Measuring attribution faithfulness on cell graphs with known drivers

Tanmay Hinge

August 2026

Abstract

Interpretability methods for graph neural networks on tissue are usually evaluated on real specimens, where nobody knows which cells actually drive the outcome. Faithfulness gets asserted rather than measured. This paper describes a synthetic testbed where the drivers are known by construction, and reports what five attribution methods do in it. The generator places cells with controlled types, marker vectors and spatial organisation; the tissue-level label is an explicit function of the recorded cell table; and each sample carries a verified irredundant counterfactual driver set, a set of marker-identical but causally inert decoys, and a signed per-cell relevance vector. The headline finding is about the measuring instrument. The standard model-randomisation control returns either verdict for GNNExplainer, rank correlation 0.89 or 0.29, purely as a function of how long the explainer's own mask was optimised, with the common 100-epoch default sitting inside the unconverged region. Any published use of that control on a mask-based explainer must therefore report its optimisation budget, and results at or below that default should be treated as measuring the explainer's initialisation rather than its relationship to the model.

Against the testbed's ground truth, which method ranks highest depends on the metric and, more sharply, on whether the two label classes are pooled. They should not be: the positive and negative classes pose structurally different targets, 92% of the variance of a pooled paired difference lies between them, and pooling averages two large opposite effects into a null. Integrated gradients beats a model-free marker heuristic by $+0.302$ $[+0.275, +0.330]$ average precision on positive samples and loses by -0.237 $[-0.253, -0.220]$ on negatives; weighted to the test split's own class balance it wins overall, $+0.018$ $[+0.003, +0.034]$. On the positive class, the one where the ground truth is a genuine minimal sufficient cause, the heuristic scores a third of chance. GNNExplainer, run here with its edge mask disabled so that only the node-feature half of the method is measured, sits below the random floor on the counterfactual driver metric, by -0.011 $[-0.015, -0.007]$ clustered by sample, but not on the constructive one, nor on signed correlation, decoy mass, or the deletion and insertion curves. And no learned method approaches the structure-recovery reference of 0.259, which is not 1 because the recorded counterfactual driver set is one arbitrary member of a family averaging six times its size.

Two further results bound what the testbed can show. Removing the subspace that carries driver membership from the frozen encoders leaves accuracy intact against a variance-matched control, while removing the subspace carrying *engagement* collapses it below chance: the models represent the property their label is a function of, and do not separably represent the constructive driver set, which is the one they have most reason to encode. And a second label family, built around counting a three-cell motif to escape the one-hop character of the first, proved unlearnable by both architectures. The most likely explanation is an expressiveness limit rather than a data problem, though the decisive test is not run here. Attribution results in this paper are therefore confined to a label whose target property is computable by a single untrained message-passing step.

Claims made during this work that did not survive re-measurement are reported in Section 9 rather than tidied away.

1 Introduction

A cell graph is a natural representation for tissue: nodes are cells with measured marker intensities, edges connect cells that are physically close, and a graph neural network predicts something about the specimen. When such a model predicts well, the next question is always which cells it used. Attribution methods answer that question with a score per cell, and those scores get read as biology.

The difficulty is that on real tissue there is no answer key. If a method highlights a cluster of T cells at a tumour margin, that looks plausible, and plausibility is generally where the evaluation stops. A method can be plausible and wrong in two distinct ways: it can highlight cells that are not causally involved, or it can produce the same highlights regardless of what the model learned. Neither failure is visible without ground truth.

This paper takes the other route: build tissue where the answer is known, then measure. The generator described in Section 3 produces synthetic specimens in which the label is a stated function of the cell table, so the cells responsible for it can be computed exactly and recorded. Models trained on that data solve the task essentially perfectly, which removes model error as a confound: wherever an explanation is wrong here, the model was still right.

Contributions.

1. Evidence that the widely used model-randomisation control is sensitive to the explainer's own optimisation budget, to the point of reversing its verdict, with the common default sitting inside the unconverged region. This is the finding with the widest reach: it is a statement about a tool the subfield treats as a settled sanity check, it is independent of this testbed, and it implies that published results running that control at or below the default may have measured their own budget.

2. Three measured limits on what ground-truth attribution benchmarks can establish, each of which applies beyond this testbed. The recorded counterfactual driver set is one arbitrary member of a large equivalence class, which puts the score of a structure-recovering estimator at 0.267 rather than 1. That is a reference level and not a cap: a method that grades within the class can and does exceed it. The trained models do not separably represent the constructive driver set, though they demonstrably represent and use the property their label is a function of. And a label family built to be harder proved unlearnable, so the benchmark's difficulty is bounded by model expressiveness rather than by ground-truth design.

3. A tissue generator and label oracle in which, for every sample, four ground-truth cell sets and a signed per-cell relevance vector are recorded and exactly recomputable from the serialised record. The counterfactual driver set is verified: removing it flips the label, and no leave-one-out subset does. Decoy cells are marker-identical to real drivers and verifiably inert, so highlighting everything of the right cell type is a losing strategy.

4. An audit of the generator itself. In the first version, a model given only CTL positions with all tumour information withheld separated the classes at AUC 0.928, through a local-density artefact that every non-spatial test passed. What it takes to close that hole is reported, because a benchmark with a shortcut measures the shortcut.

5. Measurements for five attribution methods against five model-free baselines and one oracle-informed reference, under three controls, one of them run in two modes, on models that reach test AUC ≥ 0.999, with bootstrap intervals on every comparison. Which method ranks highest depends on the metric, and on whether the two label classes are pooled. They should

not be. GNNExplainer, measured with only its node-feature mask, scores below the random floor on both average-precision metrics but not on the others tested, and no learned method reaches the structure-recovery reference.

Everything is reproducible from code and a fixed seed. Each stage of the work was gated by an acceptance test suite written before the implementation; the suite currently stands at 93 tests across five closed stages.

2 Related work

The setting is a real one. Graph neural networks over cell graphs are used to predict clinical outcomes from multiplexed tissue imaging, and the subgraphs they highlight are read as spatial motifs with biological meaning: Wu et al. [17] model tumour microenvironments as local cell subgraphs across head-and-neck and colorectal specimens and report motifs associated with recurrence and survival. That is exactly the inference this paper is trying to put a number on, and exactly the setting where no answer key exists.

The control that does most of the work in this paper is the model-parameter randomisation test of Adebayo et al. [2], which asks whether an explanation changes when the model's weights are replaced with random ones. A method whose output is unchanged is not describing the model. That work found several popular saliency methods insensitive to randomisation in the image domain; this paper applies the same idea to graphs and finds the verdict depends on an implementation detail of the explainer being tested.

The methods under test are GNNExplainer [19], which learns a soft mask over edges and over node features by maximising mutual information with the prediction; integrated gradients [16], which integrates input gradients along a path from a baseline; plain input gradients [15]; and attention rollout [1], adapted from transformers to a graph attention encoder. The encoders are GIN [18] and GATv2 [5].

What is not tested, and why it matters for how the results are read. GNNExplainer is one member of a family. PGExplainer [11] learns a shared parameterised mask generator rather than optimising a mask per instance, GraphMask [14] learns differentiable edge gates, and SubgraphX [20] searches subgraphs with Monte Carlo tree search and scores them by Shapley value. None of the three is evaluated here. Any claim in this paper is therefore about GNNExplainer specifically, not about mask-based explanation in general, and the budget-dependence result in Section 7.8 is a reason to expect the others to need the same scrutiny rather than evidence that they fail. The more conspicuous omission is CF-GNNExplainer [10], which searches directly for minimal perturbations that change the prediction. The ground truth in this testbed is counterfactual, so a counterfactual explainer is the natural fit, and its absence is a genuine gap rather than a scoping choice.

Prior benchmarks with synthetic ground truth, and a published objection to the whole approach. Sanchez-Lengeling et al. [13] is the closest antecedent in experimental shape: it observes that synthetic graph problems can have computable ground-truth attributions and uses them to evaluate attribution methods quantitatively along axes of accuracy, stability, faithfulness and consistency, concluding with recommendations about which methods to use. GraphXAI and its generator are the most developed synthetic-ground-truth generator [3]. GraphFramEx [4] proposes a systematic evaluation framework and, relevantly here, classifies explanations by whether they are *sufficient* or *necessary*, a distinction that maps onto the two driver sets this paper reports, the constructive set being the sufficient one and the counterfactual set the necessary one. That vocabulary is adopted where it fits. GraphFramEx scopes itself to node classification, whereas the

task here is graph classification, so its conclusions do not transfer directly. Cell-graph explainability has been evaluated in computational pathology [9].

The testbed here differs from these in two respects that turned out to matter. Its ground truth is *counterfactual* and verified per sample, rather than a planted motif assumed to be the explanation; and it deliberately includes decoy cells that are statistically identical to real drivers in everything but position, so that "highlight everything that looks like a T cell" is a losing strategy rather than a winning one.

An objection this paper has to answer. Faber et al. [7] argue that comparing explanations to planted ground truth is the wrong evaluation altogether, because the trained GNN need not use the ground-truth substructure, so an explanation method cannot be faulted for failing to recover it. That objection applies to this paper and it is not rhetorical: Section 5 measures it directly and finds it holds here. Ablating the subspace that carries driver membership leaves the models' accuracy intact, while ablating the subspace carrying engagement collapses it. The models represent what their label is a function of and not the recorded driver set, exactly as Faber et al. predict.

Two things follow, and they shape how this paper is organised. First, the driver-recovery numbers in Section 7 inherit that limitation and are reported as recovery of causes rather than as faithfulness. Second, and this is the reply, the control suite does not inherit it. A method whose output is unchanged when the weights are randomised is not describing the model under any reading, whatever the ground truth is or whether the model uses it. That is why the randomisation and untrained-encoder results, not the driver-recovery table, carry this paper's strongest claims.

The evaluation philosophy borrows from ROAR [8], which showed that attribution rankings can look good for reasons unrelated to the model, and from Rudin's argument that post-hoc explanation of an opaque model is a weaker guarantee than it is usually taken to be [12]. Neither of those requires ground truth; this work asks what changes when ground truth is available.

3 The testbed

This section describes each part of the testbed and, where a design decision was forced by something that went wrong, says so.

3.1 Generating tissue

A sample is a square field of $1000 \times 1000\,\mu$m containing roughly 1500 cells of four types, namely tumour, cytotoxic T lymphocyte (CTL), macrophage and stroma, in proportions 0.30/0.20/0.15/0.35. Each cell carries a six-dimensional marker vector drawn from a type-specific mean with Gaussian noise ($\sigma = 0.15$). Stroma is deliberately given a flat, mid-level profile with no strong channel, so that it is separable by the *absence* of a positive marker rather than by the presence of one, because that is what real antibody panels look like. Its implemented mean is uniform at 0.2 to 0.3 across all six channels, which is higher than the other three types' off-channels rather than lower.

Two rules in the generator exist to prevent specific failures.

Composition is fixed before the class is consulted. The total cell count and the per-type counts are drawn from a class-independent distribution, and only *placement* depends on the target class. Class-conditional count and marker marginals therefore match by construction, and no model can read the label off a histogram of cell types.

Cells cannot overlap. Every position is subject to a hard-core minimum separation of $8\,\mu$m, enforced by rejection sampling against a KD-tree. Candidates outside the field are rejected and resampled, never clipped to the boundary. This was added retroactively; Section 9 explains what it fixed.

3.2 The label

Write T for the tumour cells and C for the CTLs in a sample. A tumour cell is *engaged* if a CTL lies within r_{label} of it:

$$\text{engaged}(t) \iff \exists\, c \in C : \|t-c\| \le r_{\text{label}}, \qquad \text{score} = \frac{|\{t \in T : \text{engaged}(t)\}|}{|T|}, \qquad y = \mathbf{1}[\text{score} \ge \tau].$$

The defaults are $r_{\text{label}} = 30\,\mu$m and $\tau = 0.25$. Over 2000 samples the class balance is 0.472. The margin $|\text{score} - \tau|$ is drawn from a controlled range so that samples are not trivially far from the decision boundary.

One implementation rule matters more than it looks. A single function, `oracle.evaluate`, is the only code in the repository that computes a label, a score or a driver set. The generator calls it; graph construction calls it; the tests call it. A second implementation, even a faster inline one, would be a specification violation, because the entire project rests on the recorded ground truth being the same object that everything downstream is scored against.

3.3 What counts as a driver

Four cell sets and one per-cell vector are recorded for every sample.

Constructive drivers are the engaged tumour cells together with the CTLs that engage them. These are the cells that participate in making the label what it is.

