LENS: LEARNABLE EDGE NETWORK SPARSIFICATION FOR INTERPRETABLE HISTOPATHOLOGY

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph neural networks for whole-slide histopathology typically operate on dense spatial graphs that mix diagnostically relevant tissue interfaces with many uninformative connections, increasing compute and obscuring the tissue relationships used for prediction. We introduce **LENS**, a weakly supervised framework for task-driven per-slide topology learning: from slide-level labels alone, a relation-aware edge scorer drives Hard-Concrete ℓ_0 gates under scale-invariant density control. With these gates, **LENS** learns which tissue-patch edges to retain and freezes them at inference into a deterministic sparse adjacency. This sparse adjacency serves as both the computation substrate and a structural explanation, rather than a post-hoc overlay on a dense model. Across 5,530 slides from four lung, kidney, and breast cancer cohorts using three encoders (**SimCLR**, **CTransPath**, **UNI2-h**), **LENS** achieves the most favorable accuracy-sparsity-compute trade-off: it removes $\sim$72% of candidate edges, and its deeper variant **LENS+** attains best or second-best accuracy on all four cohorts, with up to $6.3\times$ lower compute than the strongest dense graph baseline. In a blinded, head-to-head pathologist evaluation against attention-based MIL and dense-graph attribution baselines, **LENS** achieves 97% RoI precision and 92% recall (Dice $= 0.95$), with at least 73% fewer false positives than these baselines, and 78% of retained edges are concentrated in tumor regions without any region-level supervision. Task-supervised topology learning offers a principled mechanism for jointly optimizing efficiency, accuracy, and interpretability in computational pathology. Code and models: https://anonymous.4open.science/r/LENS-014E/.

1 INTRODUCTION

Whole-slide images (WSIs) are naturally represented as spatial graphs of tissue patches, and graph neural networks are widely used for slide-level prediction (Chen et al., 2021; Lu et al., 2022; Zheng et al., 2022). Yet dense k-NN graphs pass messages along every spatial neighbor relation, mixing subtype-discriminative signal with uninformative context and requiring more than 100 GFLOPs per slide in our setting. Multiple instance learning (MIL) attention identifies important patches but not the tissue–tissue relations behind a prediction. Neither paradigm yields a sparse, deterministic topology that a pathologist can audit, which matters for trust and error analysis in clinical settings. The object to be learned is therefore not only a patch score but a task-relevant tissue topology.

What is missing is weakly supervised, task-driven topology learning: deciding, for each slide and from its label alone, which tissue relations are predictive, then freezing this decision at inference. This differs from generic graph sparsification in three ways: each WSI induces a distinct graph of 500 to more than 10,000 nodes, so a single density penalty miscalibrates retention across slides; one slide label must supervise thousands of candidate edges; and the retained structure must serve as a human-auditable explanation, not only as a compression artifact. Existing methods do not meet these constraints jointly: `DropEdge` (Rong et al., 2020) removes edges at random, `GraphLSurv` (Liu et al., 2023) sparsifies with a fixed similarity threshold, `NeuralSparse` (Zheng et al., 2020) and `PTDNet` (Luo et al., 2021) rely on node-level supervision on benchmark graphs, and generic ℓ_0 gates (Louizos et al., 2018) prune shared parameters rather than instance-specific graphs.

We introduce **LENS** (Learnable Edge Network Sparsification), which learns a slide-specific sparse topology through a relation-aware edge scorer trained end-to-end under an expected-ℓ_0 objective with

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

scale-invariant density control. At inference, the gates freeze into a deterministic sparse adjacency that is both the computation substrate and the explanation: a patch with no retained edges cannot influence the prediction, so the heatmap reflects the computation itself rather than a post-hoc overlay. At matched retention, this learned gating outperforms similarity thresholding and random removal, showing that the gains come from edge selection rather than sparsity alone. The framework has two instantiations: **LENS**, optimized for efficiency, and **LENS+**, which adds deeper graph processing and attention pooling for accuracy.

Contributions. (1) We formulate WSI graph sparsification as weakly supervised, instance-specific topology learning and introduce scale-invariant density control that stabilizes edge retention at $28 \pm 2\%$ across folds regardless of graph size (Sec. 3.3). (2) We design a relation-aware two-tower edge scorer that learns channel-selective tissue contrasts from slide-level labels, outperforming cosine-similarity scoring by 4.2 points at matched sparsity and, unlike per-graph mask learning, producing a distinct topology for each unseen slide without retraining (Sec. 3.2). (3) Across 5,530 slides from four lung, kidney, and breast cancer cohorts and three encoders (**UNI2-h**, **CTransPath**, **SimCLR**), **LENS/LENS+** remove $\sim 72\%$ of candidate edges. **LENS+** achieves the highest accuracy on three of four cohorts ($p<0.05$) and surpasses the strongest dense graph baseline at $6.3\times$ lower compute (Sec. 4). (4) In a blinded pathologist evaluation on 10 slides, the topology-derived heatmaps reach 97% RoI precision and 92% recall with at least 73% fewer false positives than MIL and dense-graph attribution baselines, and 78% of retained edges fall within tumor regions without region-level supervision (Sec. 4).

2 Related Work

Graph representations for whole-slide images. Spatial graphs over tissue patches have become a widely used structure for WSI analysis. **Patch-GCN** (Chen et al., 2021) showed that k-NN spatial message passing improves survival prediction over topology-agnostic MIL aggregation, and **SlideGraph+** (Lu et al., 2022) extended spatial connectivity to joint feature-based graphs for **HER2** biomarker prediction. **GTP** (Zheng et al., 2022) combines a GCN with differentiable MinCut pooling and Transformer blocks, generating post-hoc **GraphCAM** attribution maps over the dense graph. Hierarchical variants (Guo et al., 2023; Pati et al., 2022; Shi et al., 2023) introduce multi-scale pooling and cross-resolution attention to capture both local cytological and global architectural context. Despite this architectural diversity, most methods either fix the topology at construction or adapt it via soft weights recomputed at each forward pass, while interpretation is typically post-hoc and decoupled from the topology driving computation. The explanation therefore reflects which nodes received high attribution signal in a dense forward pass, not which tissue-tissue connections the model actually relied on.

Graph sparsification and adaptive structure learning. General sparsification methods remove edges to reduce noise, over-smoothing, or compute, but they do not directly address the weakly supervised, variable-size WSI setting considered here. **DropEdge** (Rong et al., 2020) applies stochastic edge dropout as regularization without learning which edges to remove. **NeuralSparse** (Zheng et al., 2020) samples task-relevant subgraphs via reinforcement-style feedback and **PTDNet** (Luo et al., 2021) combines edge gates with a nuclear-norm regularizer for low-rank adjacency, but both assume homogeneous benchmark graphs with fixed node counts. In computational pathology, sparsification takes two distinct forms. MIL methods (Ilse et al., 2018; Lu et al., 2021; Shao et al., 2021) concentrate attention on a subset of patches, inducing a sparse bag-level representation. **SCMIL** (Yang et al., 2024) restricts self-attention to task-relevant patch clusters via a learnable SoftFilter for survival prediction, but this is node-level: it identifies important patches without modeling tissue-tissue interactions. Graph-level sparsification in WSIs is rare. **GraphLSurv** (Liu et al., 2023) thresholds cosine similarities at a fixed δ and routes interactions through anchor nodes for scalability, producing adaptive structure driven by a hyperparameter rather than task supervision. **SparseXMIL** (Le Bescond et al., 2026) leverages sparse convolutions for memory-efficient WSI classification but does not produce a per-slide frozen topology suitable for case-level inspection. Recent attention de-sparsification work (Kapse et al., 2024) shows that overly aggressive sparsification can suppress diagnostically important tissue context, underscoring that sparsity must be controlled rather than maximized.

ℓ_0 **regularization and controlled sparsity.** Hard-Concrete gates (Louizos et al., 2018) provide a differentiable surrogate for discrete ℓ_0 penalties with exact-zero gates and closed-form keep probabilities. **`ProbMask`** (Zhou et al., 2021) enforces a global sparsity constraint across layers, and Gallego-Posada et al. (Gallego-Posada et al., 2022) reformulate sparsification as constrained optimization with an explicit target density controlled via dual variables. Dynamic sparse training (Evci et al., 2020; Mocanu et al., 2018) maintains a fixed parameter budget while evolving connectivity during training. These contributions share a common assumption: they operate on a single shared weight matrix pruned once under dense supervision, producing a mask that is static across all inputs. The WSI setting violates this assumption: each slide induces a distinct graph, supervision provides one label per thousands of edges, graph sizes vary by two orders of magnitude so a fixed penalty miscalibrates retention, and the pruned structure must serve as a human-auditable structural explanation rather than an internal compression artifact.

Positioning. **`LENS`** occupies a position in this space that is not addressed by existing methods in combination. Unlike stochastic or heuristic sparsification (**`DropEdge`**, **`GraphLSurv`**), fixed-size benchmark methods (**`NeuralSparse`**, **`PTDNet`**), prior ℓ_0 formulations for shared parameter masks, or post-hoc attribution over dense graphs (**`GTP`**, **`Patch-GCN`**, hierarchical variants), **`LENS`** combines four properties: task-supervised topology learning from a single slide label, inductive sparsification of variable-size WSI graphs, scale-invariant density control, and a frozen topology that is both computation substrate and auditable structural explanation. This combination makes the learned sparse graph inspectable and comparable across cases independently of the prediction head. Thus, the gap is specifically edge-level: prior methods may sparsify parameters, nodes, or attention, but they do not learn a frozen task-supervised tissue-interface graph per slide.

3 METHODOLOGY

We formulate WSI topology learning as instance-specific, weakly supervised edge sparsification. **`LENS`** has three components: a relation-aware edge scorer for variable-size slide graphs (Section 3.2), scale-invariant density control (Section 3.3), and a shared frozen topology used by two backbones and by the heatmap (Section 3.4). Figure 1 gives an overview.

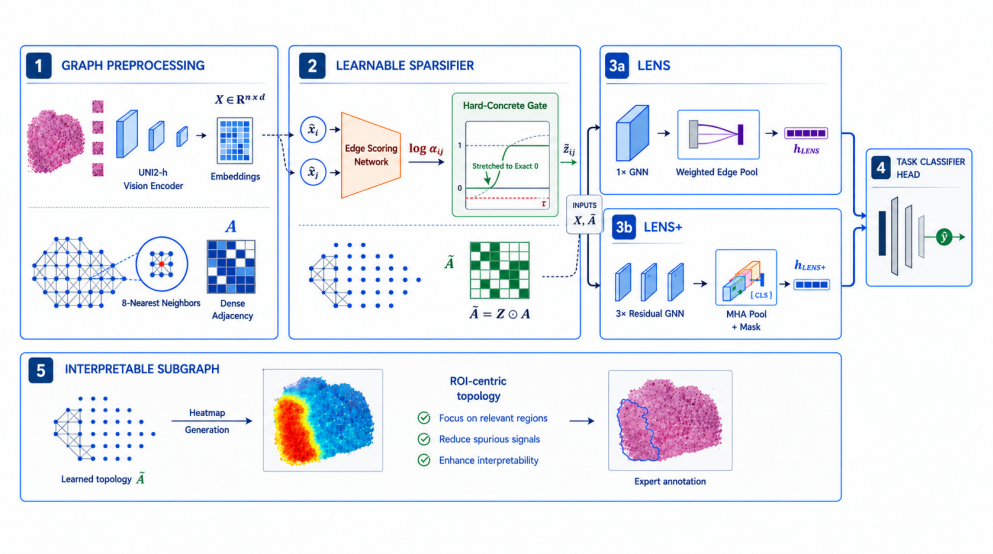

Figure 1: **`LENS` architecture.** A frozen encoder extracts WSI patch features and an 8-NN graph defines candidate tissue relations. The scorer g_φ predicts edge logits, Hard-Concrete gating with density control yields $\tilde{\boldsymbol{A}} = \boldsymbol{Z} \odot \boldsymbol{A}$, and **`LENS/LENS+`** share the resulting topology for prediction. At inference, gates are frozen and heatmaps are derived from retained edge mass; active-node masking excludes zero-edge patches from graph-level prediction.

3.1 FROM WSI TO ℓ_0CONTROLLED SPARSE GRAPHS

Graph representation. Each slide is tiled into n foreground patches $\{p_k\}_{k=1}^{n}$ with coordinates (u_k, v_k). A frozen encoder Φ produces patch features $\boldsymbol{x}_k = \Phi(p_k) \in \mathbb{R}^d$, stacked as $\boldsymbol{X} \in \mathbb{R}^{n\times d}$. We build an 8-nearest-neighbor graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ with adjacency $\boldsymbol{A} \in \{0,1\}^{n\times n}$, treating $\boldsymbol{A}$ as an over-complete pool of candidate tissue relations from which a task-driven subgraph will be learned.

Discrete objective. Given slide label y and binary edge mask $\boldsymbol{S} \in \{0,1\}^{|\mathcal{E}|}$, the ideal objective is

$$\min_{\theta,\,\boldsymbol{S}} \; \ell\Big(y,\; f_\theta\Big(\boldsymbol{X},\; \hat{\boldsymbol{A}}(\boldsymbol{S})\Big)\Big) + \lambda\,\|\boldsymbol{S}\|_0, \tag{3.1}$$

where θ parameterizes the GNN pipeline and $\hat{\boldsymbol{A}}(\boldsymbol{S})$ is the row-normalized masked adjacency. Eq. (3.1) is non-differentiable and NP-hard; we relax it with factorized Bernoulli gates following (Louizos et al., 2018).

Hard-Concrete gating. We use Hard-Concrete gates because they provide stable pathwise gradients, exact-zero gates for true sparsity, and closed-form keep probabilities π_{ij} for density control. ARM attains marginally higher accuracy but requires $2\times$ higher training cost; STE underperforms by 3.2 points at matched cost. We adopt Hard-Concrete as the default (Appendix H.2).

For each candidate edge (i,j), we predict $\log\alpha_{ij} = g_\varphi(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and construct

$$u \sim \mathcal{U}(0,1), \quad s_{ij} = \sigma\left(\frac{\log\alpha_{ij} + \log\frac{u}{1-u}}{\tau}\right), \quad z_{ij} = \mathrm{clip}_{[0,1]}\big(\gamma + (\zeta-\gamma)\,s_{ij}\big). \tag{3.2}$$

Here τ is annealed from 5.0 to 0.67 and $[\gamma,\zeta] = [-0.1,\,1.1]$. The clipped gate can be exactly zero, yielding $\tilde{\boldsymbol{A}} = \boldsymbol{A}\odot\boldsymbol{Z}$ and $\tilde{\mathcal{E}} = \{(i,j)\in\mathcal{E} : z_{ij} > 0\}$. The expected ℓ_0 penalty admits a closed form:

$$\mathbb{E}\big[\|\boldsymbol{S}\|_0\big] = \sum_{(i,j)\,\in\,\mathcal{E}} \pi_{ij}, \qquad \pi_{ij} = \sigma\Big(\log\alpha_{ij} - \tau\log\tfrac{-\gamma}{\zeta}\Big), \tag{3.3}$$

where $\pi_{ij} = P(z_{ij} > 0 \mid \alpha_{ij})$ is the *keep probability* for edge (i,j).

Training and inference. During training, gates are sampled stochastically via Eq. (3.2). At inference, sampling is disabled and z_{ij}^{inf} is computed deterministically from $\log\alpha_{ij}$ using Eq. (3.2) without noise. The resulting frozen sparse adjacency $\tilde{\boldsymbol{A}}^{\text{inf}} = \boldsymbol{A}\odot\boldsymbol{Z}^{\text{inf}}$ is used for all downstream computation.

3.2 RELATION-AWARE EDGE SCORING

The scorer g_φ maps patch pairs to gate logits and is applied independently to each candidate edge on each slide. We use a two-tower architecture that separates channel selection from edge scoring.

