outlier)	5	+1.00	0.017	+0.88	0.008

Interpretation. On six points, no coefficient computed on the full set clears $p < 0.05$. The raw $\hat{\zeta}_{\text{arch}}$ trends linearly with AH ($r = +0.81$) but its rank correlation is weaker ($\rho = +0.60$), because the AH=0.36 combination reads low. The scale-free ratio trends monotonically ($\rho = +0.83$) but is the statistic

Table S11. Training protocol, identical for Individual, GL, GFL and the three benchmarks except where a method's definition forbids it (the rows describing the shared network do not apply to Individual).

Setting	Value
Federated rounds	50
Local epochs per round	1 (134 minibatches)
Batch size	128
Learning rate (backbone, GNN, adapter)	1×10^{-3}
Learning rate (gate α_k)	1×10^{-2}
Weight decay (backbone)	1×10^{-2}
Gradient clipping	1.0
Warm-up fraction	0.1
Shared GNN	3-layer GAT, $20 \to 64{\times}4 \to 64{\times}4 \to 64$, ELU, dropout 0.1
Shared parameters	≈88k (the only tensor transmitted)
Adapter	$64 \to 128 \to 2d_k$ MLP, private
Gate initialization	$\alpha_k = 0$, one per hooked block
Node-feature refresh	every 100 steps
Model selection	per-client best validation loss
Evaluation split	pooled test set
Seeds	42, 43, 44 (paired across methods)
Public/proxy data	none

most exposed to instability: it divides by $\|\bar{g}\|$, and the AH=0.36 combination has the smallest $\|\bar{g}\|$ of the six, which is why it reads 1.80 against a range of 0.90–1.15 elsewhere. Excluding that one combination, both statistics sharpen substantially and reach significance. We therefore treat *"ζ_{arch} grows with AH"* as a consistent trend across four descriptions of the same six points, and one whose mechanism is interpretable (alike backbones share a gradient direction that averaging reinforces; unalike ones partly cancel), rather than as a hypothesis our sample size can test. Six combinations is the limit of what this evidence can support, and widening the architecture axis is the natural extension.

S8 Training Protocol

Every method in the paper is trained under the identical protocol of Table S11, with the same three seeds $(42, 43, 44)$ and, for a given seed, the same data partition and the same backbone initialization. Model selection is performed per client, at the round with the best local validation loss. The reported accuracy is that checkpoint evaluated on the pooled test set, so a client cannot score well merely by predicting its own majority classes. The centralized upper bound pools all client data and violates the privacy constraint every other method respects; it is not a competing method.

S9 Three Medical Imaging Types

Table S12 gives the full numbers behind Sec. 4.4 of the main paper, where we test whether the break-even prediction transfers beyond histopathology. Each dataset is run at the loss its label balance calls for and at the strongest DH its label structure allows. All three operating points sit at or below the break-even point located in Sec. 4.2 of the main paper, and at all three GFL tracks Individual training within one standard deviation, on two imaging types the decomposition was never fitted to.

The centralized row is reported only as an upper bound. It pools every client's data on one server and therefore breaks the privacy constraint that every other method in the table respects, so it is not a competing method.

Table S12. Three medical imaging types, at the strongest DH each label structure allows: PathMNIST at α=0.3, HAM10000 and MIMIC-CXR at α=1.0. PathMNIST: pooled-test accuracy; HAM10000 and MIMIC-CXR: macro-F1. Mean±sd over three seeds, best per row in bold. *Cent.* pools all data, an upper bound that breaks privacy.

Dataset	Loss	Individual	GL	GFL	*Cent.*
PathMNIST	plain	0.602±.018	0.586±.007	**0.603**±.015	0.873
HAM10000	balanced	**0.580**±.065	0.543±.033	0.573±.055	0.710
MIMIC-CXR	balanced	0.589±.055	0.587±.050	**0.592**±.061	0.723