Unengaged tumour cells are counter-evidence, not decoys. Because the score is a *ratio*, an unengaged tumour cell sits in the denominator, and removing it *raises* the score. Grouping them with irrelevant cells, which an early version of the design did, would have penalised an explainer for the correct behaviour of assigning them negative attribution.

Decoy CTLs are CTLs with no tumour cell within r_{label}. They are drawn from the same aggregate process as enabling CTLs and are marker-identical to them; they are also verifiably inert, in that removing all of them changes neither the score nor the label. They are what makes "highlight everything that looks like a T cell" fail.

Signed relevance assigns $+1$ to engaged tumour and enabling CTLs, -1 to unengaged tumour, and 0 to everything else. This definition is class-independent, so a signed attribution is directly comparable against it for positive and negative samples alike, with no per-class sign flipping anywhere in the protocol.

The counterfactual set is the one that carries the headline metric. It is an explicitly verified set whose removal flips the label, and which is irredundant: no leave-one-out subset flips it. Since removal is monotone in both branches, irredundancy implies that no proper subset flips the label either. Its median size is 62 cells out of roughly 1500.

The asymmetry is worth making concrete. In one positive sample, removing the 20 CTLs in its counterfactual set disengages enough tumour cells to drop the score from 0.337 to 0.249, just under τ. In one negative sample, removing 145 unengaged tumour cells shrinks the denominator and lifts the score from 0.169 to exactly 0.250, and ties count as positive, so the label flips. Positive samples flip by removing CTLs; negative samples flip by removing unengaged tumour cells. A heuristic tuned to one class is therefore necessarily anti-correlated on the other, which is why Section 7 reports average precision, signed rank correlation and decoy mass together instead of picking one.

One honest limitation. For negative samples the counterfactual set is cardinality-minimal, because the required count is exact. For positive samples it is irredundant but not proven minimal, because finding the smallest CTL set that disengages enough tumour cells is a set-cover problem. Irredundancy is what the faithfulness metrics need, and it is guaranteed; minimality is claimed only where it holds.

3.4 How arbitrary is the recorded set?

Irredundancy does not imply uniqueness, and the gap between the two turns out to be wide enough to bound what the primary metric can measure. This subsection quantifies it. The two label classes behave very differently.

Negative samples: every valid set is equally valid. The score is a ratio, so removing an unengaged tumour cell leaves the numerator alone and shrinks the denominator. Every k-subset of the unengaged pool therefore produces an *identical* new score: all $\binom{|\text{pool}|}{k}$ of them flip the label, and all are cardinality-minimal. The oracle records one of them, the lowest-`cell_id` slice. Drawing random k-subsets and passing the ablated table back through `oracle.evaluate`, all 1160 of 1160 trials flipped the label, with no exceptions. The mean recorded set holds 125 cells drawn from a mean pool of 367.

And the recorded one is not a neutral choice. `cell_id` is blocked by cell type and, within a type, follows placement order, which is aggregate-clustered: tumour cells in the first quartile by `cell_id` have mean radial dispersion 161 μm and in the last quartile 110 μm, against 355 μm for a random quarter. Taking the lowest-`cell_id` slice therefore selects a spatially compact set. Compared against random equally-valid sets of the same size, the recorded set is more compact in 56 of 58 negative samples, mean dispersion 147 μm against 280 μm. Section 10 states the consequence: the contiguity of the target is *forced* by geometry for positive samples but is an artefact of a tie-break for negative ones, and any method with a smoothness prior collects an unearned advantage on the latter.

Positive samples: the greedy finds a different answer nearly every time. Randomising the selection inside the same greedy, using a weighted draw in place of the argmax and leaving the committed leave-one-out pruning unchanged so that every set returned is irredundant by the same test the generator applies, produces 31.9 distinct sets from 32 draws on average. Mean Jaccard against the recorded set is 0.27; the recorded set itself was recovered in 1 of 42 positive samples. The union over the family averages 78 cells against a recorded 13.

The consequence for the metric is direct, and it is the origin of the 0.267 ceiling quoted in Section 6. An attribution that identifies the causal structure correctly, meaning every cell participating in some minimal sufficient set, is scored against one arbitrary member of that structure, and is penalised for the rest. This compresses the differences *between* methods and it is why a floor alone cannot calibrate the results. It does not explain a method scoring below the floor, because the random baseline is scored against the same single representative.

3.5 Auditing the generator for shortcuts

A benchmark is only as good as the absence of ways around it. Three probes are run against the finished dataset, and their results bound what any faithfulness number can mean.

A *non-spatial* probe, using cell counts and marker moments with all geometry discarded, reaches AUC ≤ 0.55. There is no composition shortcut.

A *CTL-geometry-only* probe, given the positions of CTLs with every tumour cell withheld, reaches AUC 0.657. This is above chance and is expected to be: covering more tumour requires more CTLs in a contiguous region, so CTL clustering is inherently informative about the label. What matters is that it is 0.657 and not 0.928, which is where the first version of the generator sat (Section 9).

A *global spatial statistic*, per-type-pair nearest-neighbour distances, reaches 0.706. The label *is* a bivariate spatial statistic, so this must be above chance.

The claim this testbed can test is therefore narrow and should be stated narrowly: that the specific driver *cells* are recoverable, not that no aggregate statistic correlates with the label. That is precisely why the baselines in Section 6 include nearest-CTL distance and local CTL density: they exploit exactly these residuals, so a method that merely rediscovers them cannot look faithful by accident.

4 Models

4.1 Graphs, images, and one split

Each sample's cell table is turned into a radius graph: nodes are the detected cells, node features are the six marker channels and nothing else, and edges join cells within r_{graph}. Cell type is recorded as ground truth for probing but is never a model input. Four radii are used, parameterised as the ratio $r_{\text{graph}}/r_{\text{label}} \in \{0.5, 1.0, 1.5, 2.0\}$. The ratio changes the graph a great deal: measured over 25 whole graphs, mean degree runs 1.1, 5.1, 11.5 and 20.1 across the four radii, and the fraction of isolated nodes falls from 37% at ratio 0.5 to zero at ratio 2.0.

Two architectures are trained: GIN with edge features (46,593 parameters) and GATv2 (34,561 parameters), both three layers with hidden width 64, mean-plus-max pooling and a linear head. A convolutional baseline gives a non-graph comparison on the same data: each sample is rendered to a 384^2 image with six channels, one per marker, using Gaussian kernels at $\sigma = 5\,\mu$m, and a four-block CNN (617,985 parameters) is trained on it. The CNN gets roughly 13 to 18 times the parameter budget, deliberately: images need more capacity than graphs built from the same cell table, and handicapping the baseline would make a "graphs win" result meaningless.

All models use one fixed split of 1400 training, 300 validation and 300 test samples, stratified jointly on label and margin tertile, loaded from disk rather than recomputed. Every checkpoint records a fingerprint of the dataset it was trained on, and a check script refuses to let stale numbers be quoted after the data is regenerated.

4.2 Accuracy

Table 1 gives the result: everything solves the task. A logistic regression on a single hand-built feature, the fraction of marker-classified tumour cells with a marker-classified CTL within $30\,\mu$m, also reaches test AUC 1.000, confirming the task is fully solvable from graph-visible information and that the models are not doing anything exotic.

Table 1: Test AUC. Every model has essentially solved the task, which is the precondition for measuring attribution without a model-error confound.

	$r_{\text{graph}}/r_{\text{label}}$				
model	0.5	1.0	1.5	2.0	parameters
GIN	0.9992	0.9999	0.9998	0.9993	46,593
GATv2	0.9991	1.0000	0.9999	0.9996	34,561
CNN	0.9998 (rendered image, no graph)				617,985

4.3 Training controls

Three corruptions establish that the models use what they are supposed to use (Table 2). *Marker shuffle* permutes marker vectors across cells within a sample; *position shuffle* permutes positions across markers; *label shuffle* permutes the target within each split.

Table 2: Test AUC under training-time corruptions, at $r_{\text{graph}}/r_{\text{label}} = 1.0$. The point-pattern floor is what a model given *only* positions can reach, measured with eight hand-built summary features. It is a level the point pattern *affords* rather than a floor the controls must clear: GIN finds most of it and exceeds the summary-feature measurement at 0.698 and 0.703, while GATv2 lands below it at 0.545 and 0.518. That spread is optimisation variance, not a difference in what the two controls test. Neither can be expected to fall to chance, because both preserve the point pattern exactly. Label shuffle destroys the target itself and falls to near chance: 0.468 to 0.576, against an AUC standard error of about 0.033 at $n = 300$.

corruption	GIN	GATv2	CNN
none	0.9999	1.0000	0.9998
marker shuffle	0.698	0.545	0.713
position shuffle	0.703	0.518	not run
label shuffle	0.576	0.468	0.497
measured point-pattern floor	0.608 (graph domain)		0.753 (image domain)

Two remarks on how these are scored. First, marker shuffle and position shuffle are the *same* experiment: both break the correspondence between what a cell is and where it is, while leaving both marginals and the point pattern intact. They are replicates, not independent evidence, and are reported as such. Second, they are scored as a gap from the intact model (0.9999 against roughly 0.70) rather than against an absolute ceiling. An earlier version of the gate used an absolute ceiling derived from the point-pattern floor, and that was wrong in principle: the floor is measured with eight hand-built summary features, which is a lower bound on what the point pattern affords, and a GNN over the same points legitimately extracts more.

4.4 A result about graph radius that contradicted the prediction

Ratio 0.5 was expected to be hard. At $r_{\text{graph}} = 15\,\mu$m the graph does not percolate: mean degree 1.1, and 37% of nodes isolated. The written prediction was that a GNN there would be little more than a per-node MLP.

Both architectures reach 0.999. The prediction was wrong for a reason that, once stated, is obvious: the label is a *local* property and the readout is *global pooling*. A tumour cell only needs its CTL as a direct neighbour, and CTL aggregates are tight enough ($\sigma = 18\,\mu$m) that a $15\,\mu$m radius still catches them. Connectivity is never required.

This is worth recording because near-perfect accuracy on a graph that is over one-third isolated nodes is exactly the setting where an explanation has almost no structure to travel through while the model looks entirely healthy.

It also has an uncomfortable implication for the framing of this paper, which is worth confronting rather than filing as a curiosity. If connectivity is never required and the label is a one-hop property recoverable by global pooling, then the graph is doing less work here than the phrase "cell graph benchmark" suggests: a per-node classifier with a pooled readout would suffice, and the message-passing structure is incidental to the task rather than constitutive of it. Section 8 was an attempt to build a label where that is not true, and it did not produce a learnable one. Attribution results

in this paper should therefore be read as results about explaining a model that happens to be a GNN, not about explaining graph structure.

5 Probing the frozen encoders

Before measuring attribution it is worth asking what the encoders actually represent. Linear probes are fitted on the validation split and scored on test; the training split is never touched. Each probe is reported against two floors: the same probe on a randomly initialised encoder of the same architecture, and on raw marker vectors.

Table 3: Linear probe results at $r_{\text{graph}}/r_{\text{label}} = 1.0$, reported as trained / untrained-encoder floor. `is_engaged` is the property the label is a function of. It is *not* the target Section 7 scores attributions against, which is the counterfactual and constructive driver sets, and the distinction turns out to matter (see below).

probe	metric	GIN	GATv2	raw markers
`is_engaged` ($r_{\text{graph}}/r_{\text{label}} = 0.5$)	AUC	0.881 / 0.831	0.867 / 0.865	0.502
`is_engaged` ($r_{\text{graph}}/r_{\text{label}} = 1.0$)	AUC	1.000 / 1.000	1.000 / 1.000	0.502
`is_engaged` ($r_{\text{graph}}/r_{\text{label}} = 2.0$)	AUC	0.999 / 0.998	0.998 / 0.997	0.502
`is_driver`	AUC	0.998 / 0.996	0.999 / 0.998	0.784
`graph_score`	R^2	0.924 / 0.768	0.963 / 0.873	−0.130

The headline in Table 3 is that `is_engaged`, meaning "does this tumour cell have a CTL within r_{label}", is linearly decodable at AUC 1.000 from an *untrained* encoder at ratio 1.0. Training adds nothing because there is nothing left to add. At that ratio $r_{\text{graph}} = r_{\text{label}}$, so "is there a CTL within r_{label}" is literally "is there a CTL among my graph neighbours", which any message-passing layer computes whether or not it has been trained.

The gap appears only where the property stops being one-hop. At ratio 0.5, where r_{graph} is half of r_{label}, training buys +0.050 on GIN. Where training genuinely helps everywhere is `graph_score`, the pooled ratio itself: +0.09 to +0.16. Training taught the readout, not the node representation.