Channel-wise importance (attention tower $\texttt{MLP}_\phi$). Given ℓ_2-normalized endpoint features $\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_j$, the attention tower outputs a per-edge, per-dimension importance profile:

$$\boldsymbol{a}_{[i,j]} = \mathrm{softmax}(\texttt{MLP}_\phi([\hat{\boldsymbol{x}}_i \,\|\, \hat{\boldsymbol{x}}_j])) \quad \in \Delta^{d-1}, \tag{3.4}$$

where Δ^{d-1} is the $(d-1)$-simplex. This learns which feature dimensions are diagnostically informative for each patch pair rather than applying a global feature weighting.

Relation-aware contrast. Cosine similarity favors similar patches, yet diagnostic edges often connect dissimilar regions: tumor-stroma invasion, architectural boundaries, or tissue compartments. We therefore use a channel-selective contrast descriptor:

$$\boldsymbol{e}_{[i,j]} = \big[\hat{\boldsymbol{x}}_i \,\|\, \hat{\boldsymbol{x}}_j \,\|\, |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}\big] \in \mathbb{R}^{3d}. \tag{3.5}$$

The attended difference $|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}$ suppresses uninformative feature dimensions while amplifying task-relevant dissimilarities. This formulation outperforms cosine-similarity scoring by 4.2 accuracy points at matched sparsity (Appendix H, Table H.1).

Edge logit (scoring tower $\texttt{MLP}_\psi$). The scoring tower maps $\boldsymbol{e}_{[i,j]}$ to a scalar gate logit:

$$\log\alpha_{ij} = \texttt{MLP}_\psi(\boldsymbol{e}_{[i,j]}) \quad \in \mathbb{R}, \tag{3.6}$$

which parameterizes Eq. (3.2). Because g_φ depends only on patch pairs, it generalizes to unseen slides and produces a distinct topology per WSI. Both orientations of each undirected candidate edge are scored and their gates averaged.

3.3 Scale-Invariant Density Control

A fixed ℓ_0 penalty produces size-dependent retention: without correction, small slides retain 42% of edges while large slides retain 19% (Appendix H, Table H.6). This undermines both efficiency and comparability of the learned topology.

Graph-normalized sparsity. We use the per-graph mean retention $\bar{\pi}_{\mathcal{G}} = \frac{1}{|\mathcal{E}|} \sum_{(i,j) \in \mathcal{E}} \pi_{ij}$ and add an explicit density-matching term to the objective:

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{eff}}(t)\, \bar{\pi}_{\mathcal{G}} + \lambda_{\rho}\, (\bar{\pi}_{\mathcal{G}} - \rho_e)^2, \tag{3.7}$$

where ρ_e is the target retention rate, $\lambda_{\text{eff}}(t)$ the adaptive sparsity weight, and λ_{ρ} the density matching strength. The quadratic term penalizes both over- and under-pruning.

Closed-loop λ adaptation. We adapt the sparsity weight using batch-level feedback:

$$\lambda_{\text{eff}}(t) = \lambda(t) \cdot \text{clip}\big(1 + (\bar{\pi}_{\mathcal{B}}(t) - \rho_e),\ \alpha_{\min},\ \alpha_{\max}\big), \tag{3.8}$$

where $\lambda(t)$ is the scheduled coefficient, $\bar{\pi}_{\mathcal{B}}(t) = \frac{1}{|\mathcal{B}|} \sum_{\mathcal{G} \in \mathcal{B}} \bar{\pi}_{\mathcal{G}}$, and $(\alpha_{\min}, \alpha_{\max}) = (0.2, 2.0)$. The error $\bar{\pi}_{\mathcal{B}}(t) - \rho_e$ increases sparsity pressure when too many edges are retained and relaxes it otherwise. This stabilizes retention to $28 \pm 2\%$, reduces the retention std by $5\times$, and accelerates convergence by $2\times$ (Appendix H, Table H.7).

3.4 Graph Processing, Pooling, and Explanation

Shared sparse substrate. Both variants use $\hat{\boldsymbol{A}} = \text{row_norm}(\tilde{\boldsymbol{A}} + \boldsymbol{I})$: retained edges control inter-patch message passing, self-loops preserve patch representations, and graph-level aggregation uses only active patches.

Active-node mask. For each patch i we compute its *incident retained edge mass*

$$w_i = \sum_{j \in \mathcal{N}(i)} z_{ij}, \qquad \mathcal{N}(i) = \{j : (i,j) \in \tilde{\mathcal{E}}\}, \tag{3.9}$$

and active mask $m_i = \mathbb{I}\{\, w_i > 0 \,\}$. The heatmap maps w_i to slide coordinates. The same retained edge mass w_i therefore determines both graph-level participation and the displayed heatmap.

`LENS` (efficiency-oriented). A single GCN layer is followed by edge-weighted pooling over active nodes: $\eta_i = w_i / (\sum_k w_k + \epsilon)$, with $\eta_i = 0$ when $m_i = 0$.

`LENS+` (accuracy-oriented). Three residual GCN layers produce 3-hop embeddings, followed by masked multi-head attention with a learnable [CLS] token. Patch-patch attention is masked by $\tilde{\boldsymbol{A}}$, and [CLS]-to-patch attention is restricted to active nodes ($m_i = 1$); the [CLS] embedding gives $\boldsymbol{h}_{\text{slide}}$.

Heatmap construction and structural faithfulness. Post-hoc attribution methods compute explanations after the forward pass on a dense topology, meaning the explanation is decoupled from the structure the model computed on. **`LENS`** itself derives the heatmap directly from the retained edge mass of the sparse topology used in the forward pass. With active-node masking, a patch with $w_i = 0$ is excluded from graph-level aggregation and [CLS] attention and therefore cannot influence the slide prediction. The heatmap therefore visualizes the active sparse computation substrate rather than a post-hoc attribution layered over a dense topology.

Training. Both variants are trained end-to-end with Eq. (3.7) under slide-level supervision, with the shared scorer g_{φ} and its gates learned jointly with the backbone. Full architectural details and hyperparameters are in Appendix D.

4 Results and Analysis

Setup. Four public H&E benchmarks (5,530 slides, 3,340 patients): CPTAC-Lung (3-class, n=2,071), TCGA-LUNG (binary, n=1,548), TCGA-RCC (3-class, n=878), TCGA-BRCA (3-class, n=1,033). All methods share identical **`UNI2-h`** features (Chen et al., 2024) (ViT-H/14, 1536-dim, 256×256 at $20\times$), 8-NN spatial graphs, and patient-stratified 5-fold cross-validation with a three-way split: training, validation for model selection, and a held-out test set from which all reported metrics are

computed. Results are reported as mean ± std over 15 runs (five patient-stratified folds × three seeds: 42, 123, 2024), using the validation-selected checkpoint evaluated once on each held-out test fold. Significance is assessed with a two-sided paired t-test over the 15 runs, paired by fold and seed. Eight baselines span three paradigms: MIL (**ABMIL**, **TransMIL**, **CLAM-SB/MB**), dense GNNs (**GCN**, **Patch-GCN**, **GTP**), and sparse GNNs (**GraphLSurv**), each tuned independently on the validation set (details in Appendix F). This protocol isolates the effect of topology learning and edge selection, since features, graph construction, data splits, and model selection are fixed across methods.

Table 1: **Primary CPTAC-Lung benchmark using UNI2-h, n=2,071 slides.** Accuracy, AUROC, edge retention, GFLOPs, efficiency ratio, and interpretability mechanism for MIL, dense GNN, sparse GNN, and **LENS**-family models under the same 8-NN graph construction and patient-stratified 5-fold evaluation over 3 seeds (mean$_{\pm\text{std}}$ over 15 runs). Bold/underline indicate best/second-best. $^{\star}p<0.05$, $^{\star\star}p<0.01$: **LENS+** significantly higher (paired t-test over 15 runs, paired by fold and seed).

Method	Acc (%)	AUROC (%)	Edge Ret. (%)	GFLOPs	Eff. Ratio	Interp.
Multiple Instance Learning						
ABMIL	$85.1^{\star\star}_{\pm 1.6}$	$94.2^{\star\star}_{\pm 1.0}$	N/A	29.6	2.87	Post-hoc attn
TransMIL	$86.8^{\star\star}_{\pm 1.3}$	$95.2^{\star\star}_{\pm 0.9}$	N/A	38.4	2.26	Post-hoc attn
CLAM-SB	$87.5^{\star\star}_{\pm 1.5}$	$96.0_{\pm 0.7}$	N/A	31.7	2.76	Instance attn
CLAM-MB	$88.3^{\star\star}_{\pm 1.4}$	$95.7_{\pm 0.6}$	N/A	32.9	2.68	Instance attn
Dense Graph Neural Networks						
GCN	$82.4^{\star\star}_{\pm 2.0}$	$90.8^{\star\star}_{\pm 1.6}$	100	103.7	0.79	None
Patch-GCN	$85.0^{\star\star}_{\pm 1.9}$	$92.5^{\star\star}_{\pm 1.5}$	100	178.4	0.48	Global attn
GTP	$\underline{90.5}^{\star}_{\pm 1.3}$	$\underline{96.5}^{\star}_{\pm 0.6}$	100	152.2	0.59	GraphCAM
Sparse Graph Neural Networks						
GraphLSurv	$84.2^{\star\star}_{\pm 1.8}$	$92.1^{\star\star}_{\pm 1.3}$	$\sim$45	25.5	3.30	Learned structure
LENS	$88.4_{\pm 1.3}$	$96.3_{\pm 0.7}$	$\sim$**28**	**19.2**	**4.60**	Intrinsic edge gates
LENS+	$\mathbf{91.2}_{\pm 1.1}$	$\mathbf{97.7}_{\pm 0.4}$	$\sim\underline{29}$	$\underline{24.1}$	$\underline{3.78}$	Intrinsic edge gates

Table 2: **Multi-dataset evaluation on three TCGA cohorts using UNI2-h features.** Accuracy and AUROC on TCGA-LUNG, TCGA-RCC, and TCGA-BRCA under the same evaluation protocol (patient-stratified 5-fold evaluation over 3 seeds; mean$_{\pm\text{std}}$ over 15 runs). **LENS+** achieves best or second-best performance across the three cohorts, while the sparsity and compute characteristics of the **LENS** family are reported in Table 1 because edge retention and GFLOPs are architecture-dependent. Bold/underline indicate best/second-best. $^{\star}p<0.05$, $^{\star\star}p<0.01$

	TCGA-LUNG (n=1,548)		**TCGA-RCC** (n=878)		**TCGA-BRCA** (n=1,033)	
Method	Acc	AUC	Acc	AUC	Acc	AUC
Multiple Instance Learning						
ABMIL	$78.4^{\star\star}_{\pm 1.7}$	$87.1^{\star\star}_{\pm 1.3}$	$87.8^{\star\star}_{\pm 1.5}$	$95.4^{\star\star}_{\pm 0.9}$	$83.9^{\star\star}_{\pm 1.6}$	$93.1^{\star\star}_{\pm 1.1}$
TransMIL	$79.0^{\star\star}_{\pm 1.4}$	$87.8^{\star\star}_{\pm 1.2}$	$88.6^{\star\star}_{\pm 1.3}$	$96.0^{\star\star}_{\pm 0.8}$	$84.3^{\star\star}_{\pm 1.4}$	$92.8^{\star\star}_{\pm 1.0}$
CLAM-SB	$80.5^{\star\star}_{\pm 1.5}$	$89.2^{\star\star}_{\pm 1.0}$	$89.8^{\star\star}_{\pm 1.2}$	$96.5^{\star\star}_{\pm 0.7}$	$85.8^{\star}_{\pm 1.3}$	$93.8^{\star\star}_{\pm 0.9}$
CLAM-MB	$81.2^{\star\star}_{\pm 1.3}$	$89.8^{\star\star}_{\pm 0.9}$	$90.3^{\star\star}_{\pm 1.1}$	$97.0^{\star\star}_{\pm 0.6}$	$86.5_{\pm 1.2}$	$94.5_{\pm 0.8}$
Dense Graph Neural Networks						
GCN	$77.2^{\star\star}_{\pm 2.1}$	$86.5^{\star\star}_{\pm 1.7}$	$84.1^{\star\star}_{\pm 2.0}$	$93.2^{\star\star}_{\pm 1.4}$	$79.5^{\star\star}_{\pm 2.2}$	$89.6^{\star\star}_{\pm 1.6}$
Patch-GCN	$78.8^{\star\star}_{\pm 1.9}$	$87.4^{\star\star}_{\pm 1.5}$	$86.0^{\star\star}_{\pm 1.8}$	$94.8^{\star\star}_{\pm 1.2}$	$81.8^{\star\star}_{\pm 2.0}$	$91.2^{\star\star}_{\pm 1.4}$
GTP	$\underline{82.1}^{\star\star}_{\pm 1.2}$	$\underline{90.5}^{\star\star}_{\pm 0.8}$	$89.8^{\star\star}_{\pm 1.3}$	$96.7^{\star\star}_{\pm 0.7}$	$\mathbf{88.2}_{\pm 1.1}$	$\mathbf{95.6}_{\pm 0.7}$
Sparse Graph Neural Networks						
GraphLSurv	$77.8^{\star\star}_{\pm 2.0}$	$86.9^{\star\star}_{\pm 1.5}$	$85.2^{\star\star}_{\pm 1.9}$	$94.0^{\star\star}_{\pm 1.3}$	$78.6^{\star\star}_{\pm 2.1}$	$88.4^{\star\star}_{\pm 1.5}$
LENS	$81.5^{\star\star}_{\pm 1.4}$	$90.1^{\star\star}_{\pm 0.9}$	$\underline{91.6}_{\pm 1.1}$	$\underline{97.5}^{\star}_{\pm 0.5}$	$84.2^{\star\star}_{\pm 1.5}$	$92.8^{\star\star}_{\pm 1.0}$
LENS+	$\mathbf{84.2}_{\pm 1.1}$	$\mathbf{92.1}_{\pm 0.7}$	$\mathbf{92.7}_{\pm 0.9}$	$\mathbf{98.2}_{\pm 0.4}$	$\underline{87.6}_{\pm 1.2}$	$\underline{95.4}_{\pm 0.7}$

Sparsity as signal, not sacrifice. Table 1 presents the primary benchmark. **LENS+** surpasses the strongest dense graph baseline (GTP: 91.2 vs. 90.5, $p<0.05$) while retaining only $\sim$29% of candidate edges at 6.3$\times$ lower compute, and with the highest AUROC (97.7%). Dense 8-NN graphs route messages through every tissue interface, mixing subtype-discriminative patterns with irrelevant context. Task-supervised gating suppresses these dilutive pathways, concentrating information flow along boundaries the classification objective identifies as predictive. The comparison with **GraphLSurv** sharpens the point: both methods sparsify, but end-to-end learned gating outperforms cosine-similarity thresholding at even lower retention. Figure 2A captures this spatially: **LENS** and **LENS+** occupy the Pareto-favorable high-accuracy, low-compute region, while **GTP** reaches comparable accuracy only at substantially higher cost.

Consistent multi-dataset performance. Table 2 extends evaluation to three TCGA cohorts. **LENS+** achieves best or second-best performance on all four datasets. On TCGA-LUNG and TCGA-RCC, **LENS+** significantly outperforms all baselines ($p<0.01$). On TCGA-RCC, where renal subtypes exhibit regular compartmentalized architecture, **LENS+** leads by the widest margin. On TCGA-BRCA, **LENS+** is on par with **GTP** at $\sim$6$\times$ lower compute.

Accuracy–compute–sparsity trade-off. Figure 2 projects the full design space onto accuracy and compute (panel A) and accuracy and sparsity (panel B). Panel B is the most telling: MIL and dense GNN methods do not explicitly sparsify the graph, **GraphLSurv** provides moderate sparsity at lower accuracy, and **LENS/LENS+** combine $>$ 70% edge removal with high accuracy. The **LENS** family occupies a favorable region of the design space that dense and heuristic sparse baselines do not reach. The key gain is high accuracy together with sparse, inspectable computation.

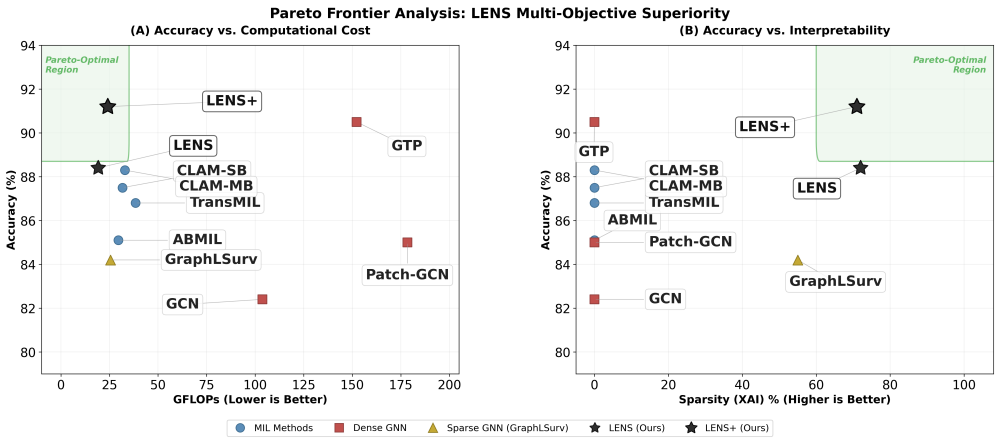

Figure 2: **Pareto frontiers on CPTAC-Lung (UNI2-h). (A)** Accuracy vs. compute. **(B)** Accuracy vs. sparsity. Upper-left is better in panel A; upper-right is better in panel B. Green shading marks the Pareto-favorable region. **LENS/LENS+** occupy the most favorable accuracy–compute and accuracy–sparsity trade-off regions. Markers: ○ MIL; □ dense GNN; △ sparse GNN; ⋆ **LENS/LENS+**.

Expert-aligned interpretability. The frozen sparse graph provides a structural explanation grounded in the computation itself. A blinded expert-alignment study with one board-certified pathologist (15+ years of experience) on 10 CPTAC-Lung WSIs (5 LUAD / 5 LSCC) quantifies this alignment. Table 3 reports 97.1% RoI precision, 92.4% recall, and a false-positive rate of 2.9%, reducing false activations by 73% relative to **GTP** and 76% relative to **CLAM**. Performance is stable across subtypes (LUAD: 97.8%; LSCC: 96.4%), suggesting that the edge scorer adapts sparsification to subtype-specific morphology rather than applying a uniform pruning mask. This single-expert, ten-slide study constitutes structural alignment evidence rather than clinical validation; prospective multi-reader evaluation is required before clinical use can be assessed, and lies outside the scope of our study.

Is learned gating reducible to thresholding? Table 4 tests this with two controlled conditions at identical $\sim$28% retention: cosine-similarity thresholding (`Threshold-28`) and uniform random removal (`Random-28`). Because retention and backbone are matched, this isolates the edge-selection mechanism. On classification, `Threshold-28` underperforms **LENS** by 3.8 points and `Random-28` by 6.4 points. On expert alignment, `Threshold-28` achieves 88% precision but

Table 3: **Expert alignment on CPTAC-Lung** (n=10 slides, blinded). All methods are evaluated against the same expert RoIs using binarized heatmaps; mean$_{\pm\text{std}}$ over slides. FP = highlighted area outside the expert RoI, as a percentage of tissue area. $^{\star\star}p<0.01$ vs. **LENS** (paired t-test).

	Prec. (%) ↑	Rec. (%) ↑	IoU ↑	Dice ↑	FP (%) ↓
LENS	$\mathbf{97.1}_{\pm 2.8}$	$\mathbf{92.4}_{\pm 4.1}$	$\mathbf{0.90}_{\pm .04}$	$\mathbf{0.95}_{\pm .02}$	$\mathbf{2.9}_{\pm 2.1}$
GTP	$78.4^{\star\star}_{\pm 15.2}$	$85.7_{\pm 9.6}$	$0.67^{\star\star}_{\pm .11}$	$0.80^{\star\star}_{\pm .08}$	$10.8^{\star\star}_{\pm 6.3}$
CLAM	$76.2^{\star\star}_{\pm 8.8}$	$83.9_{\pm 10.2}$	$0.65^{\star\star}_{\pm .13}$	$0.79^{\star\star}_{\pm .09}$	$12.1^{\star\star}_{\pm 5.2}$

Table 4: **Sparsification mechanism ablation, CPTAC-Lung (UNI2-h).** Same backbone and ∼28% retention across all sparse conditions; only the edge selection mechanism varies. Blinded expert alignment (n=10, right) are reported jointly.

	Classification		Expert Alignment			
Edge selection	**Acc (%) ↑**	**AUROC (%) ↑**	**Prec. (%) ↑**	**Rec. (%) ↑**	**Dice ↑**	**FP (%) ↓**
`Random-28`	$82.0_{\pm 2.2}$	$90.4_{\pm 1.6}$	$54.2_{\pm 11.5}$	$58.6_{\pm 12.0}$	$0.56_{\pm .10}$	$31.4_{\pm 9.8}$
`Threshold-28`	$84.6_{\pm 1.7}$	$92.8_{\pm 1.2}$	$88.0_{\pm 5.4}$	$72.0_{\pm 8.3}$	$0.79_{\pm .06}$	$8.1_{\pm 3.9}$
LENS (learned ℓ_0)	$\mathbf{88.4}_{\pm 1.3}$	$\mathbf{96.3}_{\pm 0.7}$	$\mathbf{97.1}_{\pm 2.8}$	$\mathbf{92.4}_{\pm 4.1}$	$\mathbf{0.95}_{\pm .02}$	$\mathbf{2.9}_{\pm 2.1}$

only 72% recall, consistent with conservative pruning. **LENS** achieves both simultaneously (97.1% / 92.4%, dice=0.95), a balance not reproduced by the matched-retention alternatives tested.

Controlled ablations. These experiments test whether the learned topology, not just features or architecture, drives performance. They isolate interactions, geometry, encoder, and connectivity.

(i) Zero edges. Removing all edges reduces **LENS** to independent patch scoring. Accuracy falls to 81.4% (**LENS+**: 91.2% → 82.0%), below every MIL baseline (Table 5). Inter-patch interactions, not only the classifier head, account for the gains. This supports the sparse graph's role in prediction-relevant message passing.

Table 5: **Topology ablation, CPTAC-Lung (UNI2-h).** Accuracy (%). Drop relative to the original graph in parentheses. Removing edges tests interactions; permuting coordinates tests tissue geometry.

Setting	**LENS**	**LENS+**
Original spatial graph	$\mathbf{88.4}_{\pm 1.3}$	$\mathbf{91.2}_{\pm 1.1}$
Zero edges (no interactions)	$81.4_{\pm 1.8}$ (−7.0)	$82.0_{\pm 1.7}$ (−9.2)
Permuted coordinates	$79.2_{\pm 2.4}$ (−9.2)	$81.8_{\pm 2.2}$ (−9.4)

(ii) Spatial permutation. Shuffling patch coordinates while preserving features and rebuilding the 8-NN graph drops accuracy by 9.2 points for **LENS** and 9.4 for **LENS+**. The learned graph fragments under permutation (mean component size: 56 vs. 847 patches), and the topology-derived heatmap becomes diffuse across the slide (Appendix I.3). This shows that **LENS** relies on tissue geometry in addition to patch features.

(iii) Encoder robustness. Replacing **UNI2-h** with **CTransPath** and **SimCLR** preserves rank ordering across all methods and datasets, with **LENS+** achieving best or second-best accuracy under all three encoders and sparsity stable at ∼28–29% retention regardless of encoder choice (Appendix I.1).

(iv) Candidate graph construction. k=8 gives the best observed accuracy–compute trade-off; smaller k creates coverage gaps that the scorer cannot recover, while larger k adds candidates that are gated away without accuracy gain (Appendix I.2).

Additional component ablations (temperature scheduling, edge-scorer architecture, pooling, size-aware regularization, and target density) are in Appendix H (Tables H.1–H.8); each contributes 1–4 points. Together, these ablations attribute the gains to the full topology-learning design.

What the sparse graph discovers. Edge-level analysis on annotated slides reveals that 78% of retained edges fall within expert-delineated tumor, 15% connect tumor–stroma interfaces, and 7% lie in non-diagnostic tissue, versus a near-uniform split in the dense graph (33/27/40%). This concentration emerges from slide-level labels alone: the ℓ_0 objective, optimizing only for classification, enriches the learned topology for diagnostically relevant spatial organization as a structural byproduct. Critically, this redistribution is not reproduced by matched-retention thresholding of the dense graph: the similarity-threshold control preserves a near-uniform spatial distribution, whereas task-supervised gating qualitatively restructures which tissue interfaces are retained. Figure G.2 visualizes this on a representative LUAD slide, contrasting the uniform 11,652-edge dense graph used by **GTP** with the task-supervised 2,210-edge sparse topology learned by **LENS**.

5 Discussion

Why the learned topology matters. The ablations point to the learned topology, and not only the features or the classifier, as the source of the gains. Removing all edges drops accuracy below every MIL baseline, so interactions between patches matter. Shuffling patch positions hurts even more, so the spatial layout of the kept edges matters, not just their number. Replacing the encoder leaves the ranking largely unchanged, so the gains are not tied to one feature extractor. Retention also affects accuracy: a dense version of **LENS** is 2.3 points lower (86.1% vs. 88.4%, Table H.8). What the ℓ_0 objective adds is a much smaller graph that keeps the predictive relations, most of them inside tumor.

Generalization and morphological structure. Retention stays at $\sim$28–29% across three encoders, four cohorts, and graph sizes from 500 to over 10,000 nodes. Gains are largest on TCGA-RCC, where renal subtypes have a regular, repetitive architecture, and smallest on TCGA-BRCA, where morphology is more heterogeneous. The same sparsification mechanism also works for survival prediction: we only replace the classification head with a discrete-time hazard head. On TCGA-BRCA, **LENS+** reaches the highest C-index (0.653) with a smaller fold std than the survival-native graph baselines (0.018 against 0.045 and 0.066); on TCGA-BLCA all methods stay close to chance (Appendix N).

Interpretability: scope of evidence. The heatmap comes from the same sparse graph used for prediction, so it reflects the model's computation rather than a post-hoc overlay. Its agreement with expert annotations (97.1% RoI precision, 0.90 IoU) and the concentration of kept edges in tumor (78%) show that the learned structure follows pathology-relevant tissue. A comparison with the evaluation protocols of prior work is given in Appendix J.1.

Limitations. (i) The pathologist study uses one expert and ten slides. It shows structural alignment, not clinical validity. Multi-reader, multi-site studies are needed to measure inter-reader variability, robustness to scanner and stain shift, and clinical utility. (ii) Single magnification (20×); multi-scale hierarchical graphs are a natural extension. (iii) Gate decisions are point estimates with no calibrated uncertainty, which limits auditability in low-confidence cases. (iv) H&E only; multimodal fusion (genomics, IHC) through cross-modal edge gating could recover subtypes where morphology is insufficient (Appendix K).

6 Conclusion

We formulated WSI topology learning as instance-specific, weakly supervised edge sparsification and instantiated it in **LENS**. Across 5,530 slides and three encoders (**SimCLR**, **CTransPath**, **UNI2-h**), **LENS** and **LENS+** remove $\sim$72% of candidate edges. **LENS+** reaches best or second-best accuracy on all four cohorts, at up to 6.3× lower compute than the strongest dense graph baseline. The same mechanism transfers to survival prediction without changes to the sparsification. Blinded expert alignment (97% RoI precision), matched-retention controls, and the distribution of kept edges support one conclusion: the gains come from learned topology, not from post-hoc thresholding or sparsity alone. The frozen sparse graph is both the computation substrate and the explanation. **LENS** remains a research tool, and its interpretability evidence is structural alignment until prospective clinical validation is done. More broadly, weakly supervised topology learning turns variable-size instance graphs into sparse, auditable computation structures.

AI USE STATEMENT

All research ideas, theoretical contributions, experimental designs, implementations, and interpretations of results presented in this work are entirely our own. Large language models were not used as an important, original, or non-standard component of the core methods of this research. Instead, their use was limited to preparing internal working documents, as well as aiding and polishing writing, assisting with LaTeX formatting, and identifying related work—all of which we have independently verified. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We have taken deliberate steps to make every claim in this paper independently verifiable. The method is specified in full in Section 3. The ℓ_0 relaxation, the gating construction, and their derivations, including all assumptions, are given in Appendix B, and the edge scorer in Appendix C. Pseudocode for every component and for the end-to-end pipeline is provided in Appendix L. The complete architecture and training hyperparameters for **LENS** and **LENS+** are listed in Appendix D (Tables D.1–D.2), so that both models can be rebuilt from the paper alone. All four datasets are publicly available: CPTAC-Lung through TCIA, and TCGA-LUNG, TCGA-RCC, and TCGA-BRCA through the GDC. The constituent project codes, cohort composition, and slide inclusion criteria are documented in Appendix E.1, and the full feature-extraction and graph-construction pipeline in Appendix E.2. The evaluation protocol (patient-stratified 5-fold cross-validation with disjoint three-way splits, repeated over three seeds: 42, 123, 2024) is described in Appendix E.3. Baseline configurations and their per-method tuning budget are given in Appendix F, and the annotation protocol for the expert-alignment study in Appendix J. All methods are compared under identical conditions: the same saved fold assignments, the same precomputed features, and the same fixed random seeds. Reported differences are therefore attributable to the methods themselves rather than to data partitioning or feature extraction. Hardware, software versions, and runtimes are reported in Appendix E.4, and a consolidated reproducibility checklist in Appendix F.4. Source code, configuration files, and pretrained checkpoints sufficient to reproduce every table and figure are available in an anonymous repository at `https://anonymous.4open.science/r/LENS-014E/`; the GitHub repository will be de-anonymized upon acceptance.

REFERENCES

Damien Barbier, Carlo Lucibello, Luca Saglietti, Florent Krzakala, and Lenka Zdeborová. Compressed sensing with ℓ_0-norm: Statistical physics analysis & algorithms for signal recovery. In *IEEE Information Theory Workshop (ITW)*, pp. 323–328, 2023.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Thomas Bird, Julius Kunze, and David Barber. Stochastic variational optimization. *arXiv preprint arXiv:1809.04855*, 2018.

Richard J. Chen, Ming Y. Lu, Muhammad Shaban, Chengkuan Chen, Tiffany Y. Chen, Drew F. K. Williamson, and Faisal Mahmood. Whole slide images are 2d point clouds: Context-aware survival prediction using patch-based graph convolutional networks. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, volume 12908 of *Lecture Notes in Computer Science*, pp. 339–349. Springer, Cham, 2021. doi: 10.1007/978-3-030-87237-3_33.

Richard J. Chen, Tong Ding, Ming Y. Lu, Drew F. K. Williamson, Guillaume Jaume, Andrew H. Song, Bowen Chen, Andrew Zhang, Daniel Shao, Muhammad Shaban, et al. Towards a general-purpose foundation model for computational pathology. *Nature Medicine*, 30(3):850–862, 2024. doi: 10.1038/s41591-024-02857-3.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 19314–19326, 2020.

Bao Duong, Hung Le, Biwei Huang, and Thin Nguyen. Reinforcement learning for causal discovery without acyclicity constraints. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=sNzBi8rZTy.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pp. 2943–2952. PMLR, 2020.

Utku Evci, Yani Ioannou, Cem Keskin, and Yann Dauphin. Gradient flow in sparse neural networks and how lottery tickets win. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 6577–6586, 2022.

Christian PC Franssen, Jinyang Jiang, Yijie Peng, and Bernd Heidergott. Connect: Connectivity-based regularization for structural pruning of neural networks. *Transactions on Machine Learning Research*, 2025.

Jose Gallego-Posada, Juan Ramirez, Akram Erraqabi, Yoshua Bengio, and Simon Lacoste-Julien. Controlled sparsity via constrained optimization or: How i learned to stop tuning penalties and love constraints. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 1253–1266. Curran Associates, Inc., 2022.