Two setup checks behaved as intended and are worth stating because they license the above. `is_engaged` from raw markers alone is 0.502, exactly chance, which confirms the extraction pipeline leaks no spatial information. A control task with random per-cell labels sits at or below chance for every probe, confirming that a 64-dimensional linear probe cannot memorise roughly 225,000 labels.

Decodable is not the same as used, and not the same as prominent. Two limits on how far Table 3 can be pushed, both measured rather than asserted.

First, a probe recovers a property from wherever it lives, however little of the representation that is. Fitting a probe for `is_driver` and measuring the variance of the node embeddings along the direction it finds, that direction accounts for as little as 0.0001% of total embedding variance on GIN, whose embeddings carry total variance around 1.3×10^5 against GATv2's 7. A property can be linearly decodable at AUC 0.998 and still occupy a corner of the space the representation barely varies along. High probe AUC licenses "the information is present", not "the information is prominent".

Second, presence is not use. To separate them, the probe direction can be removed and the ablated embeddings pushed through the model's own pooling and head. Removing a subspace of rank k damages the model whatever the subspace means, so the comparison must be against a

subspace of the same rank *and the same captured variance*; against a merely random subspace the result is dominated by how much of the representation was destroyed. Matched that way, removing up to 48 of 64 dimensions carrying `is_driver` changes test AUC no more than removing an arbitrary subspace of equal rank and variance, with an excess AUC drop between −0.19 and +0.04 across both architectures, all three radii and every rank tested.

How well the matching actually succeeded is worth reporting rather than assuming, since the whole comparison rests on it. Across the 72 cells, 69 fall within a factor of 1.25 of their target's captured variance, and every cell at rank 16 or below matches to within 0.82 to 1.16. Three fail, all at rank 48: `is_engaged` on GIN at ratio 2.0 (ratio 0.20), `is_engaged` on GIN at ratio 1.0 (0.32), and `is_driver` on GIN at ratio 2.0 (0.57). Two of those three are `is_engaged` cells whose mismatch *inflates* the effect this section relies on, so the failures cut against the argument rather than for it, and the rank-48 `is_engaged` numbers should be read with that in mind. Restricting the `is_driver` range to well-matched cells leaves it unchanged at −0.19 to +0.04. Removing the `is_engaged` subspace is a different matter: at rank 32 it drops GIN from 0.9990 to 0.2863, below chance, where the matched control leaves it at 0.9893, an excess of +0.70. That cell is one of the well-matched ones, at a variance ratio of 0.96. The same sign appears in all six architecture-radius combinations.

The model, in other words, represents and uses *engagement*, which is the property its label is a function of. It does not encode membership of the *constructive* driver set, every engaged tumour cell together with every enabling CTL, as a separable quantity.

The probe target needs stating precisely, because the natural reading is the wrong one. `is_driver` is membership of the *constructive* set, not of the counterfactual set that carries the primary metric. That makes the result harder to explain away rather than easier. The counterfactual set is one arbitrary member of a large family (Section 3.4), so a model would have little reason to encode it; the constructive set is canonical, it is exactly what an engaged *fraction* is computed from, and Section 7 argues the model has every reason to represent it. It does not represent it separably, and that is the finding.

Two limits on how far this can be pushed. The two ablation targets differ in scope as well as in definition: `is_driver` is fitted over all cells and counts enabling CTLs as positives, while `is_engaged` is fitted over tumour cells only. So the contrast is not a clean single-variable comparison, and a scope-matched arm is needed before the asymmetry is fully earned. And removing 48 of 64 dimensions leaves accuracy near-intact whatever is removed, so this instrument has limited power to detect use of *any* property: the correct reading of the `is_driver` result is "not detectable by this test", not "demonstrably absent". Section 10 returns to what this costs the primary metric.

What this forces on the next section. If an attribution method "recovers the drivers", it may be reading a property present in *any* GNN over this graph, including one with random weights. Recovering it would then say nothing about the trained model. The randomisation control is therefore not a footnote here; it is the primary result, and ratio 1.0 alone cannot carry a headline.

6 The faithfulness protocol

6.1 Methods, baselines and metrics

Five distinct attribution computations are tested: GNNExplainer, integrated gradients with a zero baseline, integrated gradients with the dataset marker mean as baseline, plain saliency, and attention rollout (GATv2 only). The project specification names a sixth, *CNN integrated gradients*, but on the CNN that is the zero-baseline variant under another name: pixel integrated gradients mapped back to cells through the renderer, the same code path, producing bit-identical values.

It is not reported as a separate row, because carrying it through the results tables presented five methods as six. GNNExplainer's configuration needs stating in full, because the results depend on it and two of its settings are not the canonical ones. It is PyG's implementation with `node_mask_type="attributes"` and `edge_mask_type=None`: a mask is learned over node features only, and *no edge mask is learned at all*. A cell's score is the sum of its learned feature mask across the six marker channels, which PyG postprocesses through a sigmoid, so the output is non-negative by construction. Every other hyperparameter, the mask size and entropy regularisation coefficients that shape how sparse the learned mask becomes, is left at the library default and was not swept; only the epoch budget was (Section 7.8).

Both facts bound the claims below. Section 2 describes GNNExplainer as learning a mask over nodes *and* edges, which is its formulation; what is measured here is the node-mask half. And since the regularisers control mask sparsity, and average precision is a ranking metric sensitive to sparsity, the possibility that a different regularisation setting would move GNNExplainer's score is open. Given that Section 7.8 exists precisely to show one explainer hyperparameter can reverse a verdict, leaving the other two unswept is an asymmetry worth naming rather than hiding: the results below are about GNNExplainer as configured here, not about the best GNNExplainer obtainable.

All methods are computed with respect to the positive-class logit and kept *signed*, never absolute. This convention matters: an unengaged tumour cell should receive negative attribution, and a protocol that scored |attr| against a binary mask would mark that correct behaviour as wrong.

Five *model-free* baselines are run through the identical scoring path: uniform random; node degree; *tumour-likeness*, the projection of a cell's marker vector onto the tumour mean; distance to the nearest CTL, the strongest domain heuristic; and local CTL density, which exploits the residual measured in Section 3.5. These are model-free in that they consult no trained network. Three of them are not, however, computable from observations alone: distance to nearest CTL and local CTL density are built from the oracle's `type_id` column, and tumour-likeness projects onto the generator's true tumour prototype rather than an estimate of it. Cell type is recoverable from the six markers at very low error here, so the practical inflation is small. But they are *oracle-typed* heuristics rather than observable ones, and Section 9 records why that distinction is worth keeping straight.

A sixth estimator, the *smooth blob*, is reported alongside them but is marked separately throughout because it is not model-free in the same sense. It is a disc around the centroid of the *true* driver region, so computing it requires the answer key. It measures how much credit a pure spatial-smoothness prior earns given that driver sets here are contiguous, which is worth knowing, but it is oracle-informed and must not be counted among the baselines a learned method is asked to beat.

Three metrics are reported together. *Average precision* against the counterfactual driver set is the primary number, and it is bracketed by two references rather than one. The floor is the measured random baseline, 0.058, which is close to but not identical with driver prevalence, 0.052. The upper reference is 0.267, not 1: Section 3.4 shows the recorded counterfactual set is one arbitrary member of a large equivalence class of equally valid flipping sets, so an estimator that identifies the causal structure but ranks every member of it equally scores only 0.267 against the single member that happened to be recorded. That is a reference level and not a bound. A method that grades *within* the class can exceed it, and integrated gradients does so on positive samples (Section 7.2). Numbers in Section 7 should be read against that interval with those caveats attached. *Signed Spearman correlation* against the three-valued relevance vector captures whether a method gets the direction right. *Decoy mass*, the share of total attribution magnitude landing on decoy CTLs, is the discriminative one, since decoys are marker-identical to enabling CTLs and causally inert.

Reporting three together is a convergent-validity design, and decoy mass in particular is a discriminant-validity instrument: it exists to catch estimators that cannot separate causally active from causally inert cells with identical markers. Naming the structure is not decoration: Section 7.3 resolves an apparent contradiction between the metrics by exactly the argument a discriminant-validity analysis calls for, and the framework transfers to any setting where attributions are scored against a known target, graph or otherwise.

6.2 Controls

Each control rules out a different way for a method to look faithful without being faithful. *Weight randomisation* replaces the trained weights and asks whether the attribution changes; it comes in two modes, *cascading*, which reinitialises the classifier head together with the last convolutional layer and leaves the earlier layers trained, and *full*, which reinitialises every parameter.

Two clarifications on that naming, both of which matter because this paper's headline concerns misuse of this control. First, the cascading mode reinitialises the head *and* the last convolutional layer, not the head alone. Second, "cascading randomisation" in Adebayo et al. [2] names a different procedure: randomising successively from the top layer downwards and reporting the whole trajectory, alongside an *independent* randomisation that perturbs one layer at a time. What is run here is the two endpoints of that trajectory, a partial top-down perturbation and a complete one, with the intermediate depths omitted for compute. The mode names are retained for continuity with the released artifacts, but they should be read as a two-point reduction of Adebayo et al.'s test rather than as the test itself. *Label randomisation* attributes on an encoder trained against shuffled targets and scores it against the true drivers, so a method that still finds them is reading the data rather than the model. *Untrained-encoder attribution* scores a randomly initialised encoder directly, and given what Section 5 found it is the sharpest of the three here: the driver property is decodable at AUC 1.000 before any training, so a method could score well against ground truth while describing nothing the model learned.

Rank correlation above 0.5 between the attributions on the trained and randomised networks is recorded as a *void* verdict for that setting. The threshold is a convention, so its effect is reported rather than assumed: sweeping it over 0.4, 0.5 and 0.6 moves GNNExplainer's void count from 11 to 10 to 8 of 12 settings, attention rollout's from 4 to 3 to 3 of 6, and leaves every gradient method at 3 or fewer of 14. The qualitative reading, GNNExplainer void in a clear majority of settings and gradient methods in a small minority, is stable across that range; the exact counts are not, and should be quoted with the threshold attached.

Sample sizes differ by experiment and are collected here rather than scattered: Table 4 and Table 6 use 100 test samples; the controls in Table 8, Table 9 and Table 10 use a deterministic 50-sample subset of those; the budget sweep in Table 11 uses 25 graphs; and the seed replication uses 30. GNNExplainer is optimised for 400 mask epochs throughout (Section 9.5). Section 7.8 explains that choice at length.

7 Results

7.1 Driver recovery

Table 4 is the central measurement, and Figure 1 shows it pooled across settings. Two things read straight off it.

GNNExplainer (node-feature mask) performs below random on the counterfactual metric. At every radius, on the metric that scores recovery of the cells whose removal actually

Table 4: Average precision against the two driver sets, GIN and GATv2 only. The first three columns are point estimates at $r_{\text{graph}}/r_{\text{label}} = 0.5$, 1.0 and 2.0 against the verified counterfactual set. The pooled column is the mean of those three, with a 95% bootstrap interval over the 100 test samples; baselines are identical across encoders, so they contribute one value per sample rather than six. Rows are ordered by the pooled value. **Read the counterfactual column against the interval** $[0.058, 0.267]$**, not against** 1: the floor is the measured random baseline, and the upper figure is a *reference level* rather than a bound. It is what an estimator scores when it identifies the causal structure but ranks every member of it equally, so a method that grades within the family can exceed it (Sections 3.4 and 7.2). **This column pools the two label classes and must be read alongside Table 5, which does not**: 92% of the variance of a pooled paired difference lies between classes. **The two metric columns have different floors and must not be compared across the divide**: 0.058 against the counterfactual set, 0.118 against the constructive set. Baselines, which consult no model, are marked; the smooth blob is marked separately because it consults the ground truth (Section 6) and is not a baseline any method is asked to beat. The GNNExplainer row here, and in every later table, measures its node-feature mask alone with the edge mask disabled (Section 6); read it as a result about that configuration rather than about the method as its authors define it.

	vs. counterfactual set				vs. constructive
method	0.5	1.0	2.0	pooled, 95% CI	pooled
ceiling: causal structure recovered	n/a	n/a	n/a	*0.267*	n/a
tumour-likeness *(baseline)*	0.169	0.169	0.169	0.169 [0.140, 0.198]	0.197
integrated gradients (mean baseline)	0.116	0.205	0.156	0.159 [0.133, 0.187]	0.482
integrated gradients (zero baseline)	0.107	0.172	0.162	0.147 [0.123, 0.172]	0.573
smooth blob *(oracle-informed)*	0.099	0.099	0.099	0.099 [0.088, 0.112]	*0.802*
node degree *(baseline)*	0.074	0.097	0.117	0.096 [0.085, 0.107]	0.457
attention rollout	0.075	0.087	0.108	0.090 [0.083, 0.097]	0.234
saliency	0.078	0.085	0.098	0.087 [0.078, 0.096]	0.419
local CTL density *(baseline)*	0.059	0.059	0.059	0.059 [0.052, 0.066]	0.312
floor: random (baseline)	0.058	0.058	0.058	0.058 [0.049, 0.066]	0.118
distance to nearest CTL *(baseline)*	0.047	0.047	0.047	0.047 [0.044, 0.050]	0.246
GNNExplainer	0.038	0.057	0.045	**0.047** [0.040, 0.053]	0.117

flips the label. Clustering the bootstrap by sample, since the 600 attributions are 100 samples × 6 correlated model-radius settings and resampling rows rather than samples understates the interval by about a third, the margin is -0.011 with a 95% interval of $[-0.015, -0.007]$. It holds in both label classes separately (Section 7.2). It does not extend to the constructive metric, where the same comparison is -0.001 $[-0.003, +0.002]$.