Ziyu Guo, Weiqin Zhao, Shujun Wang, and Lequan Yu. Higt: Hierarchical interaction graph-transformer for whole slide image analysis. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 755–764. Springer, 2023.

Hussein Hazimeh, Rahul Mazumder, and Tim Nonet. L0learn: A scalable package for sparse learning using l0 regularization. *Journal of Machine Learning Research*, 24(205):1–8, 2023.

Konstantin Hemker, Nikola Simidjievski, and Mateja Jamnik. HEALNet: Multimodal fusion for heterogeneous biomedical data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2057.

Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pp. 2127–2136. PMLR, 2018.

Saarthak Kapse, Srijan Das, Jingwei Zhang, Rajarsi R Gupta, Joel Saltz, Dimitris Samaras, and Prateek Prasanna. Attention de-sparsification matters: Inducing diversity in digital pathology representation learning. *Medical image analysis*, 93:103070, 2024.

Yeskendir Koishekenov. Reducing over-smoothing in graph neural networks using relational embeddings. *arXiv preprint arXiv:2301.02924*, 2023.

Loïc Le Bescond, Marvin Lerousseau, Fabrice André, and Hugues Talbot. Sparsexmil: Leveraging sparse convolutions for context-aware and memory-efficient classification of whole slide images in digital pathology. *Medical Image Analysis*, 111:104047, 2026. doi: 10.1016/j.media.2026.104047.

Pei Liu, Luping Ji, Feng Ye, and Bo Fu. Graphlsurv: A scalable survival prediction network with adaptive and sparse structure learning for histopathological whole-slide images. *Computer Methods and Programs in Biomedicine*, 231:107433, 2023.

Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through L_0 regularization. In *Advances in Neural Information Processing Systems*, pp. 6479–6489, 2017.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l_0 regularization. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/pdf?id=H1Y8hhg0b.

Ming Y Lu, Drew FK Williamson, Tiffany Y Chen, Richard J Chen, Matteo Barbieri, and Faisal Mahmood. Data-efficient and weakly supervised computational pathology on whole-slide images. *Nature biomedical engineering*, 5(6):555–570, 2021.

Ming Y Lu, Bowen Chen, Drew F K Williamson, Richard J Chen, Ivy Liang, Tong Ding, Guillaume Jaume, Igor Odintsov, Long Phi Le, Georg Gerber, Anil V Parwani, Andrew Zhang, and Faisal Mahmood. A visual-language foundation model for computational pathology. *Nature Medicine*, 30(3):863–874, March 2024. doi: 10.1038/s41591-024-02856-4.

Wenqi Lu, Michael Toss, Muhammad Dawood, Emad Rakha, Nasir Rajpoot, and Fayyaz Minhas. Slidegraph+: Whole slide image level graphs to predict her2 status in breast cancer. *Medical Image Analysis*, 80:102486, 2022.

Dongsheng Luo, Wei Cheng, Wenchao Yu, Bo Zong, Jingchao Ni, Haifeng Chen, and Xiang Zhang. Learning to drop: Robust graph neural network via topological denoising. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 779–787, 2021.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):2383, 2018.

Balas K Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24 (2):227–234, 1995.

Pushpak Pati, Guillaume Jaume, Antonio Foncubierta-Rodriguez, Florinda Feroce, Anna Maria Anniciello, Giosuè Scognamiglio, Nadia Brancati, Maria Foncubierta, Maria Gabrani, Orcun Goksel, et al. Hierarchical graph representations in digital pathology. *Medical Image Analysis*, 75: 102264, 2022.

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

Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, et al. TransMIL: Transformer based correlated multiple instance learning for whole slide image classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 2136–2147, 2021.

Jiangbo Shi, Lufei Tang, Zeyu Gao, Yang Li, Chunbao Wang, Tieliang Gong, Chen Li, and Huazhu Fu. Mg-trans: Multi-scale graph transformer with information bottleneck for whole slide image classification. *IEEE Transactions on Medical Imaging*, 42(12):3871–3883, December 2023. doi: 10.1109/TMI.2023.3313252.

Emmanuel Soubies, Laure Blanc-Féraud, and Gilles Aubert. A continuous exact ℓ_0 penalty (cel0) for least squares regularized problem. *SIAM Journal on Imaging Sciences*, 8(3):1607–1639, 2015.

Emmanuel Soubies, Laure Blanc-Féraud, and Gilles Aubert. A unified view of exact continuous penalties for ℓ_2–ℓ_0 minimization. *SIAM Journal on Optimization*, 27(3):2034–2060, 2017.

Yixin Su, Rui Zhang, Sarah M. Erfani, and Zhenghua Xu. Detecting beneficial feature interactions for recommender systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4357–4365, 2021.

Dongfang Sun and Yingzhen Yang. Locally regularized sparse graph by fast proximal gradient descent. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 2069–2077, 2023.

Seiya Tokui and Kenta Oono. Evaluating the variance of likelihood-ratio gradient estimators. *International Conference on Machine Learning*, pp. 3414–3423, 2017.

William D Travis, Elisabeth Brambilla, Allen P Burke, Alexander Marx, and Andrew G Nicholson. WHO classification of tumours of the lung, pleura, thymus and heart. *International Agency for Research on Cancer*, 2015.

Abhishek Tyagi, Arjun Iyer, William H Renninger, Christopher Kanan, and Yuhao Zhu. Dynamic sparse training of diagonally sparse networks. *arXiv preprint arXiv:2506.11449*, 2025.

Eugene Vorontsov, Alican Bozkurt, Adam Casson, George Shaikovski, Michal Zelechowski, Kristen A Severson, Eric Zimmermann, James M Hall, Neil Tenenholtz, Nicolò Fusi, et al. A foundation model for clinical-grade computational pathology and rare cancers detection. *Nature Medicine*, 30(10):2924–2935, 2024. doi: 10.1038/s41591-024-03141-0. URL https://doi.org/10.1038/s41591-024-03141-0.

Xi Wang, Hao Chen, Caixia Geng, Yu Wang, Rui Gao, Xuming Song, Man Yu, Shusheng Gao, Jun Kong, Fusheng Wang, et al. Ctranspath: A transformer-based model to integrate whole slide images and associated clinical data for cancer diagnosis. *Medical Image Analysis*, 80:102508, 2022.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Antonio C Wolff, M Elizabeth H Hammond, Kimberly H Allison, Beth E Harvey, Pamela B Mangu, John MS Bartlett, Michael Bilous, Ian O Ellis, Patrick Fitzgibbons, Wedad Hanna, et al. Human epidermal growth factor receptor 2 testing in breast cancer: American Society of Clinical Oncology/College of American Pathologists clinical practice guideline focused update. *Journal of Clinical Oncology*, 36(20):2105–2122, 2018.

Boqian Wu, Qiao Xiao, Shunxin Wang, Nicola Strisciuglio, Mykola Pechenizkiy, Maurice van Keulen, Decebal Constantin Mocanu, and Elena Mocanu. Dynamic sparse training versus dense training: The unexpected winner in image corruption robustness. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://arxiv.org/abs/2410.03030.

Zekang Yang, Hong Liu, and Xiangdong Wang. Scmil: Sparse context-aware multiple instance learning for predicting cancer survival probability distribution in whole slide images. In *arXiv preprint arXiv:2407.00664*, 2024.

Penghang Yin, Shuai Zhang, Jiancheng Lyu, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations*, 2019.

Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pp. 11458–11468. PMLR, 2020.

Yi Zheng, Rushin H Gindra, Emily J Green, Eric J Burks, Margrit Betke, Jennifer E Beane, and Vijaya B Kolachalama. A graph-transformer for whole slide image classification. *IEEE transactions on medical imaging*, 41(11):3003–3015, 2022.

Xiao Zhou, Weizhong Zhang, Hang Xu, and Tong Zhang. Effective sparsification of neural networks with global sparsity constraint. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3599–3608, 2021.

APPENDIX CONTENTS

A Additional Related Work on ℓ_0-Sparsification and Graph Models **15**

B ℓ_0-Relaxation and Hard–Concrete Gates **16**

B.1 ℓ_0-Regularized mask learning and variational relaxation 17

B.2 Reparameterized edge-gates with Hard–Concrete and expected ℓ_0 17

B.3 End-to-end expected-ℓ_0 objective and training vs inference 18

C Edge-Scoring Network g_φ **18**

C.1 Preprocessing of pairwise inputs 19

C.2 Feature-importance attention 19

C.3 Relation-aware contrast and edge-feature assembly 19

C.4 Edge-importance head and two-tower formulation 20

C.5 Coupling with Hard-Concrete gating and graph aggregation 20

D Architecture Specifications **21**

E Experimental Setup **23**

E.1 Datasets 23

E.2 Feature Extraction and Graph Construction 24

E.3 Evaluation Protocol 24

E.4 Computational Infrastructure 25

F Baseline Implementation Details **25**

F.1 Multiple Instance Learning 25

F.2 Dense Graph Neural Networks 25

F.3 Sparse Graph Neural Networks 26

F.4 Reproducibility Checklist 26

G Extended Results **26**

G.1 Multi-Dimensional Performance 26

G.2 Learned Topology vs. Dense Baseline 27

H Architecture Design Ablations **28**

H.1 Edge Scoring Architecture 28

H.2 Choice of ℓ_0 Estimator 28

H.3 Temperature Annealing 29

H.4 GNN Backbone and **LENS+** Architecture 29

H.5 Pooling Mechanisms 29

H.6 Size-Aware Regularization 29

H.7 Lambda Scheduling 30
H.8 Target Density Sensitivity 33

I Input Representation Ablation **34**
I.1 Feature Encoder 35
I.2 Candidate Graph Construction (k-NN) 35
I.3 Spatial Structure: Topology vs. Features 37

J Interpretability and Quantitative Evaluation **40**
J.1 Comparison with Existing Interpretability Evaluations 41

K Failure Mode Analysis **47**
K.1 Molecular subtype not resolvable from H&E (BRCA HER2-enriched) 48
K.2 Local evidence overridden by global architecture (RCC cystic change) 48
K.3 Intrinsic morphologic ambiguity (poorly differentiated NSCLC) 48

L Algorithms **48**

M Limitations and Future Work **52**

N Extension to Survival Prediction **52**

A Additional Related Work on ℓ_0-Sparsification and Graph Models

Many recent works have revisited ℓ_0-based sparsification from both optimization and modeling perspectives. Louizos et al. introduced stochastic hard-concrete gates as a differentiable surrogate for ℓ_0 allowing neural networks to learn which weights to keep or prune during training while explicitly regularizing the expected number of active parameters (Louizos et al., 2018). In more classical settings, Hazimeh et al. developed **`L0Learn`**, a scalable toolkit for ℓ_0-regularized regression and classification that combines coordinate descent with local combinatorial search, demonstrating that exact or near-exact ℓ_0 solutions are tractable for large-scale linear models (Hazimeh et al., 2023). Complementing these algorithmic advances in sparse linear models, Soubies et al. developed **`CEL0`**, a continuous exact penalty for least-squares problems that recovers the ℓ_2-ℓ_0 objective while remaining amenable to gradient-based optimization, and provided a unified view of exact continuous penalties (Soubies et al., 2015; 2017). These contributions, together with related analyses in compressed sensing and statistical physics, provide much of the theoretical and algorithmic foundation for using ℓ_0-style penalties in modern deep models (Barbier et al., 2023).

Building on these foundations, recent work has focused on controlling sparsity in deep networks in a more global and principled way. **`ProbMask`**, introduced as "Effective Sparsification" (Zhou et al., 2021), enforces a single global sparsity level across layers via probabilistic masking, thereby avoiding ad hoc layer-wise pruning rates and showing strong compression–accuracy trade-offs on large vision models. Gallego-Posada et al. revisit Louizos-style stochastic gates in the "Controlled Sparsity" framework and formulate sparsification as a constrained optimization problem with an explicit target density controlled via dual variables. The resulting sparse solutions achieve densities that closely match the desired sparsity level on both convolutional networks and transformers (Gallego-Posada et al., 2022). A different mechanism for controlling sparsity relies on dynamic sparse training (**`DST`**) methods such as Sparse Evolutionary Training and **`RigL`**, which maintain a fixed parameter budget while continuously updating the sparse connectivity pattern during training, and have been shown to

match or surpass dense baselines on image classification and language modeling tasks (Evci et al., 2020; 2022; Mocanu et al., 2018). More recent studies examine when and why such sparse-to-sparse training schemes can outperform dense models, for example in diagonally structured architectures and in robustness evaluations with noisy or perturbed inputs, and show that the evolution of the sparse connectivity during training is often more important than the final static mask (Tyagi et al., 2025; Wu et al., 2025).

In addition to parameter sparsity, we emphasize sparsity that directly encodes structure, in particular, task-driven edge sparsification at the adjacency level. Sparsification of graphs, feature-interaction networks, and causal DAGs has been explored from several complementary angles. Sun & Yang formulate graph construction as an ℓ_0-regularized sparse coding problem with local geometric regularization, and derive a fast proximal-gradient algorithm whose hard-thresholding step corresponds to an ℓ_0 penalty on the entries of the learned adjacency (Sun & Yang, 2023). **`PTDNet`** learns to prune task-irrelevant edges in graph neural networks by combining a sparsity-inducing penalty on edge gates with a nuclear-norm regularizer that encourages low-rank structure in the sparsified adjacency, leading to improved robustness on noisy graphs and interpretable connectivity patterns (Luo et al., 2021). **`NeuralSparse`** removes task-irrelevant edges via reparameterized edge sampling with downstream feedback; although implemented with Gumbel sampling, it is compatible with HC/Bernoulli gates and an explicit edge-budget term. More recently, **`CoNNect`** introduces connectivity-based regularizers designed to preserve global information flow and avoid fragmented subnetworks, effectively implementing an ℓ_0-like prior over paths and structural motifs and showing benefits for one-shot and structured pruning of large neural networks (Franssen et al., 2025). Related ideas also appear in causal discovery and feature-interaction modeling: Duong et al. learn sparse causal graphs via a reinforcement-learning objective that favors compact DAGs without imposing an ℓ_0 penalty (Duong et al., 2025), whereas Su et al. use an ℓ_0 activation regularizer on edges to select a small set of beneficial feature interactions in recommender systems (Su et al., 2021).

Within computational pathology, sparsification usually appears as region-of-interest (ROI) selection or graph pruning/learning of sparse adjacency on whole-slide images. **`CLAM`** uses attention-based multiple-instance learning to identify a small subset of highly diagnostic patches, effectively inducing a sparse bag-level representation of each slide (Lu et al., 2021). Transformer-based MIL models such as **`TransMIL`** rely on correlated self-attention to highlight a limited number of informative instances while modeling long-range spatial dependencies across the slide (Shao et al., 2021). Graph-based WSI methods make sparsity explicit at the graph level: **`GraphLSurv`** learns adaptive and sparse WSI graphs for survival prediction and empirically shows that such adaptive sparse structures are more suitable than static or dense graphs for modeling WSIs (Liu et al., 2023); other pipelines introduce graph sparsification or edge-thresholding steps before graph embedding to discard weak or noisy connections, for example, **`SparseXMIL`** and related graph-based WSI classifiers (Le Bescond et al., 2026). Recent work on attention de-sparsification in self-supervised WSI representation learning provides a complementary perspective on sparsity in this domain, showing that overly sparse attention can miss important tissue context and proposing mechanisms to diversify attention over the slide (Kapse et al., 2024), thereby underscoring that sparsity in WSIs needs to be controlled rather than maximized.

In this context, our approach treats the WSI patch graph as the object to be sparsified under explicit control of density and structure. In contrast to methods that apply ℓ_0-style penalties to weight matrices or treat graph sparsification as post-hoc denoising (Evci et al., 2020; Franssen et al., 2025; Luo et al., 2021), we integrate ℓ_0-based gating directly into the adjacency, tie it to target-density control in the spirit of controlled sparsity (Gallego-Posada et al., 2022) and **`ProbMask`** (Zhou et al., 2021), and couple it with task-driven graph aggregation for WSI classification. This design controls which tissue-patch relations remain connected, rather than only how many parameters are active. The resulting frozen sparse graph is reusable at inference and aligned with the slide-level task. Thus, the relevant distinction is not sparsity alone, but where the sparsity is imposed: on a shared parameter matrix, on selected patches, or on the slide-specific tissue-interface graph used for computation.