Which method wins reverses with the metric, and no single metric should be quoted alone. This is the central result of the section, and it is stated first because every narrower claim below is conditional on it. Take integrated gradients against tumour-likeness, a heuristic that projects a marker vector onto the tumour mean and consults no model whatsoever. Four measurements are available for that pair, and they do not agree:

- *Average precision, counterfactual set.* Integrated gradients (mean) reaches 0.159 against 0.169, a paired difference of -0.010 with a 95% interval of $[-0.063, +0.044]$. Unresolved, and Section 10 shows it is not resolvable on this dataset.
- *Average precision, constructive set.* Integrated gradients (zero) reaches 0.573 against 0.197. A decisive win for the learned method.
- *Signed rank correlation.* Integrated gradients (mean) reaches $+0.508$ against -0.449 (Table 6). A decisive win for the learned method, by a margin no interval here approaches.

- *Decoy mass.* 0.156 against 0.056. A win for the heuristic.

The heuristic wins one of the four and loses two decisively. A paper that quoted only the first would report that nothing learned beats a marker heuristic; one that quoted only the third would report the opposite. Section 7.3 explains *why* they disagree, namely that it is a property of the task's class asymmetry rather than a defect in any metric, and that explanation is the reason no single number is quoted as the headline here.

A note on multiplicity. Tables 4 to 9 support on the order of fifty pairwise comparisons, and no correction is applied to any of them. For the null results this is harmless: failing to reject is not made easier by a stricter threshold. For the positive claims it is not, and those are flagged as exploratory where they are not independently corroborated, specifically the claims that individual methods score significantly below the marker heuristic. The two claims below are exempt because each is corroborated across metrics or radii rather than resting on one cell.

What survives the disagreement. GNNExplainer sits below the floor on the *counterfactual* metric, at 0.047 against 0.058 with a paired margin of -0.011 $[-0.015, -0.007]$ clustered by sample, and that holds within each label class separately. It does *not* hold on the constructive metric once the explainer is run at the budget the protocol specifies: 0.117 against 0.118, a difference of -0.001 $[-0.003, +0.002]$. It does *not* hold on the other metrics the repository records: against random it is $+0.114$ on signed correlation, better by 0.007 on decoy mass, and better by 0.040 and 0.069 on the deletion and insertion curves.

The second surviving claim is that no learned method reaches the structure-recovery reference: on the stratified scale of Section 7.2 the best covers 58% of the $[0.054, 0.259]$ range. That is a reference level rather than a bound: on positive samples alone integrated gradients scores 0.308 against a positive-class reference of 0.169, i.e.
182% of it, because the reference is the score of an estimator that ranks every member of the family equally, and any method grading within the family can exceed it.

On the choice of driver set. The counterfactual set carries the primary metric because it is the object the generator verifies. But it is a greedy irredundant flipping set, one arbitrary member of a large family (Section 3.4), and a model computing an engaged *fraction* has more reason to represent the constructive set, which is every engaged tumour cell and every enabling CTL. Both are therefore reported throughout. The smooth blob tops the constructive column at 0.802, but that number says nothing about smoothness priors competing with real methods: the blob is drawn around the true driver centroid and is marked oracle-informed for that reason.

7.2 The primary metric must be read class-conditionally

Everything above pools positive and negative samples. That is not safe here, and the reason is the class asymmetry Section 7.3 sets out for a different purpose: on positive samples the counterfactual set is CTL-adjacent, on negative samples it is a quorum of unengaged tumour cells. Those are structurally different targets, so a method can be strong on one and weak on the other, and the pooled mean then reports neither.

It happens here, at a size that dominates everything else in this section. Split by class, and weighting the pooled column to the test split's own class balance of 0.4733 rather than to the 0.42 that the unstratified 100-sample attribution draw happened to yield:

Three things follow, and they revise claims made earlier in this section.

The comparison that Table 4 reports as unresolvable is resolved, and its sign is the opposite of the pooled one. Integrated gradients (mean) against tumour-likeness is $+0.302$ $[+0.275, +0.330]$ on positives and -0.237 $[-0.253, -0.220]$ on negatives. Pooled at the sample's

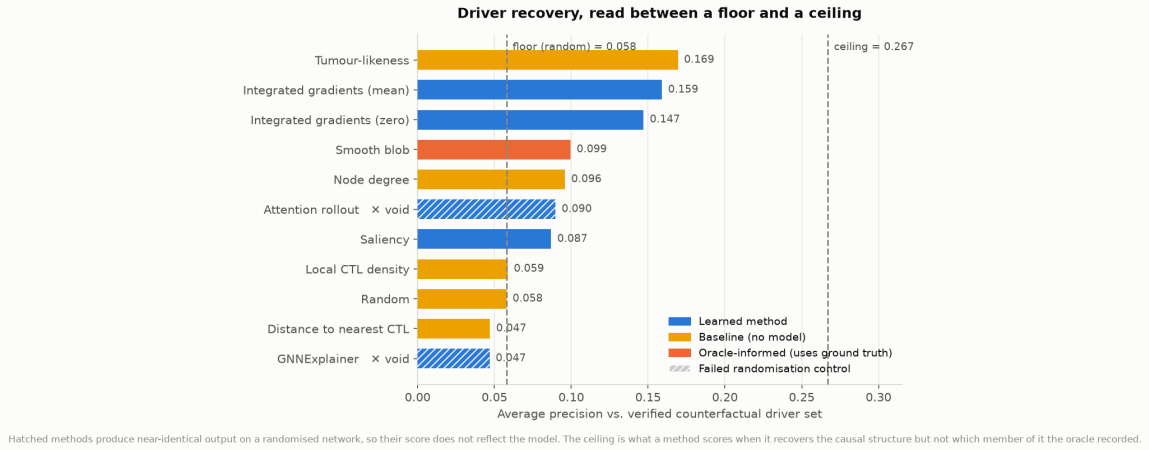

Figure 1: Driver recovery pooled over all model-radius settings, with model-free baselines in yellow and learned methods in blue. Hatched bars failed the randomisation control in a majority of settings under at least one mode, so their height should not be read as evidence about the model. The lower dashed line is the random floor, 0.058; the upper one at 0.267 is a reference level rather than a bound. It is what an estimator scores when it identifies the causal structure but ranks every member of it equally (Section 3.4), and a method that grades within the family can exceed it. The smooth blob is excluded from the baseline group and drawn separately: it is computed from the ground truth and is not a baseline any method is asked to beat. Both dashed lines and every bar here pool the two label classes; Table 5 gives the class-conditional version, in which the apparent tie between the best learned method and the best model-free baseline does not survive. What does read off this ordering is that GNNExplainer sits below the random floor.

Table 5: Average precision against the counterfactual set, split by label class. The stratified column re-weights to the test split's class balance; intervals are within-class bootstraps. Floor and ceiling are class-conditional too: random scores 0.017 on positives and 0.087 on negatives, and the structure-recovery reference of Section 3.4 is 0.169 and 0.339. **Read each column against its own floor and ceiling.** 92% of the variance of the pooled paired difference between any two methods here is between-class rather than within-class.

method	$y = 1$ (n=42)	$y = 0$ (n=58)	stratified, 95% CI
ceiling: structure recovered	*0.169*	*0.339*	*0.259*
integrated gradients (mean)	**0.308**	0.051	0.173 [0.160, 0.186]
integrated gradients (zero)	0.279	0.052	0.159 [0.146, 0.173]
tumour-likeness *(baseline)*	0.005	**0.288**	0.154 [0.143, 0.165]
smooth blob *(oracle-informed)*	0.099	0.100	0.099 [0.088, 0.113]
node degree *(baseline)*	0.056	0.125	0.092 [0.084, 0.101]
saliency	0.127	0.058	0.091 [0.084, 0.098]
attention rollout	0.067	0.106	0.088 [0.082, 0.094]
local CTL density *(baseline)*	0.024	0.083	0.055 [0.051, 0.060]
floor: random (baseline)	*0.017*	*0.087*	*0.054* [0.049, 0.059]
distance to nearest CTL *(baseline)*	0.044	0.049	0.047 [0.044, 0.050]
GNNExplainer	0.014	0.071	0.044 [0.041, 0.047]

42% positive rate those nearly cancel to −0.010; re-weighted to the split's 47.33% they give +0.018 [+0.003, +0.034]. The pooled null was not a measurement of similarity. It was two large opposite effects averaging out, with the sign decided by an unstratified draw.

The power calculation in Section 10 is therefore computed on the wrong quantity. Its standard deviation of 0.2784 is almost entirely the class split (92% of the variance); within class it is 0.066 to 0.092 against effect sizes roughly 25× larger. The comparison needs on the order of 70 samples, not 2752, and is already settled at $n = 100$.

On the class where the target is a genuine minimal sufficient cause, the marker heuristic fails completely. Tumour-likeness scores 0.005 on positive samples against a positive-class random floor of 0.017, a third of chance. Its pooled 0.169 is a negative-class score, and on negatives the target *is* a set of tumour cells, so projecting onto the tumour mean approximates the answer key rather than recovering a cause.

The honest summary of this testbed is therefore not that nothing learned beats a model-free heuristic. It is that the two label classes pose structurally different problems, that the learned methods win the one where the ground truth is causal and lose the one where it is a quorum, and that a pooled average of the two is not a quantity anyone should report. That is a finding about ground-truth attribution benchmarks generally, and it is the reason Table 4's pooled column should be read only alongside this one.

7.3 Direction and decoys

Attention rollout is *anti-correlated* with the signed ground truth, slightly worse than random. Local CTL density puts 42% of its mass on decoys, which is exactly what that baseline was included to demonstrate: local density is a shortcut, and a faithful method must not follow it.

Table 6: Signed rank correlation against the three-valued relevance vector, and the share of attribution mass landing on causally inert decoy CTLs. **Pooling differs from Table 4**: this table averages every row flat, including the CNN where a method applies to it, whereas Table 4 averages three per-radius means over GIN and GATv2 only. For the three methods with CNN rows the conventions differ: integrated gradients (mean) reads +0.508 here against +0.565 under Table 4's rule. Decoy mass is a share of total attribution magnitude, so unlike the other two metrics it is not invariant to a monotone rescaling of a method's scores; read it as a concentration measure rather than a separation measure. The decoy-prevalence null is 0.156.

method	signed ρ	decoy mass
integrated gradients (zero)	+0.623	0.167
distance to nearest CTL *(baseline)*	+0.618	**0.086**
local CTL density *(baseline)*	+0.565	**0.423**
integrated gradients (mean)	+0.508	0.156
saliency	+0.413	0.214
smooth blob *(oracle-inf.)*	+0.283	0.200
GNNExplainer	+0.116	0.149
node degree *(baseline)*	+0.017	0.136
random *(baseline)*	−0.003	0.156
attention rollout	**−0.111**	0.224
tumour-likeness *(baseline)*	−0.449	0.056

Table 6 also contains an apparent contradiction that is worth resolving, because it is a property of the task rather than a defect in the metrics. Distance to nearest CTL scores 0.047 on average precision, below random, yet has the *second-highest* signed correlation and the *second-lowest* decoy mass of anything tested. Integrated gradients (zero) edges it on the first at +0.623 against +0.618,

and tumour-likeness on the second at 0.056 against 0.086. That follows directly from the class asymmetry described in Section 3: for positive samples the counterfactual set is CTL-adjacent, for negative samples it is maximally CTL-distant, so a heuristic tuned to one class is anti-correlated on the other. Tumour-likeness has the mirror-image property: strong average precision at 0.169 but signed ρ of -0.449, because it ranks all tumour cells highly while the relevance vector marks unengaged tumour as -1. Either metric alone would have told a misleading story.