B ℓ_0-Relaxation and Hard–Concrete Gates

This appendix provides additional details on the ℓ_0-based edge sparsification used in Section 3.1 of the main text. We first formulate binary edge selection as an $\ell_0$regularized mask-learning problem

and introduce a variational relaxation over Bernoulli masks. We then instantiate this relaxation with reparameterized Hard–Concrete edge gates, which yield low-variance pathwise gradients and a closed-form expected-ℓ_0 penalty. Next, we assemble these components into an end-to-end expected-ℓ_0 objective and clarify the distinction between training with stochastic gates and inference with a frozen deterministic sparse graph. This distinction is important because the optimization uses continuous relaxations only during training, whereas the reported explanations and compute measurements use the frozen deterministic inference graph. Finally, we describe the resulting pruning dynamics and practical sparsity schedules that control how the graph becomes sparse over the course of training.

B.1 ℓ_0-Regularized mask learning and variational relaxation

To solve the task with as few edges as possible, we optimize an ℓ_0-regularized empirical risk over binary edge masks, i.e.,

$$\min_{\theta,\, \boldsymbol{S} \in \{0,1\}^{|\mathcal{E}|}} \mathcal{L}_{\text{task}}(\theta;\, \boldsymbol{S}) \;+\; \lambda \|\boldsymbol{S}\|_0, \tag{B.1}$$

which directly promotes a truly sparse, interpretable subgraph by penalizing the number of active edges. This aligns the optimization with the inference goal of a deterministic sparse topology.

The difficulty is that this objective Eq. (B.1) is discrete and non-differentiable. In fact, this formulation creates two obstacles to standard gradient-based optimization: 1) the adjacency mask $\boldsymbol{S}$ is binary, so gradients with respect to $\boldsymbol{S}$ do not exist, and 2) the ℓ_0-penalty induces an NP-hard combinatorial search (Natarajan, 1995). We cast discrete mask learning as stochastic variational optimization (Bird et al., 2018): for any objective $\mathcal{F}(z)$ and any distribution $q(z)$,

$$\min_z \mathcal{F}(z) \le \mathbb{E}_{z\sim q(z)}[\mathcal{F}(z)]. \tag{B.2}$$

We therefore optimize the upper bound $\mathbb{E}_{z\sim q(z)}[\mathcal{F}(z)]$ with a parametric q_π (e.g., Concrete/Hard–Concrete), enabling gradient-based learning of π while approximating the binary problem. For edge gating, each binary S_{ij} is modeled with a factorized Bernoulli family

$$q_\pi(\boldsymbol{S}) \;=\; \prod_{(i,j)\,\in\,\mathcal{E}} \texttt{Bernoulli}\big(S_{ij} \mid \pi_{ij}\big), \tag{B.3}$$

which turns the discrete problem into the smooth surrogate

$$\min_{\theta,\,\pi} \; \mathbb{E}_{S\sim q_\pi}\big[\mathcal{L}_{\text{task}}(\theta;\boldsymbol{S})\big] \;+\; \lambda \sum_{(i,j)\,\in\,\mathcal{E}} \pi_{ij}, \tag{B.4}$$

where π_{ij} acts as an activation probability for edge (i,j), yielding a differentiable objective that preserves ℓ_0-style sparsity control.

In effect, we upper-bound the original ℓ_0-regularized objective (B.1) by its variational expectation (B.4), converting the discrete mask selection into a differentiable optimization over π. The sparsity term becomes directly differentiable (see Louizos et al., 2018), whereas the expected task loss remains challenging due to averaging over a high-dimensional binary mask, i.e., integration over a combinatorial discrete support, yielding high-variance gradients and costly Monte Carlo estimates (Bengio et al., 2013; Williams, 1992).

We mitigate this with low-variance pathwise gradients via Concrete/Hard-Concrete reparameterization (Louizos et al., 2018; Maddison et al., 2017). Alternative Bernoulli-gate estimators, including Augment-REINFORCE-Merge (ARM) and straight-through estimator (STE) variants (Bengio et al., 2013; Tokui & Oono, 2017; Yin et al., 2019), are evaluated in Appendix H.2; they underperform Hard-Concrete in accuracy or compute and are not used as the default. This yields tractable updates of π during training, after which we freeze a deterministic mask for inference.

B.2 Reparameterized edge-gates with Hard–Concrete and expected ℓ_0

This subsection instantiates the relaxation using reparameterized Hard–Concrete edge gates and derives the corresponding expected ℓ_0 penalty.

We replace discrete edge masks with differentiable Concrete/Hard–Concrete gates: draw $u \sim \mathcal{U}(0,1)$, set $g = \log u - \log(1-u)$ (logistic noise) and sample a Concrete gate

$$s = \sigma\big((\log\alpha + g)/\tau\big) \;\in\; (0,1). \tag{B.5}$$

To permit exact zeros/ones we apply a stretch and a clamp,

$$\bar{s} = \gamma + (\zeta - \gamma)\, s, \qquad z = \texttt{clip}_{[0,1]}(\bar{s}). \tag{B.6}$$

We use $\boldsymbol{Z} = \{z_{ij}\}$ in place of $\boldsymbol{S}$ inside the normalized operator $\hat{\boldsymbol{A}}(\cdot)$. We anneal the temperature τ (cosine with warmup) to sharpen gates over training. Under Hard-Concrete the keep-probability has a closed form,

$$\pi_{ij} = \mathbb{P}(z_{ij} > 0) = \sigma\Big(\log \alpha_{ij} - \tau \log \frac{-\gamma}{\zeta}\Big), \tag{B.7}$$

so the expected-ℓ_0 penalty becomes $\sum_{(i,j)} \pi_{ij}$, yielding the differentiable sparsity term in the objective

$$\mathbb{E}\big[\|\boldsymbol{S}\|_0\big] = \sum_{(i,j)\,\in\,\mathcal{E}} \sigma\Big(\log \alpha_{ij} - \tau \log \frac{-\gamma}{\zeta}\Big). \tag{B.8}$$

This penalizes the expected number of active edges directly as a smooth function of the gate logits and is used in the end-to-end objective in Appendix B.3 End-to-end expected-ℓ_0 objective and training vs inference.

Alternative estimator (STE). While our main approach uses Hard-Concrete reparameterization and optimizes the expected ℓ_0 objective with continuous gates $z_{ij} \in [0, 1]$, one can alternatively enforce hard discretization in the forward pass and backpropagate with a straight-through estimator (STE), replacing the clamp's zero derivative with a simple surrogate (e.g., identity or a **`hardtanh`** band). We do not rely on STE in our default training; we include it only as an estimator ablation in Appendix H.2.

Proposition B.1. *For any fixed temperature $\tau > 0$, the Gumbel–Sigmoid relaxation is fully differentiable with respect to α_{ij}, enabling gradient-based optimization (e.g., see Bengio et al., 2013).*

B.3 End-to-end expected-ℓ_0 objective and training vs inference

This subsection expands the brief training vs. inference summary in Section 3.1. We first describe how stochastic masks are used during training, and then how they are frozen at inference time.

In this work, $\mathcal{L}_{\text{task}}$ is the slide-level cross-entropy classification loss, optimized under the expected-ℓ_0 sparsification program

$$\min_{\theta,\,\varphi} \mathbb{E}_{\boldsymbol{Z}}\big[\mathcal{L}_{\text{task}}\big(\, f_\theta(\boldsymbol{X}, \boldsymbol{Z} \odot \boldsymbol{A}), \boldsymbol{y}\,\big)\big] + \lambda \sum_{(i,j)} \pi_{ij}, \tag{B.9}$$

where the edge-scorer $\boldsymbol{g}_\varphi$ outputs gate logits α that parameterize Hard-Concrete gates $z_{ij} \in [0, 1]$, and $\boldsymbol{Z}$ is sampled via the (Hard-)Concrete reparameterization for pathwise gradients. The regularizer uses the standard expected-ℓ_0 surrogate, with π_{ij} denoting "keep" probability (equivalently, the probability of a nonzero gate under the Hard-Concrete parameterization), so the penalty $\sum_{(i,j)} \pi_{ij}$ directly controls the expected number of retained edges. In practice, λ sets the sparsity–accuracy trade-off, while the temperature schedule τ gradually sharpens the gate distribution during training. At inference, sampling is disabled and each edge gate is computed deterministically from the learned logit using the noiseless Hard–Concrete transformation,

$$z_{ij}^{\text{inf}} = \text{clip}_{[0,1]}\left(\gamma + (\zeta - \gamma)\sigma\left(\frac{\log \alpha_{ij}}{\tau}\right)\right) \;\in\; [0, 1]. \tag{B.10}$$

The frozen deterministic gate matrix $\boldsymbol{Z}^{\text{inf}} = \{z_{ij}^{\text{inf}}\}_{(i,j)\,\in\,\mathcal{E}}$ defines the frozen sparse adjacency

$$\tilde{\boldsymbol{A}}^{\text{inf}} = \boldsymbol{A} \odot \boldsymbol{Z}^{\text{inf}}, \tag{B.11}$$

which is used for all downstream computation.

C Edge-Scoring Network g_φ

This appendix details the architecture of the edge-scoring network g_φ used in Section 3.2 to produce edge logits for ℓ_0-based sparsification. Appendix B defines the Hard-Concrete gate; here we focus

on the deterministic mapping from patch-pair features to $\log \alpha_{ij}$. This separation makes the scorer inductive: the learned g_φ can assign edge logits to candidate tissue relations in unseen slides without storing slide-specific edge parameters. We describe how node pairs are preprocessed, how feature-importance attention and relation-aware descriptors are constructed, and how the two-tower MLP maps them to gate logits.

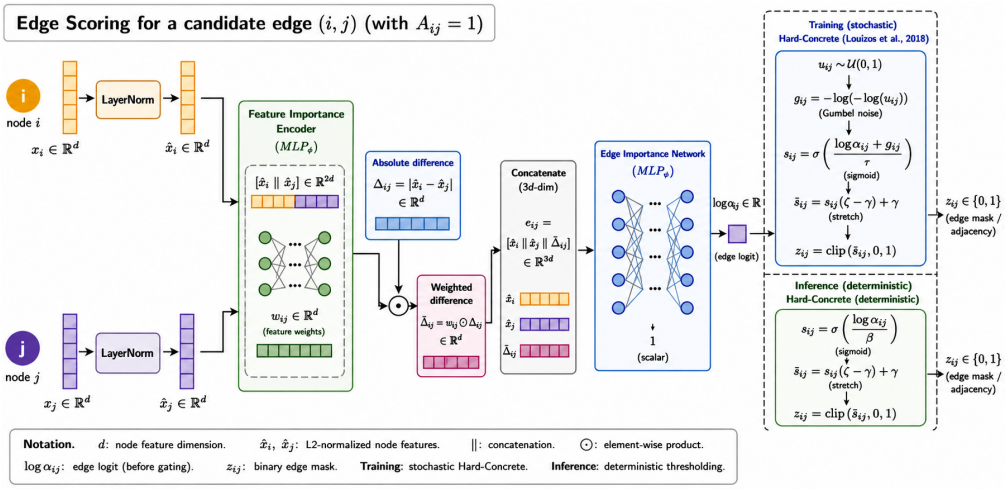

Figure C.1: **Relation-aware two-tower edge scoring.** For each candidate edge (i, j), normalized endpoint features are passed to an attention tower that produces channel-wise weights $\boldsymbol{a}_{[i,j]}$. The attended contrast $|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}$ is concatenated with the endpoints and passed to a scoring tower that outputs the gate logit $\log \alpha_{ij}$. Hard-Concrete gating converts these logits into stochastic edge gates during training and a frozen deterministic sparse adjacency at inference using the noiseless gate transformation. $(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_j) \;\to\; \boldsymbol{a}_{[i,j]} \;\to\; \boldsymbol{e}_{[i,j]} \;\to\; \log \alpha_{ij} \;\to\; z_{ij}$.

C.1 Preprocessing of pairwise inputs

This subsection specifies how node embeddings are normalized and combined into pairwise inputs that will be fed to the edge-scoring network.

We ℓ_2-normalize node embeddings to obtain $\hat{\boldsymbol{x}}_i$, form the ordered concatenation $[\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j]$ and compute the absolute difference $|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j|$. This preprocessing stage standardizes the magnitude of node embeddings and produces a symmetric contrast term, which will be modulated by learned feature-importance weights in the next step.

C.2 Feature-importance attention

This subsection describes how we compute a channel-wise feature-importance vector for each node pair, using a small MLP followed by a feature-wise softmax. We pass the ordered pair $[\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j] \in \mathbb{R}^{2d}$ through a two-layer MLP ($2d \to 128 \to d$, i.e., $3{,}072 \to 128 \to 1{,}536$ for **`UNI2-h`**) with a ReLU hidden layer and a feature-wise softmax to obtain a per-edge, per-dimension importance profile, i.e., let $\texttt{MLP}_\phi : \mathbb{R}^{2d} \to \mathbb{R}^d$ be

$$\texttt{MLP}_\phi\big([\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j]\big) = \boldsymbol{W}_2^T \,\texttt{ReLU}\big(\boldsymbol{W}_1^T [\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j] + \boldsymbol{b}_1\big) + \boldsymbol{b}_2,$$

and define the attention weights by a softmax across the d feature dimensions, i.e.,

$$\boldsymbol{a}_{[i,j]} = \texttt{softmax}\big(\texttt{MLP}_\phi\big([\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j]\big)\big) \;\in\; \Delta^{d-1}, \tag{C.1}$$

which ensures a normalized importance profile $\boldsymbol{a}_{[i,j]}$. Thus, $\boldsymbol{a}_{[i,j]}$ acts as a channel-wise attention vector that highlights edge-specific informative dimensions while down-weighting less relevant ones.

C.3 Relation-aware contrast and edge-feature assembly

This subsection defines a relation-aware contrast term between node pairs and shows how we assemble it with the endpoints into a rich edge feature used for scoring.

Motivated by prior work showing that explicit *pairwise structure* helps preserve class separability and mitigates representation collapse (see Koishekenov, 2023), we equip each edge with a contrastive descriptor by concatenating normalized endpoints with an attended difference, i.e.,

$$\boldsymbol{e}_{[i,j]} = \big[\, \hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j \,||\, \mathrm{relation}(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_j) \,\big] \;\in\; \mathbb{R}^{3d}. \tag{C.2}$$

We instantiate $\mathrm{relation}(\cdot,\cdot)$ as an *attended discrepancy* that highlights task-relevant channels while attenuating non-predictive channels, improving discriminability and stability. We compared (i) the plain absolute difference $|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j|$ with (ii) its channel-wise reweighting by $\boldsymbol{a}_{[i,j]}$ (Hadamard product). The reweighted version improves accuracy from 87.5% to 88.4% at the same retention (Table H.1), so we adopt

$$\mathrm{relation}(\hat{\boldsymbol{x}}_i, \hat{\boldsymbol{x}}_j) = |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}. \tag{C.3}$$

This edge feature encodes channel-selective differences between the two patches, so the scorer can weight the dissimilarities that matter for the task.

In summary, $\boldsymbol{e}_{[i,j]}$ aggregates the two endpoints and an attended discrepancy term that is both contrastive and channel-selective, making it a rich descriptor for edge importance.