7.4 Deletion and insertion curves

Every metric so far scores against a recorded ground-truth set, which is what Faber et al. [7] object to. The protocol also computes two that do not: *deletion AUC*, the area under the prediction curve as cells are removed in attribution order (lower is better), and *insertion AUC*, the same as they are added back (higher is better). These are model-facing rather than ground-truth-facing. Table 7 reports them.

Table 7: Deletion and insertion AUC, pooled over GIN and GATv2 at three radii. Both rank cells by |attr| and normalise each curve, so they compare curve shape rather than magnitude and are unfriendly to signed methods. Random is the reference at 0.413 on both.

method	deletion AUC ↓	insertion AUC ↑
saliency	**0.274**	**0.696**
local CTL density *(oracle-typed)*	0.322	0.658
attention rollout	0.348	0.393
GNNExplainer	0.375	0.479
integrated gradients (zero)	0.439	0.508
node degree *(baseline)*	0.441	0.532
random (baseline)	*0.413*	*0.413*
tumour-likeness *(oracle-typed)*	0.430	0.347
integrated gradients (mean)	0.513	0.400
distance to nearest CTL *(oracle-typed)*	0.535	0.334
smooth blob *(oracle-informed)*	0.569	0.418

The ordering is not the one the driver-recovery tables give, and that is the point. GNNExplainer, below the random floor on both average-precision metrics, is *better* than random on both curves (-0.040 and $+0.069$ paired, intervals clear of zero). Integrated gradients (mean), the strongest method on driver recovery, is worse than random on both. Saliency, mid-table throughout Section 7, is the best method here by a clear margin.

Two readings are available and this paper cannot separate them. Either the curves and the driver-recovery metrics measure genuinely different things, model-facing faithfulness versus recovery of causes, which is the distinction Section 10 draws, or the curves are dominated by the |attr| ranking, which discards the sign that Section 6 argues is essential here. The second is not controlled for. The table is reported because omitting the only model-facing measurements in a paper about faithfulness is a selection this work is not entitled to make, not because it settles anything.

7.5 Weight randomisation

Table 8 is where the paper's strongest claim sits. GNNExplainer's attributions remain 0.38 to 0.86 rank-correlated with its attributions on a randomised network, void in 10 of 12 settings. The effect is strongly setting-dependent, near-total on GATv2 at every radius and weakest on GIN at

the densest graph, which is also the setting with the most model-specific information to find, so it should be quoted as a range and never as a single number.

Table 8: Rank correlation between attributions on the trained model and on a randomised one, at a converged 400-epoch budget for GNNExplainer. High is bad. Values above 0.5 are recorded as void. Ranges run over every encoder a method applies to: GIN and GATv2 at three radii, plus the CNN for the gradient methods, so the denominator of the void count is 12 for GNNExplainer, 6 for attention rollout (GATv2 only) and 14 for the rest.

method	full	cascading	void settings
GNNExplainer	+0.482 to +0.835	+0.375 to +0.858	10 of 12
attention rollout	+0.171 to +0.456	+0.957 to +0.993	3 of 6
integrated gradients (mean)	−0.387 to +0.124	−0.102 to +0.603	3 of 14
integrated gradients (zero)	−0.367 to +0.191	−0.123 to +0.610	2 of 14
saliency	−0.038 to +0.361	−0.212 to +0.601	2 of 14

Attention rollout splits sharply between the two randomisation modes: it passes under full randomisation and is decisively void under cascading, at 0.957 to 0.993. That disagreement is itself a finding about the control. Figure 2 plots the modes separately for that reason; averaging over them would report a moderate number for a method that is void under one of them.

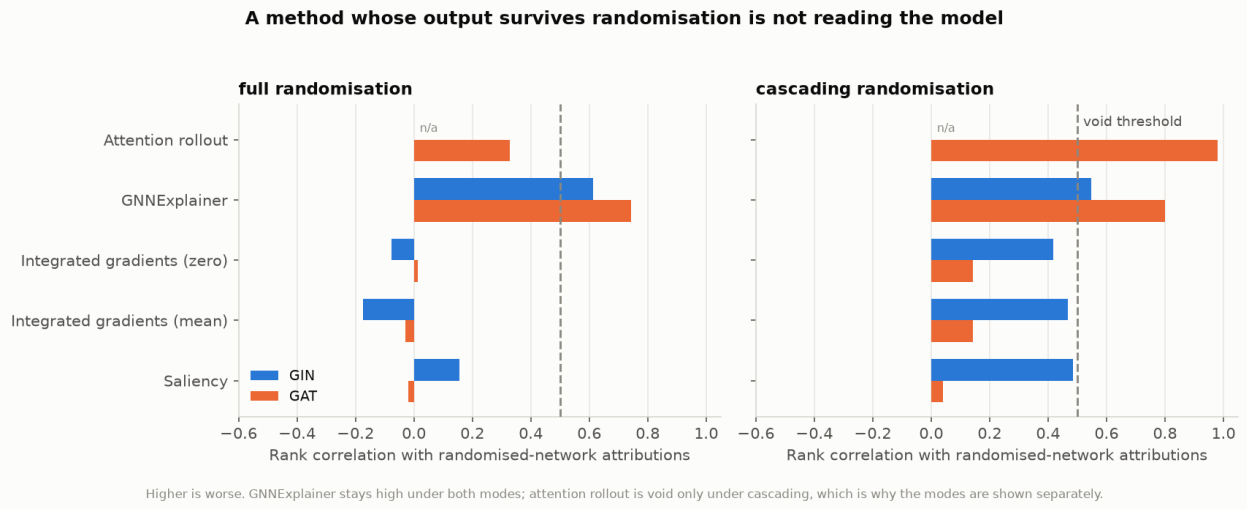

Figure 2: The randomisation control, faceted by mode. Higher is worse: a bar to the right of the dashed line means the method produces nearly the same output whether the network is trained or random. GNNExplainer stays high under both modes. Attention rollout is the reason the two panels are not averaged: it sits near zero on the left and near one on the right.

7.6 Untrained encoders and shuffled labels

Table 9 reports the untrained-encoder and label-shuffle controls, matched encoder for encoder so that both columns are average precision against the counterfactual set. On the untrained encoder, average precision spans 0.045 to 0.106 around a random baseline of 0.058, which is essentially chance for every method. On label-shuffled encoders it spans 0.046 to 0.100.

One result stands out. Saliency on GIN recovers drivers roughly twice as well from a *label-shuffled* encoder (0.095) as from the correctly trained one (0.050).

Table 9: The other two controls, matched encoder for encoder. All three columns are average precision against the counterfactual driver set, so the comparison is like for like; random is 0.058. Bold marks the settings where a control beats the correctly trained model, which happens in 2 of 12 for the untrained encoder and 1 of 12 for shuffled labels. **These are unpaired point estimates and this table carries no intervals**: the control columns come from a 50-sample cohort and the trained column from the 100-sample primary sweep, so a bold cell is not by itself evidence of an effect. Paired per-sample intervals, recomputed on the nine rows where per-sample values were retained, give half-widths from 0.008 to 0.098, a twelvefold range, so no single noise floor is usable for this table. Of the two bold untrained cells, saliency on GIN is $+0.012$ $[-0.004, +0.030]$ and GNNExplainer on GIN is $+0.012$ $[-0.005, +0.034]$; neither clears zero. The one bold label-shuffle cell, saliency on GIN, is $+0.047$ $[+0.032, +0.062]$ and does. The three CNN rows have no per-sample values at all, because `attribution_controls.json` stored per-setting means only. The paired, interval-carrying version of this comparison is Table 10, and where the two disagree that is the one to trust. Both controls run at $r_{\text{graph}}/r_{\text{label}} = 1.0$ only.

method	encoder	untrained	labels shuffled	trained
GNNExplainer	GAT r1.0	0.050	0.053	0.056
GNNExplainer	GIN r1.0	**0.066**	0.050	0.059
saliency	GIN r1.0	**0.060**	**0.095**	0.050
saliency	GAT r1.0	0.085	0.088	0.119
saliency	CNN	0.061	0.046	0.143
attention rollout	GAT r1.0	0.051	0.068	0.087
integrated gradients (mean)	GIN r1.0	0.106	0.100	0.245
integrated gradients (mean)	GAT r1.0	0.052	0.067	0.165
integrated gradients (mean)	CNN	0.045	0.048	0.052
integrated gradients (zero)	GIN r1.0	0.092	0.065	0.168
integrated gradients (zero)	GAT r1.0	0.048	0.066	0.177
integrated gradients (zero)	CNN	0.063	0.074	0.189

A second result did not survive re-running the primary sweep at the specified budget. At 400 mask epochs the trained encoder is ahead in both architectures (label-shuffled 0.050 against trained 0.059 on GIN, 0.053 against 0.056 on GATv2), and the indifference reading an earlier version drew from the reverse sign at 30 epochs is withdrawn (Section 9.5). The untrained-encoder and weight-randomisation evidence for GNNExplainer's model-independence is unaffected.

Note also that this table's control columns come from a 50-sample cohort and its trained column from the 100-sample primary sweep. The matched comparison, on the same 50 graphs both sides, is Table 10; where the two disagree, the matched one is the one to trust.

7.7 Both controls were run at the one radius where they detect least

Table 9 runs at $r_{\text{graph}}/r_{\text{label}} = 1.0$ and nowhere else. That was a consequence of how the code was organised rather than a decision: the label-shuffle arm needs a corrupted checkpoint, the training configuration produced control checkpoints only at ratio 1.0, and the untrained-encoder arm, which needs no checkpoint at all but only a fresh initialisation, was gated behind the same condition.

The radius is the worst available choice for this test. Section 5 shows `is_engaged` is decodable at AUC 1.000 from an *untrained* encoder at ratio 1.0, because $r_{\text{graph}} = r_{\text{label}}$ there makes the property one-hop; training adds nothing because there is nothing left to add. Asking whether training changes an explanation, at the one radius where training demonstrably changes nothing about the node property, cannot detect much.

Both arms were therefore re-run off ratio 1.0: the untrained-encoder arm across all three radii, and the label-shuffle arm across 0.5 and 1.0 only, since a label-shuffled checkpoint at ratio 2.0 would

have to be trained and was not. Ratio 1.0 was repeated as a replication check. All 18 previously published control values reproduce exactly. Table 10 gives GNNExplainer, paired over the same 50 samples.

Table 10: GNNExplainer: control minus trained average precision against the counterfactual set, paired per sample with a 95% bootstrap interval. Positive means the method recovers drivers *better* from a corrupted or untrained encoder than from the correctly trained one. Ratio 1.0 is the only radius previously reported and is the only one at which the comparison is indistinguishable.

control	encoder	$r_{\text{graph}}/r_{\text{label}} = 0.5$	$r_{\text{graph}}/r_{\text{label}} = 1.0$
untrained encoder	GIN	+0.013 [+0.008, +0.018]	+0.012 [−0.005, +0.034]
untrained encoder	GATv2	+0.013 [+0.009, +0.017]	−0.004 [−0.013, +0.003]
labels shuffled	GIN	+0.019 [+0.013, +0.027]	−0.003 [−0.009, +0.004]
labels shuffled	GATv2	+0.012 [+0.009, +0.015]	−0.001 [−0.011, +0.008]

At ratio 0.5, where the property stops being one-hop and training buys +0.050 on GIN (Section 5), GNNExplainer recovers drivers *significantly better* from a corrupted encoder than from a trained one, on both controls and in both architectures, with every interval clear of zero. The untrained arm extends to ratio 2.0, where the effect is larger still: $+0.031$ $[+0.021, +0.042]$ on GIN and $+0.027$ $[+0.017, +0.037]$ on GATv2.

The conclusion the original table supported is therefore correct and was measured at its weakest point. The honest form of the claim is not that GNNExplainer is indifferent to the model, which ratio 1.0 alone can only fail to reject, but that at radii where there is model-specific structure to find it does measurably *worse* for having a trained model to explain.

Two cautions. This strengthening applies to GNNExplainer alone: integrated gradients shows large, decisively signed untrained-minus-trained gaps at ratio 1.0 (-0.108 to -0.192; the trained encoder is ahead by that margin), so the radius is not uninformative in general, only for this method. And the effects are small in absolute terms, hundredths of average precision against a floor of 0.058, resting on 50 paired samples.