C.4 Edge-importance head and two-tower formulation

This subsection specifies the MLP that maps edge features to scalar logits and explains how the attention and scoring branches form a two-tower edge-scoring network. We use a three-layer MLP that maps the edge feature $\big[\, \hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j \,||\, |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]} \,\big]$ to a scalar gate logit $\log \alpha_{ij}$ measuring importance via two affine blocks, each followed by ReLU and dropout (0.2), and a final linear readout, with spectral normalization on all layers, i.e.,

$$\log \alpha_{ij} = \mathbf{MLP}_{\psi}\big(\, \big[\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j \,||\, |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}\big] \,\big) \;\in\; \mathbb{R}, \tag{C.4}$$

where $\mathbf{MLP}_{\psi}\colon \mathbb{R}^{3d} \to \mathbb{R}$, with widths $3d \to 128 \to 128 \to 1$ ($4{,}608 \to 128 \to 128 \to 1$ for **UNI2-h**). The final-layer bias is initialized to 1.0, so almost all edges are kept at the start of training.

Definition C.1 (Two-tower edge scorer)**.** The edge-scorer $\boldsymbol{g}_{\varphi}$ is a two-tower module with parameters $\varphi = (\phi, \psi)$. Tower $\mathbf{MLP}_{\phi}$ ("attention") produces channel weights $\boldsymbol{a}_{[i,j]}$, and tower $\mathbf{MLP}_{\psi}$ ("scoring") outputs the gate logit $\log \alpha_{ij}$. The towers are applied in sequence: the output of $\mathbf{MLP}_{\phi}$ enters the input of $\mathbf{MLP}_{\psi}$, i.e.,

$$\begin{cases} \boldsymbol{a}_{[i,j]} = \mathbf{softmax}\big(\mathbf{MLP}_{\phi}\big([\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j]\big)\big) \;\in\; \Delta^{d-1}, \\ \log \alpha_{ij} = \mathbf{MLP}_{\psi}\big(\, \big[\hat{\boldsymbol{x}}_i \,||\, \hat{\boldsymbol{x}}_j \,||\, |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}\big] \,\big) \;\in\; \mathbb{R}. \end{cases}$$

Hence,

$$\log \alpha_{ij} = \boldsymbol{g}_{\varphi}(\hat{\boldsymbol{x}}_i,\, \hat{\boldsymbol{x}}_j). \tag{C.5}$$

This two-tower formulation separates "what to look at" (channel importance via $\mathbf{MLP}_{\phi}$) from "how important the relation is" (edge logit via $\mathbf{MLP}_{\psi}$), while allowing joint training under the same sparsification objective.

C.5 Coupling with Hard-Concrete gating and graph aggregation

This subsection explains how the edge logits from g_{φ} are converted into Hard-Concrete gates and then into the sparse operator used by the GNN.

Differentiable stochastic gating. For train-time sparsification, we apply the Hard-Concrete relaxation from Appendix B to $\log \alpha_{ij}$, yielding stochastic gates $z_{ij} \in [0,1]$, exact-zero outcomes after stretch-and-clamp, and closed-form keep probabilities π_{ij}. Because the scorer input is ordered, both orientations of each undirected candidate edge are scored, and z_{ij} and z_{ji} are averaged, in training and at inference.

Learning objective. We train end-to-end with slide-level supervision on the gated graph and the expected-ℓ_0 objective. At inference, sampling is disabled and each gate is computed deterministically from the learned logit using the noiseless Hard-Concrete transformation at the final temperature $\tau{=}\tau_{\min}{=}0.67$, $z_{ij}^{\inf} = \mathrm{clip}_{[0,1]}\big(\gamma + (\zeta - \gamma)\sigma\big(\log \alpha_{ij}/\tau\big)\big)$. After symmetrization, this yields a frozen deterministic gate matrix $\boldsymbol{Z}^{\inf} \in [0,1]^{n\times n}$, nonzero only on candidate edges, which defines the

frozen sparse adjacency $\tilde{\boldsymbol{A}}^{\text{inf}} = \boldsymbol{A} \odot \boldsymbol{Z}^{\text{inf}}$. The same frozen adjacency is used for message passing and RoI-aligned visualization.

Graph aggregation and message passing ($\texttt{GNN}_\theta$). During training, the gated adjacency used by the GNN is

$$\hat{\boldsymbol{A}} = \texttt{row_norm}(\boldsymbol{A} \odot \boldsymbol{Z} + \boldsymbol{I}) \tag{C.6}$$

where $\boldsymbol{Z} \in [0,1]^{n\times n}$ contains the symmetrized stochastic Hard-Concrete gates on candidate edges. This produces stable, degree-normalized message mixing, where each node aggregates a normalized mixture of its neighbors. We then perform neighborhood aggregation and update node embeddings via a standard graph layer.

At inference, $\boldsymbol{Z}$ is replaced by the frozen deterministic gate matrix $\boldsymbol{Z}^{\text{inf}}$, giving

$$\hat{\boldsymbol{A}}^{\text{inf}} = \texttt{row_norm}(\boldsymbol{A} \odot \boldsymbol{Z}^{\text{inf}} + \boldsymbol{I}). \tag{C.7}$$

Thus, ($\texttt{GNN}_\theta$) performs message passing only along retained tissue-patch edges, while self-loops preserve each patch's own representation. Together, Appendices B and C specify the pipeline from patch features to gate logits $\log \alpha_{ij}$, Hard-Concrete gates, and the frozen sparse operator used by the GNN layers at inference.

D Architecture Specifications

Both **LENS** and **LENS+** share the same edge scorer, Hard-Concrete gating module, and topology-learning mechanism; they differ only in graph-processing depth and graph-level pooling. Differences between the two variants therefore reflect graph-processing capacity, not a change in how edges are selected. The main architecture figure is in Section 3 (Figure 1); the edge scorer is detailed in Appendix C. Two tables below list the shared settings and the variant-specific components.

Shared edge scorer. The scorer has two towers. The attention tower $\texttt{MLP}_\phi$ maps the concatenated endpoint features $[\hat{\boldsymbol{x}}_i \,\|\, \hat{\boldsymbol{x}}_j] \in \mathbb{R}^{3,072}$ to a channel-importance vector $\boldsymbol{a}_{[i,j]} \in \Delta^{1,535}$ through a two-layer MLP (3,072 $\to$ 128 $\to$ 1,536), followed by a softmax. The scoring tower then forms the relation descriptor $\boldsymbol{e}_{[i,j]} = [\hat{\boldsymbol{x}}_i \,\|\, \hat{\boldsymbol{x}}_j \,\|\, |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}] \in \mathbb{R}^{4,608}$ (three concatenated 1,536-dim vectors with **UNI2-h** features). A three-layer MLP $\texttt{MLP}_\psi$ (4,608 $\to$ 128 $\to$ 128 $\to$ 1, ReLU, dropout 0.2, spectral normalization on all layers) maps this descriptor to $\log \alpha_{ij}$. The final-layer bias is initialized to 1.0. Initialization: Kaiming normal (ReLU layers), Xavier uniform with gain 0.1 (output). The scorer is applied independently to each candidate edge, so the same g_φ generalizes inductively across slides with different graph sizes. The scorer has 1.20M parameters (attention tower 0.59M, scoring tower 0.61M) and is shared by both variants. The first layer of each tower is split into per-node terms, $\boldsymbol{W}[\hat{\boldsymbol{x}}_i \,\|\, \hat{\boldsymbol{x}}_j] = \boldsymbol{W}_1\hat{\boldsymbol{x}}_i + \boldsymbol{W}_2\hat{\boldsymbol{x}}_j$, which are computed once per patch and reused across its candidate edges. In the scoring tower, only the contrast block $\boldsymbol{W}_3\big(|\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}\big)$ is computed per edge.

Shared ℓ_0 gating. Hard-Concrete gates use stretch bounds $[\gamma, \zeta]=[-0.1, 1.1]$ and $\beta=0.66$. The temperature τ stays at 5.0 during the 15-epoch warm-up and is then annealed to 0.67 with a cosine schedule over 20 epochs (epochs 15 to 35), with target retention $\rho_e=0.30$. During training, stochastic gates $z_{ij} \in [0,1]$ are sampled. Both orientations of each candidate edge are scored, and z_{ij} and z_{ji} are averaged, so every undirected edge has a single gate in training and at inference. At inference, sampling is disabled and the gates are computed deterministically (see *Inference checks* below). This gives a frozen gate matrix $\boldsymbol{Z}^{\text{inf}} \in [0,1]^{n\times n}$, nonzero only on candidate edges, and the frozen sparse adjacency $\tilde{\boldsymbol{A}}^{\text{inf}} = \boldsymbol{A} \odot \boldsymbol{Z}^{\text{inf}}$.

LENS (efficiency). Features are aggregated once on $\hat{\boldsymbol{A}}=\text{row_norm}(\tilde{\boldsymbol{A}}+\boldsymbol{I})$ and passed through a single linear layer (1,536 $\to$ 128). Graph-level pooling uses softmax attention over the retained edge mass $w_i = \sum_j z_{ij}$, so patches with stronger retained connectivity receive larger weight. This pooling adds no learned parameters. Classifier: MLP 128 $\to$ 64 $\to$ C (ReLU, layer norm, spectral norm). 0.21M backbone parameters (1.40M including the shared scorer), 1-hop receptive field.

LENS+ (accuracy). Linear projection 1,536 $\to$ 256 with layer norm, ReLU, dropout (0.2). Three residual GCN layers (256 $\to$ 256), each with layer norm, spectral norm, dropout (0.2). Multi-head attention pooling with a learnable [CLS] token (8 heads, head dimension 32, dropout 0.2, $\mathbb{R}^{256\times256}$ projections). [CLS]-to-patch attention is restricted to active nodes, $m_i=\mathbb{I}[w_i>0]$. Thus,

Table D.1: **Shared architecture and training hyperparameters for LENS and LENS+.** Both variants use the same edge scorer, Hard-Concrete gating, and training setup; variant-specific components are listed in Table D.2.

Component	Hyperparameter	Value (both variants)
Node embedding	Patch embedding dim	1,536 (UNI2-h)
Candidate graph	Construction	8-NN, undirected
Attention tower ($\texttt{MLP}_\phi$)	Input dimension	$2 \times 1{,}536 = 3{,}072$
	Layers	$3{,}072 \to 128 \to 1{,}536$, softmax
Scoring tower ($\texttt{MLP}_\psi$)	Input dimension	$3 \times 1{,}536 = 4{,}608$
	Layers	$4{,}608 \to 128 \to 128 \to 1$
	Activation	ReLU
	Dropout	0.2
	Normalization	Spectral (all layers)
	Output bias init	1.0
Scorer total	Parameters	1.20M
ℓ_0 gating	Distribution	Hard-Concrete
	Stretch bounds $[\gamma, \zeta]$	$[-0.1,\ 1.1]$
	β	0.66
	Temperature schedule	$5.0 \to 0.67$ (20 epochs)
	Inference	Deterministic, τ=0.67
Training	Optimizer	AdamW, lr 10^{-3}, wd 5×10^{-4}
	Batch size	1 (slide)
	Epochs	80, best-validation checkpoint
	λ_0	0.01
	λ_ρ	0.03
	Target ρ_e	0.30
	Adaptive range $(\alpha_{\min}, \alpha_{\max})$	$(0.2,\ 2.0)$
	Sparsity warm-up / ramp	15 / 20 epochs

patches whose gates are all exactly zero cannot contribute to the slide representation. Classifier: MLP $256 \to 128 \to 64 \to C$ (ReLU, layer norm, spectral norm, dropout 0.1). 0.90M backbone parameters (2.10M including the shared scorer), 3-hop receptive field.

Training. Both variants are trained end-to-end with slide-level cross-entropy, expected-ℓ_0 sparsification, and density matching. We use AdamW with learning rate lr=1×10^{-3}, weight decay 5×10^{-4}, batch size 1, and 80 epochs, without early stopping. The checkpoint with the highest validation accuracy is evaluated once on the held-out test fold. We set λ_0=0.01 and λ_ρ=0.03, chosen on the validation set. The sparsity terms are switched off for a 15-epoch warm-up and then ramped in linearly over 20 epochs. The adaptive sparsity weight (Eq. (3.8)) uses $(\alpha_{\min}, \alpha_{\max})$=$(0.2, 2.0)$.

Inference checks. At inference, gates are frozen deterministically from the learned logits using the noiseless Hard-Concrete transformation with the final temperature τ=0.67:

$$z_{ij}^{\text{inf}} = \text{clip}_{[0,1]}\left(\gamma + (\zeta - \gamma)\sigma\left(\frac{\log \alpha_{ij}}{\tau}\right)\right).$$

All graphs are undirected. Because the scorer input is ordered, both orientations of a candidate edge are scored, and the paired values z_{ij}^{inf} and z_{ji}^{inf} are averaged, as in training, so each undirected edge has a single weight. Retained edges keep this continuous value as edge weight, and edges with weight 0 are removed. Edge retention is the fraction of undirected candidate edges with nonzero weight. Self-loops are then added, and the adjacency is row-normalized for message passing, as in training.

Table D.2: **Variant-specific architecture of `LENS` and `LENS+`.** Shared edge scoring, Hard-Concrete gating, and training settings are listed in Table D.1; this table lists the differences in graph depth, pooling, and classifier size.

Component	Hyperparameter	LENS	LENS+
GNN input	Projection	None	$1{,}536 \to 256$
	Normalization	None	Layer norm
	Dropout	None	0.2
GNN layers	Number of layers	1	3
	Dimensions	$1{,}536 \to 128$	$256 \to 256$
	Residual connections	None	Per layer
	Layer normalization	None	Per layer
	Spectral normalization	None	Per layer
	Dropout	None	0.2
	Receptive field	1-hop	3-hop
Pooling	Method	Softmax over edge mass	MHA + [CLS]
	Learned parameters	None	Yes
	Number of heads	N/A	8
	Head dimension	N/A	64
	Dropout	N/A	0.2
Classifier	Architecture	2-layer MLP	3-layer MLP
	Dimensions	$128 \to 64 \to C$	$256 \to 128 \to 64 \to C$
	Dropout	None	0.1
	Normalization	LN + Spectral	LN + Spectral

E Experimental Setup

This appendix details the dataset construction, feature extraction, graph construction, evaluation protocol, and compute measurement used for all methods. All methods share the same features, graphs, splits, and evaluation code, so reported differences come from the models themselves.

Complexity. Let n be the number of patches, d=1,536 the feature dimension, h=128 the scorer hidden size, and $|\mathcal{E}|$ the number of candidate edges of the 8-NN graph. Edge scoring runs once per slide, at training and at inference. With the first layer of each tower split into per-node terms (Appendix D), its cost is $\mathcal{O}(n\,d\,h + |\mathcal{E}|\,d\,h)$. Message passing then runs only on the retained edges, $\sim$28% of the candidates, which reduces its cost by $\sim 3.6\times$ relative to the full candidate graph. All GFLOPs reported in the paper include the edge scorer.

E.1 Datasets

We evaluate on four public H&E-stained WSI benchmarks spanning 5,530 slides from 3,340 cases across lung, kidney, and breast cancers (Table E.1). All datasets provide slide-level labels only; no patch-level annotations are used during training.

CPTAC-Lung (TCIA, n=2,071 slides, 435 cases) is the primary benchmark, with three nearly balanced classes: Normal (719, 34.7%), LUAD (667, 32.2%), and LSCC (685, 33.1%).

TCGA-LUNG (GDC, n=1,548 slides, 1,026 cases) provides cross-dataset generalization with a balanced binary split: LUAD (808, 52.2%) vs. LSCC (740, 47.8%).

TCGA-RCC (GDC, n=878 slides, 846 cases) presents marked class imbalance: ccRCC (487, 55.5%), pRCC (283, 32.2%), chRCC (108, 12.3%). The 12.3% minority mirrors clinical prevalence.

TCGA-BRCA (GDC, n=1,033 slides, 1,033 cases) uses PAM50 molecular subtypes with severe imbalance: Luminal A/B merged (775, 75.0%), Basal-like (124, 12.0%), and HER2-enriched (134, 13.0%). Normal-like cases are excluded.

Table E.1: **Dataset summary.** All cohorts provide slide-level labels only. CV is the coefficient of variation of class counts (higher means more imbalanced).