7.8 The control depends on the explainer's optimisation budget

Table 11 and Figure 3 report what is, in the author's view, the most transferable finding here. A 30-epoch GNNExplainer mask has barely left its random initialisation: mask movement of 0.03 means the attribution is about 97% rank-identical to its one-epoch state. Two such masks, one from the trained network and one from the randomised one, sit near a shared starting point, and the control dutifully reports a high correlation that is mostly an artefact of both being unoptimised. Reseeding the mask independently between the two conditions changes little; epochs were the confound.

By 200 to 800 epochs the correlation settles into a 0.28 to 0.38 band while mask movement keeps rising, so the optimiser is still working and the metric has genuinely converged rather than the search stalling. The band is not monotone, since the ratio-2.0 series ticks back from 0.288 at 400 epochs to 0.342 at 800, and that movement is within the run-to-run spread of the control itself, which the seed replication puts at up to 0.08 within an architecture. It should not be read as a trend.

One caveat on the budget used elsewhere in this paper. Convergence is demonstrated at $r_{\text{graph}}/r_{\text{label}} = 2.0$, where the series runs to 800 epochs and settles between 200 and 400. The ratio-1.0 series stops at 200 epochs and never reaches a plateau, so the 400-epoch budget used throughout Sections 7 and 7.7 is *assumed* converged at that radius rather than shown to be. Noth-

Table 11: GNNExplainer under increasing mask-optimisation budget, 25 test graphs. "Mask movement" is how far the optimised mask has travelled from its state after one epoch. The randomisation correlation moves by a factor of three; driver recovery does not move at all.

		randomisation correlation			
$r_{\text{graph}}/r_{\text{label}}$	epochs	same mask seed	independent seed	AP	mask movement
1.0	30	0.945	0.865	0.041	0.030
1.0	100	0.801	0.741	0.043	0.060
1.0	200	0.700	0.659	0.047	0.134
2.0	30	0.890	0.852	0.038	0.041
2.0	100	0.578	0.554	0.037	0.055
2.0	200	0.378	0.366	0.037	0.086
2.0	400	**0.288**	**0.283**	0.040	0.178
2.0	800	0.342	0.340	0.049	0.364

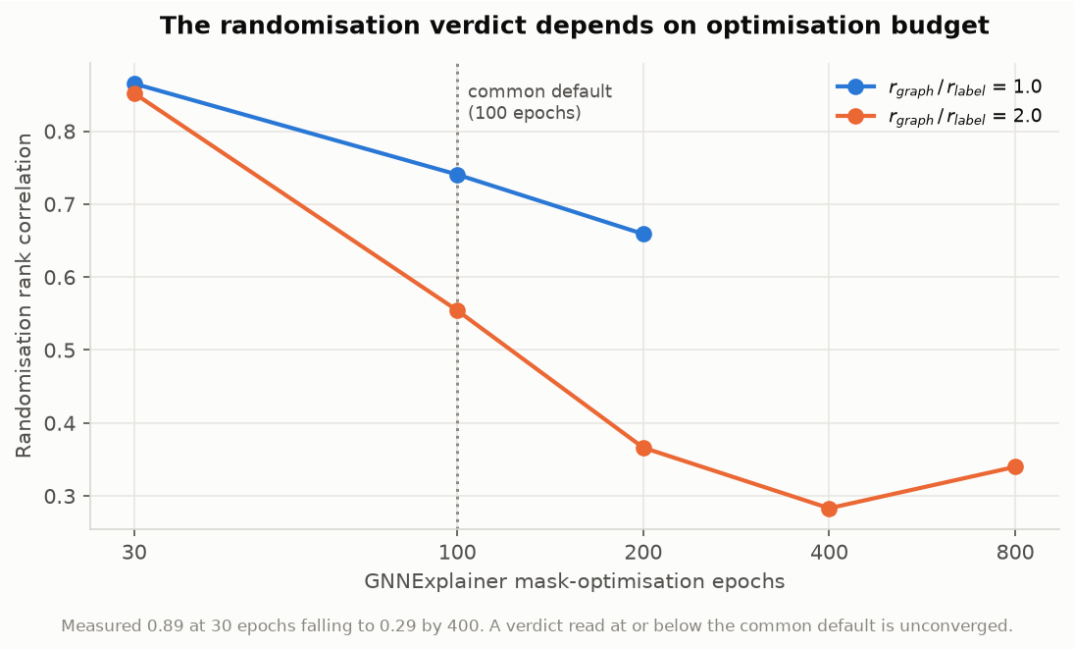

Figure 3: The randomisation verdict for GNNExplainer as a function of how long its mask is optimised. The plotted series is the independently seeded mask. Following the $r_{\text{graph}}/r_{\text{label}} = 2.0$ curve, the one run to convergence, the correlation reads 0.85 at 30 epochs, where the method looks completely model-blind, and 0.28 by 400. The same-seed column of Table 11 moves 0.89 to 0.29 over the same range, and that is the pair quoted in the abstract. The $r_{\text{graph}}/r_{\text{label}} = 1.0$ curve stops at 200 epochs and never reaches convergence. The dotted line marks the common 100-epoch default, which sits inside the unconverged region. This is a 3× swing on the metric that decides whether a method is called faithful, produced by a parameter of the *explainer* rather than of the model or the data.

ing in the results turns on it, since driver recovery is flat across a 27× range of budget, but the randomisation correlations at ratio 1.0 carry that assumption.

The practical consequence is short. Any published use of the model-randomisation control on a mask-based explainer must report the optimisation budget, and any result at or below the common 100-epoch default should be treated as unconverged. This is a requirement on how the control is used, not a defect in the control: at 30 epochs the mask has barely left its initialisation, so what the test compares is two unoptimised masks rather than two explanations.

The same table shows that average precision against the counterfactual set is comparatively insensitive to budget: 0.037 to 0.049 across a 27× range and both graph radii, against 0.058 for random. "Comparatively" is the right word rather than "flat". Re-running the full primary sweep at 400 epochs after an earlier version had run it at 30 (Section 9) moved pooled counterfactual AP from 0.044 to 0.047 and constructive AP from 0.100 to 0.117, enough to take the constructive comparison from below the random floor to indistinguishable from it. GNNExplainer's failure to recover drivers on the counterfactual metric is not a budget artefact; the size of its shortfall is partly one.

7.9 Seed stability

Two additional training seeds per architecture were run to check that the conclusions are not one lucky initialisation (Figure 4). Three readings follow.

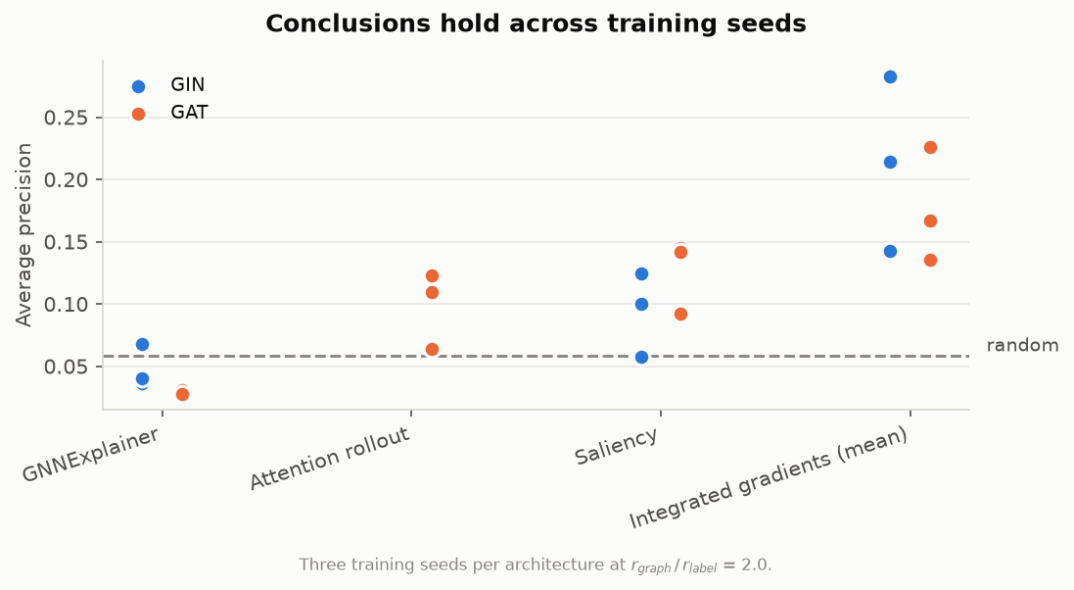

Figure 4: Three training seeds per architecture at $r_{\text{graph}}/r_{\text{label}} = 2.0$. Accuracy is seed-stable (test AUC 0.9993 to 0.9999 across the six checkpoints plotted; 0.9997 to 0.9999 across the two replicate seeds). GNNExplainer is below the random line in five of six runs. Integrated gradients is the top-scoring learned method in five of six model-seed combinations.

The claims hold, with one hedge. GNNExplainer is below random in five of six runs; the exception is GIN seed 1 at 0.068, marginally above 0.058. The honest statement for GIN is therefore "at or below random"; for GATv2 it is unambiguous, at 0.028 to 0.032, roughly half the random baseline in every run.

The randomisation result is *architecture*-dependent, not seed-dependent. Within an architecture the seed spread is at most 0.08; between architectures the gap is about 0.5. GNNExplainer is close

to model-blind on GATv2 (0.83 to 0.89) and only moderately so on GIN (0.29 to 0.37). Reporting a single pooled mean hid this, and the split is the more useful finding: the method's model-sensitivity depends on what it is explaining.

The replication now covers both randomisation modes, and the mode disagreement that Section 7.5 treats as a finding about the control is itself seed-stable. For attention rollout on GATv2 the cascading correlation is +0.996, +0.993 and +0.997 across the three seeds while the full-randomisation correlation is −0.320, −0.436 and −0.124. A method can therefore be pinned at near-total model-blindness by one mode of the control and cleared by the other, reproducibly, with nothing changing but which parameters were reinitialised. That is a property of the control, not of one initialisation.

One scope note on those numbers. At $r_{\text{graph}}/r_{\text{label}} = 2.0$ on 30 graphs the full-mode correlation for attention rollout is reproducibly negative, whereas Table 8 reports a full-mode range of +0.171 to +0.456 pooled over a different set of settings. The two are not in conflict, but that range should not be read as covering this one.

8 A second label family, and why no model could learn it

The strongest objection to the results above is that the engagement label is a *one-hop* property. Section 5 showed `is_engaged` is decodable at AUC 1.000 from an untrained encoder, so perhaps explainers look model-independent because there is nothing model-specific to find. A second label family was built to answer that objection directly.

The niche label counts three-cell motifs: $y = 1$ if and only if at least $k = 10$ triads of one tumour cell, one CTL and one macrophage have all pairwise distances within r_{motif}. This is not one-hop and cannot be read off a single aggregation step. Driver sets are genuinely sparse, at a median of 31 cells against roughly 200 for the engagement label.

r_{motif} had to be measured rather than chosen. Accidental triads formed by background cells alone number 122 per sample at 30 μm, 20.8 at 20, 3.4 at 15 and 1.0 at 12. At the engagement radius the background would swamp a planted signal of about 10 by more than twelvefold, and the label would have measured global density rather than motif presence. It was set to 15 μm.

The generator passes every invariant the first one does: exact recomputation from the serialised record, hard-core separation, matched decoys, and a non-spatial shortcut probe at 0.465 against a ceiling of 0.55. Class balance is 0.5225.

Neither architecture learns it. GIN reaches test AUC 0.468 and GATv2 0.518, both at chance, while the same architectures reach 0.999 on the engagement label.

This is not a data problem. A logistic regression on a single feature, the triad count computed from marker-classified cell types, separates the classes at AUC 1.000. The information the graph exposes is sufficient.

It is also not the readout. The engagement label is a ratio, so mean pooling is the right inductive bias, and a count label ought to prefer sum pooling. Tested directly: mean+max gives 0.523, sum+max 0.497, sum+mean 0.419. No improvement.

The most likely explanation is expressiveness. The label requires counting a triangle motif, a 3-clique in the proximity graph with a type constraint on its vertices, and message-passing networks bounded by 1-Weisfeiler-Lehman provably cannot count triangles [6]. GIN is exactly 1-WL expressive by construction. GATv2 is no stronger in this respect, and the reason is worth one sentence rather than an assertion: attention reweights the messages a node aggregates from its neighbours but does not enlarge the neighbourhood being aggregated over, so the computation still factors through the multiset of one-hop neighbour states that bounds 1-WL. Learned coefficients

change which neighbours dominate, not what information is available to distinguish two nodes whose neighbourhood multisets agree. The observed pattern fits: GATv2 reaches train AUC 0.773 while sitting at 0.518 on test, memorising individual graphs rather than learning a function it can represent. This is an interpretation consistent with the evidence, not something demonstrated here; the direct test would compare against a substructure-aware architecture and check that the gap closes.