Dataset	Classes	Slides	Cases	CV	Source
CPTAC-Lung	3	2,071	435	0.04	TCIA
TCGA-LUNG	2	1,548	1,026	0.06	GDC
TCGA-RCC	3	878	846	0.65	GDC
TCGA-BRCA	3	1,033	1,033	1.08	GDC
Total		**5,530**	**3,340**		

All splits are patient-level, so slides from the same patient never appear in both training and test partitions.

E.2 Feature Extraction and Graph Construction

Primary encoder `UNI2-h`. We use the UNI2-h pathology foundation model (Chen et al., 2024) (ViT-H/14, 1536-dim, pretrained on 200M+ H&E and IHC images from 350K+ slides). Patches are tiled at 256×256 pixels, $20\times$ magnification, following the current field standard. We use official pre-extracted UNI2-h embeddings released by the Mahmood Lab (`huggingface.co/datasets/MahmoodLab/UNI2-h-features`) for TCGA and CPTAC cohorts.

Secondary encoders. For the encoder ablation (Appendix I.1), we also use **`CTransPath`** (Wang et al., 2022) (768-dim) and the **`SimCLR`** ResNet-18 encoder from **`GTP`** (Zheng et al., 2022), (512-dim, 512×512 patches), which is pretrained on lung cancer histopathology. Features for these two encoders are extracted once and cached.

Graph construction. For each slide, patch spatial coordinates are used to build an 8-nearest-neighbor undirected graph. This 8-NN graph is only the candidate graph; **`LENS`** learns the retained inference graph by gating its edges. This produces a binary adjacency matrix $\boldsymbol{A} \in \{0,1\}^{n\times n}$ stored as a sparse COO tensor. For **`UNI2-h`**, we build the 8-NN graphs from the patch coordinates released with the features.

E.3 Evaluation Protocol

Three-way split. We use patient-stratified 5-fold cross-validation with an explicit three-way partition per fold: a training set for optimization, a validation set for checkpoint and hyperparameter selection, and a held-out test set for final evaluation. Each fold is repeated with three seeds (42, 123, 2024), giving 15 runs per method and dataset. All reported metrics (accuracy, AUROC) come exclusively from the test set. The validation-selected checkpoint is evaluated once on test data that is never used for training or selection. The same fold assignments are reused for every method and ablation.

Metrics. We report classification accuracy and macro-averaged one-vs-rest AUROC. Macro-averaging assigns equal weight to each class, mitigating majority-class dominance on imbalanced datasets (e.g., TCGA-RCC with a 12.3% minority class).

Statistical testing. For each dataset and metric, we compare **`LENS+`** with every other method using a two-sided paired t-test. Pairing is by run: for each fold and seed, both methods are trained and tested on exactly the same split, and we take the difference $\delta_r = m_r^{\text{LENS+}} - m_r^{\text{base}}$ for $r=1,\ldots,15$. The null hypothesis is that the mean difference is zero. The test statistic is $t = \bar{\delta}/(s_\delta/\sqrt{15})$, where $\bar{\delta}$ and s_δ are the mean and standard deviation of the 15 differences, with 14 degrees of freedom. Pairing removes the variation that comes from fold difficulty, which affects all methods in the same way.

Computational cost. Forward-pass FLOPs are measured with `fvcore`, averaged over 10 test slides per dataset chosen to span the node-count distribution, and reported in GFLOPs (10^9 operations). For sparse message passing, we count $2 \times \text{nnz}(\boldsymbol{A}_{\text{sparse}}) \times d$ inside the full model estimate. For **`LENS`** and **`LENS+`**, the count includes the edge scorer on all candidate edges.

Efficiency ratio. $\eta = \text{Accuracy}(\%)/\text{GFLOPs}$ gives accuracy per unit of computation.

E.4 Computational Infrastructure

All experiments run on NVIDIA H200 GPUs (Hopper architecture, 143.7 GB HBM3e, compute capability 9.0) with CUDA 12.8 (driver 580.95.05). Implementation: PyTorch 2.8.0, PyTorch Geometric 2.6.1, Python 3.10.

On CPTAC-Lung, training a single **LENS** model requires approximately 3–4 hours per fold, with 5-fold CV completing in $\sim$18 hours per seed and dataset. Feature extraction for **CTransPath** and **SimCLR** takes $\sim$2 hours per dataset and is cached.

Reproducibility. Fixed seeds (42, 123, 2024) for PyTorch, NumPy, and Python random. Patient-level stratified splits saved to disk and reused across all methods. Precomputed features shared across all baselines. Shared evaluation pipeline (scikit-learn for accuracy/AUROC; fvcore for FLOPs).

F Baseline Implementation Details

All methods use the same **UNI2-h** features (1,536-dim, 256×256 patches at 20×), the same patient-stratified folds and seeds, and the evaluation protocol of Appendix E. All graph-based methods start from the same undirected 8-NN candidate graph. For the encoder ablation (Appendix I.1), all methods are also run with **CTransPath** and **SimCLR** features. The only changes to each method are the input dimension (1,536) and the number of output classes.

Fairness of comparison. Each baseline keeps its original design and runs on the same features, splits, graphs, and evaluation code as **LENS**. Differences in the results therefore come from how each method aggregates, pools, thresholds, or gates information, not from preprocessing or evaluation choices.

Hyperparameter tuning. We start from the settings recommended in each official repository. Hyperparameters are then tuned with Bayesian optimization using Weights & Biases sweeps, maximizing validation accuracy on the validation split only. The test folds are never used during tuning.

For every method, the checkpoint with the best validation accuracy is evaluated on the test fold.

F.1 Multiple Instance Learning

ABMIL (Ilse et al., 2018). Gated attention MIL with a projection $1{,}536 \to 512$, attention dimension 128, and a classifier MLP $512 \to 256 \to C$. Adam (2×10^{-4}, wd 10^{-5}), batch 1, 20 epochs, early stopping (patience 10). Dropout 0.1.

TransMIL (Shao et al., 2021). L=2 Transformer blocks, d=512, k=8 heads (d_h=64). Nyström attention with m=64 landmarks. PPEG with grouped convolutions ($k \in \{3, 5, 7\}$). Classifier: MLP $512 \to 256 \to C$. Adam (2×10^{-4}, wd 10^{-5}), batch 1 with gradient accumulation over 4 steps, 20 epochs, early stopping (patience 10). Dropout 0.1. Patches are capped at 5,000 per slide when necessary. Built on: https://github.com/szc19990412/TransMIL.git.

CLAM-SB / CLAM-MB (Lu et al., 2021). Projection $1{,}536 \to 512$ and gated attention ($\boldsymbol{V}_a, \boldsymbol{U}_a \in \mathbb{R}^{256\times512}$). CLAM-SB uses a single attention branch; CLAM-MB uses one branch per class. Instance-level clustering loss (c_1=0.7, c_2=0.3, top/bottom B=8 patches, SVM margin α=1.0). Classifier: MLP $512 \to 256 \to C$. Adam (2×10^{-4}, wd 10^{-5}), batch 1, 200 epochs, early stopping (patience 20). Dropout 0.25, gradient clipping (max norm 1.0), learning-rate decay on plateau (factor 0.5, patience 5). Built on: https://github.com/mahmoodlab/CLAM.git.

F.2 Dense Graph Neural Networks

GCN. Input projection $1{,}536 \to 512$, then a 3-layer GCN ($512 \to 256 \to 128$) with symmetric normalization $\boldsymbol{H}^{(l+1)}=\mathrm{ReLU}(\tilde{\boldsymbol{D}}^{-1/2}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-1/2}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$. Pooling: concatenated global mean and max ($\mathbb{R}^{256}$). Classifier: MLP $256 \to 128 \to C$. Adam (10^{-3}, wd 5×10^{-4}), batch 8, 150 epochs, step learning rate (10^{-4} at epoch 30, 10^{-5} at epoch 100). Dropout 0.1, label smoothing 0.1. All candidate edges are used. Implemented with PyTorch Geometric `GCNConv`.

Patch-GCN (Chen et al., 2021). 4-layer GCN (512 channels) with residual and dense skip connections, and edge features from normalized spatial distances. Attention-based global pooling. Adapted

from survival prediction to classification by replacing the Cox hazard head with a softmax classifier over C classes. Adam (2×10^{-4}, wd 10^{-5}), batch 1 with accumulation over 32 steps, 20 epochs, early stopping (patience 10). Dropout 0.1, cosine annealing ($T_{\max}$=20). All candidate edges are used. Built on: https://github.com/mahmoodlab/Patch-GCN.git.

GTP (Zheng et al., 2022). A single GCN layer, followed by MinCut pooling to ∼100 super-nodes, 3 Transformer blocks (k=8 heads, d_h=64, MLP hidden 128), and a classifier $128 \to 64 \to C$. MinCut regularizer weight λ_{cut}=0.01. Pre-norm Transformer without positional encoding; spatial structure comes from the graph. Adam (10^{-3}, wd 10^{-4}), batch 8, 150 epochs, step learning rate (10^{-4} at epoch 30, 10^{-5} at epoch 100). Dropout 0.1. All candidate edges are used. Built on: https://github.com/vkola-lab/tmi2022.git.

F.3 Sparse Graph Neural Networks

GraphLSurv (Liu et al., 2023). Learnable projection $\boldsymbol{T} \in \mathbb{R}^{d\times128}$, cosine similarity in the projected space, and a threshold δ that masks low-similarity edges. Hybrid message passing (GCN-HMP) with mixing coefficient λ. Adapted from survival prediction to classification by replacing the survival head with a softmax classifier over C classes. Dataset-specific settings: CPTAC-Lung and TCGA-LUNG use pure GCN-HMP×1 with δ=0.8, λ=0.0, anchor ratio r=0.2, and smoothness α=0.3; TCGA-BRCA and TCGA-RCC use GCN-HMP×1 followed by GCN×2 with δ=0.9, λ=0.3, r=0.1, and α=0.1. Adam (10^{-4}, wd 10^{-5}), batch 16, 300 epochs, learning rate multiplied by 0.8. Gradient clipping (max norm 1.0). Following the original method, each slide is reduced to its top s=1,000 patches by Sobel-filter selection, so **GraphLSurv** sees fewer patches than the other methods. Edge retention is ∼45% on CPTAC-Lung. Built on: https://github.com/liupei101/GraphLSurv.git.

F.4 Reproducibility Checklist

- Seeds 42, 123, and 2024 for PyTorch, NumPy, and Python; 15 runs per method and dataset.
- Patient-level stratified splits saved to disk and reused by every method.
- Precomputed **UNI2-h**, **CTransPath**, and **SimCLR** features shared by every method.
- The validation-selected checkpoint is evaluated once on the held-out test fold.
- One evaluation pipeline for all methods: scikit-learn for accuracy and AUROC, fvcore for GFLOPs.
- Full hyperparameters for **LENS**/**LENS+** in Appendix D, and for baselines above.
- Code, configuration files, and trained models: https://anonymous.4open.science/r/LENS-014E/.

G Extended Results

This section provides two visualizations that complement the main results: a multi-objective profile across five axes, and the learned sparse topology on one slide compared with the dense graph. Both are diagnostic views, not additional benchmarks: Figure G.1 summarizes the trade-offs, and Figure G.2 shows the graph that task-supervised sparsification produces.

G.1 Multi-Dimensional Performance

Figure G.1 compares methods on CPTAC-Lung (**UNI2-h** features) across five axes: accuracy, AUROC, computational efficiency, edge sparsity, and interpretability. The plot is a compact view of the trade-offs, not a ranking metric. **LENS+** gives the best overall balance (91.2% accuracy, 96.7% AUROC, 3.78 efficiency ratio, 71% sparsity), and **LENS** gives the highest efficiency ratio (4.60) at 88.4% accuracy. Dense **GTP** reaches similar accuracy (90.5%) at a much lower efficiency ratio (0.59). MIL methods give attention-based patch scores but no topology-level explanation, and **GraphLSurv** is lower on both accuracy (84.2%) and efficiency (3.30). The two variants cover two operating points: **LENS** for limited compute and **LENS+** for accuracy. Note that MIL methods and dense GNNs do not sparsify the graph, so their sparsity score is zero by construction.

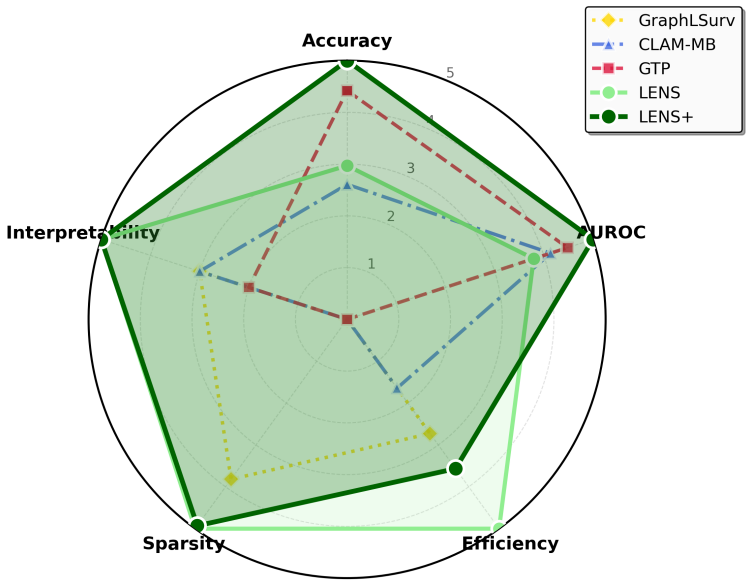

Figure G.1: **Five-axis performance profile on CPTAC-Lung (`UNI2-h`).** Accuracy, AUROC, efficiency ratio, edge sparsity, and interpretability for all methods. Each axis is rescaled to 0–5 for display only; raw values are given in Section G.1 and Table 1. MIL methods and dense GNNs have zero sparsity by construction.

G.2 Learned Topology vs. Dense Baseline

Figure G.2 shows what task-supervised sparsification produces on a representative LUAD slide. The dense graph used by **`GTP`** connects every patch to its 8 spatial neighbors, giving 11,652 edges, many of them between stroma or background patches. On this slide, **`LENS`** keeps 2,210 edges (∼19% of candidates, below the 28% average), removing most edges in background tissue and keeping dense connectivity inside the expert-annotated tumor region.

Two points stand out. First, the kept edges in the zoom-in form connected subgraphs, not isolated patches, so **`LENS`** does not reduce to independent patch scoring. Second, this structure is learned from the slide-level label alone; no region annotation or pathologist input is used in training. The main paper quantifies this: 78% of kept edges lie within expert-delineated tumor, 15% connect tumor and stroma, and 7% lie in non-diagnostic tissue, compared with a more even 33/27/40% split in the dense graph.

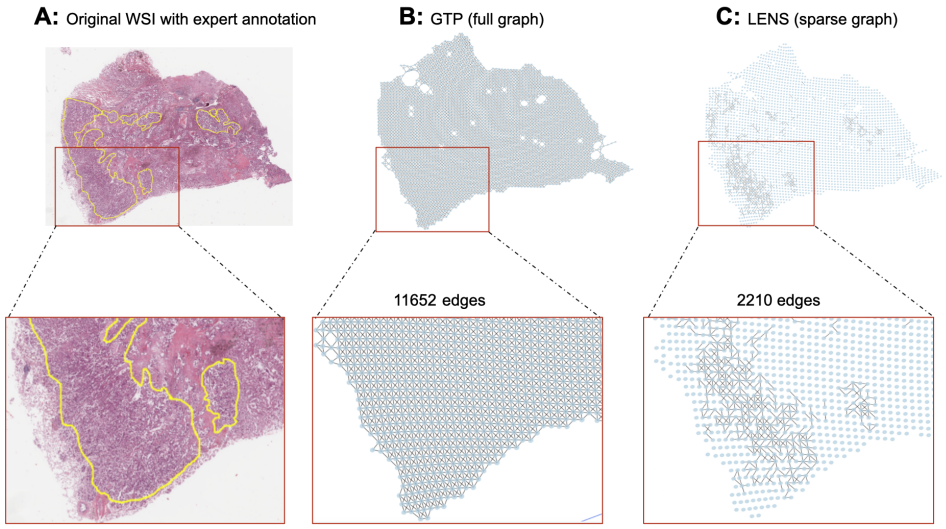

Figure G.2: **Learned sparse topology versus dense graph on a representative LUAD slide.** (A) H&E WSI with expert-annotated tumor RoIs (yellow contours); the red box marks the zoomed region. (B) Dense **GTP** graph: all 11,652 candidate edges of the 8-NN grid, spread across tumor, stroma, and non-diagnostic tissue. (C) **LENS** sparse graph: 2,210 edges kept on this slide ($\sim$19% of candidates), concentrated in the expert-annotated tumor region. The zoom-in shows that kept edges form connected local subgraphs within the tumor rather than isolated patches. The topology is learned from slide-level labels only.