What this family produces is a boundary condition worth stating plainly:

> Faithfulness benchmarks are limited by model expressiveness, not just by label design. A ground-truth task that a model class provably cannot represent is unusable for evaluating that class's explanations, however well constructed the ground truth is.

Attribution cannot be assessed on this label, because there is no model that solved it; explaining a chance-level classifier would measure nothing. The one-hop objection to Section 7 therefore remains formally open. The natural next step is to weaken the label from counting to *existence* ($k = 1$), which is a detection problem rather than a counting one and may fall inside 1-WL, while keeping the driver sparsity the family was built for.

9 Corrections

Claims made during this work that were wrong and were withdrawn on re-measurement are set out here rather than quietly fixed. Four arose during the original study (Sections 9.1 to 9.3: three below, and the fourth corrected where it occurred in Section 7, because it was a claim about the headline comparison and belongs beside the number that replaced it). Four more were found in a later audit of the finished manuscript (Section 9.4). A subsequent full audit, which re-derived every number rather than only the changed ones, withdrew a further ten, including three headline claims (Section 9.5). Eighteen in total. All are reported rather than quietly fixed, because in each case the error was of a kind that a reader of the final numbers alone could not detect, and because most of them are mistakes that seem easy to repeat.

Worth noting that the fourth repeats the second. Section 7.8's retraction was caused by generalising from the most favourable cell of a grid, and the abstract of an earlier draft did the same thing, quoting integrated gradients at its best radius against a baseline pooled over all of them. Recognising a failure mode is not the same as being immune to it.

9.1 The generator was separable at AUC 0.928 without looking at a tumour cell

The original design placed one enabling CTL per engaged tumour cell. At the working cell density a single CTL already engages about six tumour cells, since median tumour nearest-neighbour spacing is $10.8\,\mu$m against $r_{\text{label}} = 30\,\mu$m, so this oversupplied enabling CTLs roughly fourfold and raised local CTL density inside the engaged region about ninefold above background. Decoy CTLs, placed uniformly, were then separable from enabling CTLs by local density alone.

Measured: a gradient-boosted model given *only* CTL positions, with all tumour information withheld, reached AUC 0.928.

This passed every test in the suite at the time, because those tests constrained only non-spatial shortcuts and were permutation-invariant by construction. Had it survived, every attribution method would have scored well by finding the dense clump, and the benchmark would have reported a far more optimistic picture than the truth.

The fix was to draw all CTLs, enabling and decoy alike, from a common aggregate process, and to choose enabling CTLs as a greedy *covering* set of the engaged region at whole-aggregate granularity rather than one per cell. Decoy aggregates use the identical generator, anchored beyond $r_{\text{label}} + 3\sigma_{\text{agg}}$ from every tumour cell. Afterwards the CTL-only probe reads 0.657, and the standardised mean difference in local density between enabling and decoy CTLs is 0.074.

Two tests were added: one asserting that local CTL density is matched between enabling and decoy CTLs, and one capping the CTL-only probe.

A related realisability failure was found later, during work on the image baseline: 2.23% of cells overlapped physically, and about 7.7% of driver cells had a partner within $2\,\mu$m. No attribution method, graph or pixel, can separate two cells at the same point, so this put an irreducible error floor under the headline metric. Hard-core placement removed it entirely.

9.2 The randomisation verdict was read at an unconverged budget

The first version of this work reported GNNExplainer as void on the basis of a correlation of about 0.93 with its output on a randomised network. That number was produced at 30 mask-optimisation epochs against a common default of 100, and it does not survive a proper budget, which is the substance of Section 7.8.

The retraction was then itself over-corrected. On the ratio-2.0 probe alone the verdict was revised to "not void, $\rho \approx 0.3$". That generalised from the single most favourable cell of the grid. The full control sweep re-run at 400 epochs across all six encoders gives a mean of 0.68 and a range of 0.375 to 0.858, void in 10 of 12 settings.

The epistemic history, kept deliberately: 0.93 (30 epochs, unconverged) $\rightarrow$ 0.29 (converged, one cell, over-generalised) $\rightarrow$ 0.38 to 0.86 (converged, full grid). Two of those three were wrong, in opposite directions.

9.3 Two control tables compared different metrics

The third error was found in an audit of the finished results and is the most ordinary of the three. The untrained-encoder table put average precision against the *constructive* driver set in its control column, and average precision against the *counterfactual* set in its trained column. The label-randomisation sentence quoted a range of constructive values against 0.058, which is the random baseline for the counterfactual metric; the constructive baseline is 0.118. Comparing one to the other's baseline roughly doubles the apparent size of the control.

Two claims did not survive. "Saliency under label shuffle reaches 0.343, higher than any method achieves on a correctly trained model" is false on its own metric: trained saliency reaches 0.593. And the blanket reading that several methods recover drivers better from an untrained network shrinks to 2 of 12 matched settings, both marginal and both near chance. The corrected comparison is Table 9.

Compounding it, the untrained figures had never been refreshed after the controls were re-run at 400 epochs, so they came from a superseded file. The acceptance test for that section asserted only that the JSON keys existed, which is why neither problem was caught. It now parses the tables out of the report and holds every cell to the data files they claim to summarise, including a completeness check so that a table cannot report a favourable subset.

The general lesson is dull and worth stating anyway: a metric and its baseline travel together, and a test that checks a result was *produced* is not a test that checks it was *reported correctly*.

9.4 Four errors found in a later audit

A subsequent audit of the finished manuscript found four more, none of which a reader of the numbers alone could have detected. They are grouped here because they share a shape: each is a place where a convention was applied without being checked, and three of the four were caught only by recomputing something that had never been recomputed.

A pooled value that was not a pooled value. Table 4's pooled column is the mean of the three per-radius entries. Recomputing all eleven rows, ten reproduce exactly and node degree does not: it was printed as 0.074, which is its ratio-0.5 entry. Node degree is the only baseline that is not radius-invariant, because degree is a direct function of the connection radius, so it is the only row where the distinction could show. The corrected value is 0.096, which moves it above attention rollout and saliency and changes the ordering the figure caption describes.

An oracle counted as a baseline. The smooth blob is a disc around the centroid of the *true* driver region. It was listed among six "model-free baselines", marked identically to the other five in Table 4, counted among them in Figure 1's caption, and used to support the claim that "a pure spatial-smoothness prior beats every real method" at 0.802 on the constructive set. An estimator given the location of the answer scoring well is not a fact about smoothness priors. The prose defining it was accurate throughout; the tables and captions were not, which is the more dangerous failure, because a benchmark's baseline taxonomy is what other work reuses. It is now marked oracle-informed and excluded from every baselines-versus-methods comparison.

Half a limitation stated as a whole one. Section 10 asserted that the spatial contiguity of driver sets is "forced by the geometry rather than chosen". That holds for positive samples. For negative samples every k-subset of the unengaged pool flips the label identically, so the choice among them is entirely free, and the lowest-`cell_id` convention happens to select a spatially compact set in 56 of 58 samples. The claim was made from the positive-sample case and generalised without checking the negative one.

A bolded cell that does not clear noise. Table 9 bolds saliency on GIN in the untrained column, 0.060 against 0.050. That comparison put a 50-sample control mean against a 100-sample trained mean and had no interval on either side. Recomputed on matched samples it is $+0.012$ $[-0.004, +0.030]$. The label-shuffle claim on the same row survives at $+0.047$ $[+0.032, +0.062]$; the untrained one does not. One row, two claims, one of them real.

The lesson here is narrower than the previous section's and worth stating separately: a convention that is correct in the case you checked is not thereby correct in the case you did not, and the cases most likely to differ are exactly the ones a uniform-looking table hides.

9.5 A full audit, and the claims it withdrew

A subsequent audit re-derived every numeric claim in the manuscript rather than only those added in the previous revision, and put the paper in front of readers who had not seen its development. It withdrew more than the previous four rounds combined, and the pattern is worth naming: each earlier round checked the numbers that had *changed*, and each missed something in the numbers that had not.

The headline comparison was a pooling artefact, and its sign was set by an unstratified draw. This is the substantial one, and Section 7.2 now carries the corrected analysis. The claim that integrated gradients is "statistically indistinguishable" from a marker heuristic, the power calculation concluding the comparison needed 2752 samples, and the conclusion that "nothing learned outperformed the best model-free baseline" were all computed on a mean over two label classes whose ground truths are structurally different and whose effects run in opposite directions

at roughly ±0.3 AP. Ninety-two per cent of the variance the power calculation treated as noise was that class split. Worse, the 100-sample attribution cohort was drawn without stratifying on label, at 42% positive against the test split's 47.33%, and re-weighting to the split's own balance reverses the sign of the headline difference. Three claims withdrawn.

"Below the random floor on every metric" was true of two metrics out of six. The supporting sentence cited the two average-precision columns; Table 6 on the same page already showed GNNExplainer above random on signed correlation and better than random on decoy mass, and Section 7.4 adds two more it also wins.

Two model-facing metrics were computed, committed, and never reported. Deletion and insertion AUC are in the results file for every method. They are the only measures in this project that do not depend on a recorded ground-truth set, which is precisely the objection Section 2 says the paper must answer, and their ordering differs from the driver-recovery ordering. Section 7.4 now reports them.

The primary sweep did not run at the budget the Methods section claimed. Every version of this paper up to this audit stated that GNNExplainer was optimised for 400 mask epochs throughout. It was not: the committed results file was byte-identical to a superseded artifact produced at 30 epochs, and the per-stage findings document in the repository recorded that only the *controls* had been re-run at 400. Neither results artifact stored an epoch count or a dataset fingerprint, so there was nothing to check the claim against, which is why five separate verification passes did not catch it.

The primary sweep has now been re-run at 400 epochs and the tables in Section 7 report it. The effect is real but bounded: pooled counterfactual average precision for GNNExplainer moves from 0.044 to 0.047, constructive from 0.100 to 0.117, and signed correlation from +0.111 to +0.116. One claim changes as a result: GNNExplainer is no longer below the random floor on the constructive metric, where it is now indistinguishable from it. The paper's assertion that driver recovery is "flat" across budget is downgraded to "comparatively insensitive".

In a paper whose central contribution is that explainer optimisation budget must be reported alongside any randomisation result, this is the most uncomfortable entry in this section. The generalisable form of it is the one already stated in Section 9.3: a result artifact that does not record the settings that produced it cannot be audited, and this project fingerprinted its checkpoints but not its results.

A fifth claim fell with the budget correction. At 30 epochs GNNExplainer recovered drivers marginally better from a label-shuffled encoder than from a trained one, which earlier versions read as indifference to whether the labels were real, "what a method reading graph structure rather than the model looks like". At 400 epochs the trained encoder is ahead in both architectures. The randomisation and untrained-encoder evidence for GNNExplainer's model-independence is unaffected; the label-shuffle leg of it is not.

Smaller corrections, each of a kind already catalogued above. The 0.267 figure was described as a ceiling when it is a reference level an estimator can exceed, and does exceed by 82% on positive samples. The subspace-ablation experiment was reported as probing the counterfactual driver set when it probes the constructive one. Node degree's *constructive* pooled value repeated, uncorrected, the exact error Section 9.4 documents for its counterfactual value one column over. The smooth blob was relabelled oracle-informed in one table and not the other, in the revision that announced the relabelling. Three "model-free" baselines consume the oracle's cell-type column. Two tables pool over different encoder sets without saying so. And the text claimed distance-to-nearest-CTL had the highest signed correlation and lowest decoy mass of anything tested when the table beside it showed it second on both.

The uncomfortable general lesson, which the previous four rounds did not reach: a check scoped to what changed cannot find an error in what did not, and a check that verifies numbers against artifacts cannot find a number that is the wrong quantity. Every gate this manuscript passed was of one of those two kinds.

10 Limitations

Faithfulness to the model is not recovery of causes, and this paper scores the second. Faithfulness ordinarily means that an explanation describes what the model used. The ground truth here is what *causes the label*. Those two things come apart, and nothing in the protocol forces them together. A method could describe a trained model perfectly and still score near zero, if that model is not representing the counterfactual driver set.

That is not a hypothetical here; it is measured. Section 5 removes the subspace carrying `is_driver` from the frozen representation and finds the model's accuracy unaffected relative to a rank- and variance-matched control, while removing the `is_engaged` subspace collapses it below chance. The models represent and use engagement, the quantity their label is a function of, and do not separably encode membership of the *constructive* driver set, which is the target the probe was fitted against. That is the canonical set, not the arbitrary one, so the result cannot be dismissed on the grounds that the target was ill-defined; it should be read as an upper bound on what this instrument can detect rather than as a demonstration of absence (Section 5).

So the primary metric scores attribution methods against a target the model demonstrably does not represent, and the ceiling in Table 4 quantifies the resulting headroom loss: 0.267 rather than 1. Section 7 reports both driver sets for this reason, and the ordering does change between them. What bounds the problem is the control suite. A method whose output is unchanged by randomising the weights is not describing the model under any reading, so the randomisation and untrained-encoder results stand independently of which driver set is the right target. The driver-recovery numbers do not, and should be read as recovery of causes rather than as faithfulness in the strict sense.

One label family, one synthetic tissue. The claim is not that GNNExplainer is useless in general. It is that on a task where the ground truth is known by construction, its output was close to indistinguishable from its output on an untrained network, and it recovered the causal cells below chance.

The distance from this tissue to real multiplexed imaging is worth enumerating rather than gesturing at, because it is where the results stop travelling. The testbed has six marker channels against the twenty to sixty a real panel carries; Gaussian per-cell noise against spillover between channels, autofluorescence and batch effects; exact positions against segmentation error, merged and split cells, and cell-type assignment that is itself a noisy classifier; one uniform $1\,\mathrm{mm}^2$ field against tissue architecture with glands, vessels, necrosis and margins, where density varies by orders of magnitude across a slide; and four discrete types in fixed proportions against a continuous phenotype space with ambiguous intermediate states. The one-at-a-time sweep described below is specified to probe the first three of those and was not run, which is the concrete gap between this testbed and a claim about practice.

The one-hop objection is open. Section 8 was built to close it and did not.

Driver sets are spatially contiguous, and for negative samples that is an artefact. For positive samples contiguity is forced by the geometry: enabling CTLs come from tight aggregates. For negative samples it is not forced at all. Every k-subset of the unengaged tumour pool flips the label identically (Section 3.4), so the oracle's choice among them is free, and the rule it uses, lowest

`cell_id`, selects a spatially compact set, because `cell_id` follows placement order and placement is aggregate-clustered. The recorded target is more compact than a random equally-valid set in 56 of 58 negative samples, 147 μm mean radial dispersion against 280 μm.

Either way the consequence is the same and it is a real limitation: any method with a smoothness prior collects an advantage it has not earned. But the two halves have different remedies. The positive-sample case is intrinsic to the label and can only be disclosed. The negative-sample case is a tie-break that could be changed, by drawing the recorded set at random from the pool or scoring against the pool itself, and until it is, part of what the primary metric rewards on negative samples is agreement with an arbitrary indexing convention.

Resolution, and a comparison this dataset cannot settle. At 100 test samples per cell, the 95% interval on a paired difference between two methods runs from ± 0.003 to ± 0.054 AP depending on the pair, with a median of ± 0.016. The widest intervals belong to the highest-variance methods, which includes the integrated-gradients-versus-tumour-likeness comparison: that one is resolvable only to about ± 0.05.

An earlier version of this section computed that resolving the pooled integrated-gradients-versus-heuristic difference would need $n \geq 2752$ test samples against the 300 available, and concluded the comparison was out of reach by an order of magnitude. That calculation is arithmetically correct and answers the wrong question. Its standard deviation of 0.2784 is almost entirely the label-class split, with 92% of the variance lying between classes rather than within them, so it measures a blocking factor the design should have conditioned on, not sampling noise. Within class the standard deviation is 0.066 to 0.092 against effect sizes roughly 25$\times$ larger, the required n is of order 70, and the comparison is settled at the sample size already in hand (Section 7.2). The claim of unresolvability is withdrawn.

Capacity asymmetry. The CNN has 13 to 18 times the parameters of the GNNs. This is deliberate and argued in Section 4, but it means the graph-versus-image comparison is between representations at their own natural capacity, not between equal budgets.

The failure sweep was not run. A one-at-a-time sweep over marker noise, cell density, detection error, label margin and graph radius is specified and gated, and would map the regime where accuracy stays high while faithfulness collapses. It costs roughly 18 hours of GPU time and was not executed; the specification and its acceptance tests are in the repository, and the runner is not.

Failing a randomisation control is evidence about model sensitivity, not about biology. It does not by itself establish that the highlighted cells are uninteresting. They may be the right cells, found for the wrong reason. That distinction matters for how such a result would be used in practice.

11 If you already have such a figure

The audience with most at stake here is not the one this paper has been addressing. Cell-graph GNNs are used to predict clinical outcomes from multiplexed tissue, and the subgraphs they highlight get read as spatial motifs with biological meaning [17]. If you have such a figure in a manuscript, three checks follow from the results above and none of them require re-running the model.

First, ask whether a randomisation control was run and at what budget. If the explainer is mask-based and the mask was optimised for 100 epochs or fewer, Section 7.8 says that control has probably measured the mask's initialisation rather than the model, and the result should be treated as unmeasured rather than as a pass.

Second, ask whether the highlighted cells are distinguishable from what a marker heuristic would highlight. On this testbed, projecting a marker vector onto a cell-type mean, a computation that consults no model at all, matched the best learned method on the primary metric. A highlight map that agrees with cell typing has not yet demonstrated that it found anything the cell types did not already give you.

Third, if a method does fail a randomisation control, note what that does and does not mean. It establishes that the explanation is not describing the model. It does not establish that the highlighted cells are biologically uninteresting; they may be the right cells identified for the wrong reason. Those two readings have very different consequences for what you do next, and the control cannot distinguish them.

12 Conclusion

Building tissue where the drivers are known is more work than evaluating on tissue where they are not, and most of that work goes into making sure the benchmark cannot be gamed. The generator had to be rebuilt once after a local-density artefact let a model reach AUC 0.928 on the label without looking at a tumour cell. The payoff is that faithfulness becomes a measurement rather than an assertion.

What the measurement says, in this setting: GNNExplainer, with only its node-feature mask measured, recovers the causal cells below chance and produces largely the same output on an untrained network; attention rollout is anti-correlated with the signed ground truth and is void under cascading randomisation; and integrated gradients is the only method left standing after the controls. Conditioned on class, integrated gradients beats every model-free baseline on the positive class by a wide margin and loses on the negative class, where the target is a quorum of tumour cells and a marker heuristic approximates the answer key. Weighted to the test split's own class balance, it wins. A pooled average of the two classes is not a quantity anyone should report, and the pooled summary an earlier draft gave instead is withdrawn (Section 9.5).

Two results reach past the testbed. The randomisation control, the standard tool for deciding whether an explanation is about the model, can be made to return either verdict by changing the explainer's optimisation budget, so that budget must be reported alongside it. And a benchmark's difficulty is bounded by what the model class can represent: making a ground-truth task harder eventually makes it impossible rather than more demanding, at which point it measures nothing about explanations at all.

Reproducibility

Code, specifications, acceptance tests, trained weights and figures are in the project repository, which is public, at `https://github.com/tanmayhinge/tissue-graph-probing`. This manuscript is written from the per-stage findings documents in that repository, so passages of it overlap substantially with text already posted there. That is prior dissemination by the same author rather than reuse of anyone else's work, and it is disclosed here because an automated similarity check will find it. The datasets are not committed, at roughly 10 GB, and regenerate deterministically from a fixed master seed in a few minutes. The full pipeline is:

```
python scripts/generate.py --config configs/base.yaml --n 2000 --out data/base
python scripts/train_gnn.py
python scripts/precompute_renders.py --with-marker-shuffle
python scripts/train_cnn.py
python scripts/probe.py
python scripts/attribute.py --gnnexplainer-epochs 400
python scripts/generate_niche.py --config configs/niche.yaml --n 2000 --out data/niche
python scripts/train_gnn.py --config configs/train_niche.yaml
python scripts/train_gnn.py --config configs/train_control_r0.5.yaml --skip-main
python scripts/control_ratio_sweep.py
python scripts/control_ratio_sweep.py --arms label --ratios 0.5 1.0
python scripts/counterfactual_family.py
python scripts/probe_intervene.py
python scripts/seed_compare_modes.py
python scripts/figures.py
python -m pytest tests/ -q
```

Each stage has an acceptance test suite written before the implementation: 24 tests for the generator and label oracle, 27 for graph construction and the GNNs, 15 for the image baseline, 12 for probing and 15 for attribution, 93 in total, all passing. A further 14 gate the unrun failure sweep, of which the 4 covering the niche generator run today.

Every checkpoint records a fingerprint of the dataset it was trained on, and a staleness check (`check_staleness`) refuses to let a number be quoted after the data underneath it has changed. All 23 engagement-label runs and both niche runs are current against the data on disk: the 21 from the original study plus the two ratio-0.5 label-shuffle controls added for Section 7.7. Training the 21 original runs takes about 3.4 hours in total on one Apple M2 CPU, one process at a time; the memory ceiling, not compute, is the binding constraint.

Two gaps in that guarantee are worth stating. `reports/attribution_results.parquet` and `reports/attribution_controls.json` record neither a dataset fingerprint nor the hyperparameters that produced them, so a results file cannot be checked against the settings it is quoted under, which is how the mask-budget discrepancy in Section 9 survived. The newer control artifacts do carry fingerprints. The bootstrap procedure behind every interval in this paper was, until recently, not committed either. It is now `tgp.stats.bootstrap`, which resamples the pairing unit and takes one value per sample, so the pseudo-replication that produced an over-narrow interval in an earlier draft is unavailable to a caller by construction. Its acceptance tests pin that property directly.

References

[1] S. Abnar and W. Zuidema. Quantifying attention flow in transformers. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2020.

[2] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[3] C. Agarwal, O. Queen, H. Lakkaraju, and M. Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 2023.

[4] K. Amara, R. Ying, Z. Zhang, Z. Han, Y. Zhao, Y. Shan, U. Brandes, S. Schemm, and C. Zhang. GraphFramEx: towards systematic evaluation of explainability methods for graph neural networks. In *Proceedings of the First Learning on Graphs Conference (LoG)*, PMLR 198, 2022.

[5] S. Brody, U. Alon, and E. Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations (ICLR)*, 2022.

[6] Z. Chen, L. Chen, S. Villar, and J. Bruna. Can graph neural networks count substructures? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[7] L. Faber, A. K. Moghaddam, and R. Wattenhofer. When comparing to ground truth is wrong: on evaluating GNN explanation methods. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 332-341, 2021.

[8] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[9] G. Jaume, P. Pati, B. Bozorgtabar, A. Foncubierta, A. M. Anniciello, F. Feroce, T. Rau, J.-P. Thiran, M. Gabrani, and O. Goksel. Quantifying explainers of graph neural networks in computational pathology. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[10] A. Lucic, M. ter Hoeve, G. Tolomei, M. de Rijke, and F. Silvestri. CF-GNNExplainer: counterfactual explanations for graph neural networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, PMLR 151, 2022.

[11] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang. Parameterized explainer for graph neural network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[12] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206-215, 2019.

[13] B. Sanchez-Lengeling, J. N. Wei, B. K. Lee, E. Reif, P. Wang, W. W. Qian, K. McCloskey, L. J. Colwell, and A. B. Wiltschko. Evaluating attribution for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

[14] M. S. Schlichtkrull, N. De Cao, and I. Titov. Interpreting graph neural networks for NLP with differentiable edge masking. In *International Conference on Learning Representations (ICLR)*, 2021.

[15] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: visualising image classification models and saliency maps. In *ICLR Workshop*, 2014.

[16] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

[17] Z. Wu, A. E. Trevino, E. Wu, K. Swanson, H. J. Kim, H. B. D'Angio, R. Preska, G. W. Charville, P. D. Dalerba, A. M. Egloff, R. Uppaluri, U. Duvvuri, A. T. Mayer, and J. Zou. Graph deep learning for the characterization of tumour microenvironments from spatial protein profiles in tissue specimens. *Nature Biomedical Engineering*, 6(12):1435-1448, 2022.

[18] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations (ICLR)*, 2019.

[19] R. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. GNNExplainer: generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[20] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji. On explainability of graph neural networks via subgraph explorations. In *International Conference on Machine Learning (ICML)*, PMLR 139:12241-12252, 2021.