H Architecture Design Ablations

All ablations use CPTAC-Lung with **UNI2-h** features and target 30% edge retention unless stated otherwise. Ablation variants are reported as mean $\pm$ std over 3 runs; the default configuration in each table is reported from the main evaluation in Table 1 (15 runs). These experiments isolate whether **LENS**'s gains come from task-supervised topology learning, the edge scorer, density control, or backbone capacity. More specifically, these ablations are designed to separate three sources of improvement: the edge-scoring function, the sparsification dynamics, and the graph-level use of the learned topology. This distinction is important because high performance could otherwise be attributed to a stronger GNN backbone, a favorable sparsity level, or post-hoc pruning rather than to task-supervised topology learning itself.

H.1 Edge Scoring Architecture

Table H.1 evaluates the edge scorer under fixed retention. Cosine similarity (**GraphLSurv**-style) trails full **LENS** by 4.2 points, showing that heuristic similarity is insufficient for task-relevant tissue-interface selection. Each architectural addition contributes: pairwise contrast adds +1.9 points over endpoint-only scoring, and channel-wise attention adds a further +0.9.

H.2 Choice of ℓ_0 Estimator

Table H.2 compares three ℓ_0-style estimators at $\sim$28–31% edge retention on CPTAC-Lung. ARM attains marginally higher accuracy ($88.6 \pm 1.2\%$) than Hard-Concrete ($88.4 \pm 1.3\%$) at the same AUROC tier, but requires $2\times$ higher training cost due to two stochastic forward passes per update. Hard-Concrete matches ARM within noise while retaining the tightest density ($28 \pm 2\%$) at $1\times$ cost. STE reaches comparable retention ($31 \pm 4\%$) but substantially lower accuracy and AUROC ($85.2 \pm 2.3\%$ / $93.6 \pm 1.5\%$), indicating that its biased gradient approximation yields less reliable sparse masks. We adopt Hard-Concrete as the default estimator because it matches ARM within noise at half the cost, outperforms STE, and provides closed-form keep probability π_{ij} required for scale-invariant density control.

Table H.1: **Edge-scoring architecture ablation on CPTAC-Lung with `UNI2-h` features.** All variants use the same backbone, Hard-Concrete gating, training protocol, and target retention $\rho_e = 30\%$; only the edge-scoring function changes. Cosine similarity represents a heuristic **`GraphLSurv`**-style score, the endpoint MLP tests learned scoring without explicit contrast, the absolute-difference variant adds pairwise relation information, and the attended-relation variant is the full **`LENS`** scorer. Realized retention remains near 28% across variants, so the 4.2-point gain of full **`LENS`** over cosine similarity reflects task-learned relation scoring rather than different sparsity levels.

Variant	Acc (%)	AUC (%)	Ret. (%)	GFLOPs
Cosine similarity	$84.2_{\pm 1.9}$	$91.9_{\pm 1.4}$	$28_{\pm 3}$	15.1
MLP on $[\boldsymbol{x}_i \| \boldsymbol{x}_j]$	$85.6_{\pm 1.8}$	$93.6_{\pm 1.2}$	$28_{\pm 2}$	16.8
+ Absolute diff	$87.5_{\pm 1.7}$	$94.6_{\pm 1.0}$	$28_{\pm 2}$	17.5
+ Attended relation (`LENS`)	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{96.3}_{\pm 0.8}$	$28_{\pm 2}$	19.2

Table H.2: **ℓ_0-estimator comparison on CPTAC-Lung at $\sim$30% target edge density.** The table compares Straight-Through Estimator (STE), Augment-REINFORCE-Merge (ARM), and Hard-Concrete ℓ_0 gating under the same backbone, target density, and training protocol. ARM reaches similar accuracy to Hard-Concrete but requires $2\times$ training cost because it uses two stochastic forward passes per update. STE has $1\times$ cost but lower accuracy and AUROC despite comparable retention. Hard-Concrete is therefore the default estimator: it matches ARM within noise at half the cost, outperforms STE, and provides closed-form keep probabilities π_{ij} needed for scale-invariant density control.

Estimator	Acc (%)	AUROC (%)	Ret. (%)	Cost
STE	$85.2_{\pm 2.3}$	$93.6_{\pm 1.5}$	$31_{\pm 4}$	$1\times$
ARM	$\mathbf{88.6}_{\pm 1.2}$	$\mathbf{96.5}_{\pm 0.7}$	$29_{\pm 2}$	$2\times$
HC-ℓ_0 (ours)	$88.4_{\pm 1.3}$	$96.3_{\pm 0.7}$	$\mathbf{28}_{\pm 2}$	$\mathbf{1}\times$

H.3 Temperature Annealing

Table H.3 compares fixed-temperature training with cosine annealing. Fixed-τ variants underperform annealing by 0.9–1.4 points. Cosine annealing from τ=5.0 $\rightarrow$ 0.67 over 20 epochs enables broad exploration early and sharp keep/drop decisions near convergence. Figure H.1 shows the mechanism: annealed schedules produce bimodal gate distributions (mass at 0 and 1), while fixed τ yields diffuse mid-range values ($\pi \approx 0.5$).

H.4 GNN Backbone and **`LENS+`** Architecture

Table H.4 separates topology learning from backbone capacity. With the same sparse topology, **`LENS-GCN`** reaches 88.4% accuracy at 19.2 GFLOPs and **`LENS-GAT`** reaches 88.6% at 24.1 GFLOPs, outperforming dense GCN by 6.0 points and dense GAT by 4.7 points while using much less compute. **`LENS+`** further improves accuracy to 91.2% through deeper graph processing and masked attention over the learned sparse topology, exceeding dense **`GTP`** by 0.7 points at $6.3\times$ lower compute.

H.5 Pooling Mechanisms

Topology-aware pooling outperforms agnostic alternatives (Table H.5). Edge-weighted pooling improves over standard attention by +0.8 points, showing that retained edge mass provides useful aggregation signal. **`LENS+ MHA`** with [CLS] adds +2.8 points by modeling global context over the sparse topology.

H.6 Size-Aware Regularization

Fixed ℓ_0 penalties create size-dependent retention: small slides retain 42%, whereas large slides only 19% (Table H.6). Graph-normalized penalties reduce the across-slide standard deviation of retention

Table H.3: **Temperature-schedule ablation for Hard-Concrete edge gates on CPTAC-Lung.** All variants use the same edge scorer, backbone, target density, and training protocol; only the gate-temperature schedule changes. Fixed-temperature runs underperform the annealed schedules, indicating that a constant τ either leaves gates diffuse or sharpens them too early. The default cosine annealing schedule, $\tau_{\text{init}} = 5.0 \rightarrow 0.67$ over 20 epochs after a 15-epoch warm-up, achieves the best accuracy/AUROC while keeping realized retention stable at $\sim$28%. This supports the training design: high temperature encourages early exploration, while low final temperature yields decisive keep/drop gates.

Schedule	τ_{init}	τ_{min}	Acc (%)	AUC (%)	Ret. (%)
Fixed temperature					
Fixed 0.67	0.67	0.67	$87.3_{\pm 1.6}$	$95.6_{\pm 0.9}$	$30_{\pm 3}$
Fixed 1.0	1.0	1.0	$87.5_{\pm 1.5}$	$95.7_{\pm 0.8}$	$29_{\pm 3}$
Fixed 2.0	2.0	2.0	$87.0_{\pm 1.7}$	$95.3_{\pm 1.0}$	$31_{\pm 4}$
Cosine annealing					
τ_{init}=2.0	2.0	0.67	$87.8_{\pm 1.4}$	$95.9_{\pm 0.8}$	$28_{\pm 2}$
τ_{init}=5.0 (default)	5.0	0.67	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{96.3}_{\pm 0.8}$	$28_{\pm 2}$
τ_{init}=10.0	10.0	0.67	$88.1_{\pm 1.5}$	$96.1_{\pm 0.8}$	$27_{\pm 2}$
Annealing horizon (τ_{init}=5.0)					
10 epochs	5.0	0.67	$87.6_{\pm 1.6}$	$95.7_{\pm 0.9}$	$29_{\pm 3}$
20 epochs (default)	5.0	0.67	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{96.3}_{\pm 0.8}$	$28_{\pm 2}$
30 epochs	5.0	0.67	$88.2_{\pm 1.4}$	$96.2_{\pm 0.8}$	$28_{\pm 2}$

Table H.4: **GNN backbone and `LENS+` architecture ablation on CPTAC-Lung with `UNI2-h` features.** Dense GCN, dense GAT, and dense **`GTP`** use the full 8-NN graph, while all **`LENS`** variants use the learned sparse topology with 28–29% edge retention. **`LENS`**-GCN reaches 88.4% accuracy at 19.2 GFLOPs and **`LENS`**-GAT reaches 88.6% at 24.1 GFLOPs, outperforming dense GCN/GAT while using substantially less compute. **`LENS+`** improves accuracy through deeper graph processing and masked multi-head attention over the learned sparse topology, reaching 91.2% accuracy and 96.7% AUROC. The table separates the effect of learned topology from the effect of backbone capacity.

Architecture	Acc (%)	AUC (%)	Ret. (%)	GFLOPs
Dense baselines (100% edges)				
GCN (3-layer)	$82.4_{\pm 2.1}$	$90.8_{\pm 1.8}$	100	103.7
GAT (3-layer)	$83.9_{\pm 2.0}$	$92.3_{\pm 1.7}$	100	118.3
GTP	$\underline{90.5}_{\pm 1.3}$	$\underline{96.5}_{\pm 0.6}$	100	152.2
LENS variants (28% edges)				
`LENS` (GCN)	$88.4_{\pm 1.4}$	$96.3_{\pm 0.8}$	$28_{\pm 2}$	19.2
`LENS` (GAT)	$88.6_{\pm 1.3}$	$96.4_{\pm 0.8}$	$28_{\pm 2}$	24.1
`LENS+` variants (29% edges)				
1-layer + edge pool	$88.4_{\pm 1.4}$	$96.3_{\pm 0.8}$	$28_{\pm 2}$	19.2
3-layer + 4-head MHA	$90.3_{\pm 1.3}$	$97.2_{\pm 0.6}$	$29_{\pm 2}$	22.8
3-layer + 8-head MHA	$\mathbf{91.2}_{\pm 1.1}$	$\mathbf{96.7}_{\pm 0.4}$	$29_{\pm 2}$	24.1

from 0.18 to 0.06; adding explicit density matching further tightens to std = 0.04 and improves accuracy by 2 points. This supports the need for WSI-specific density control rather than a raw ℓ_0 penalty.

H.7 Lambda Scheduling

This ablation tests whether sparsity control can rely on a fixed coefficient or requires feedback from the observed retention. Because WSI graphs vary substantially in size, a fixed penalty can impose different effective sparsity pressures across slides. Closed-loop scheduling corrects this by increasing

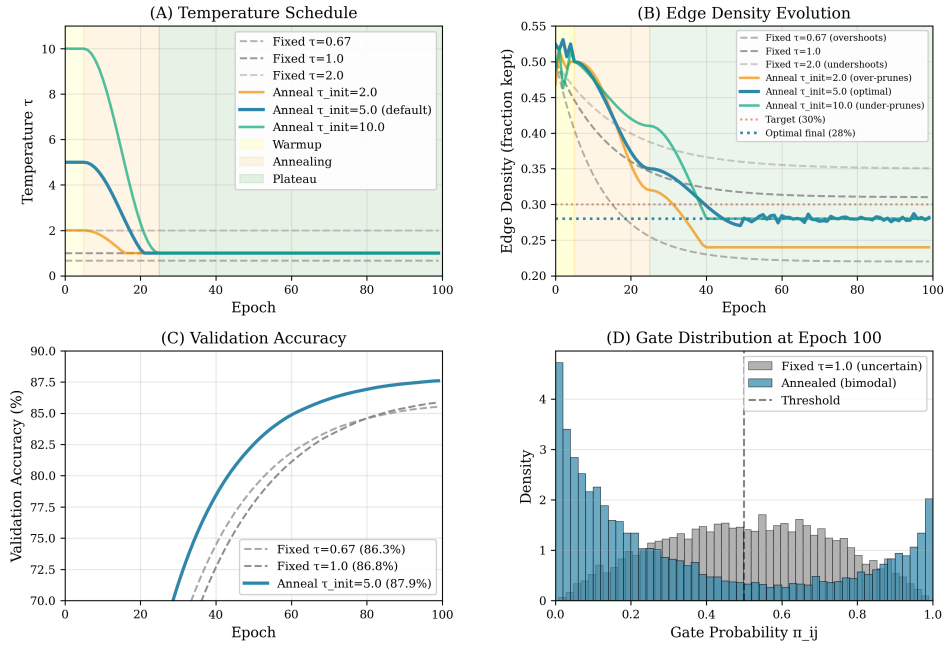

Figure H.1: **Temperature annealing dynamics for Hard-Concrete edge gates on CPTAC-Lung.** (A) Temperature schedules compare fixed-τ training with cosine annealing from τ_{init} to $\tau_{\min} = 0.67$. (B) Edge-density trajectories show that annealing reaches the target retention smoothly, while fixed temperatures lead to less stable density control. (C) Validation accuracy improves as gates transition from exploratory soft decisions to sharper keep/drop decisions. (D) Gate distributions at convergence show the mechanism: annealed schedules yield bimodal gates with mass near 0 and 1, whereas fixed-τ schedules leave many mid-range ambiguous gates. Together with Table H.3, this supports the default annealing schedule used in **LENS**.

Table H.5: **Pooling ablation on CPTAC-Lung with `UNI2-h` features.** The table compares topology-agnostic pooling for **LENS**, topology-aware pooling using the learned sparse graph, and masked multi-head attention for **LENS+**. Global mean/max pooling ignore the learned topology and reach 86.3–86.7% accuracy; standard attention improves to 87.6%, but edge-weighted pooling reaches 88.4% by using retained edge mass as the aggregation signal. **LENS+** further improves to 91.2% accuracy and 96.7% AUROC by applying masked multi-head attention with a [CLS] token over the learned sparse topology. The main message is that the learned topology is useful not only for pruning message passing, but also for graph-level aggregation.

Pooling	Acc (%)	AUC (%)
Topology-agnostic (LENS)		
Global mean	$86.3_{\pm 1.7}$	$94.7_{\pm 1.0}$
Global max	$86.7_{\pm 1.6}$	$95.0_{\pm 0.9}$
Standard attention	$87.6_{\pm 1.5}$	$95.7_{\pm 0.8}$
Topology-aware (LENS)		
Degree-weighted	$87.9_{\pm 1.4}$	$96.0_{\pm 0.8}$
Edge-weighted (default)	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{96.3}_{\pm 0.8}$
Multi-head attention (LENS+)		
4-head MHA + [CLS]	$90.3_{\pm 1.3}$	$97.2_{\pm 0.6}$
8-head MHA + [CLS]	$\mathbf{91.2}_{\pm 1.1}$	$\mathbf{96.7}_{\pm 0.4}$

pressure above the target retention and relaxing it below the target, stabilizing both density and convergence.

Table H.6: **Size-aware regularization ablation on CPTAC-Lung with `UNI2-h` features.** The table compares a raw ℓ_0 penalty, graph-normalized retention, and graph-normalized retention with explicit density matching. A raw ℓ_0 penalty produces size-dependent sparsity: small slides retain 42% of edges, whereas large slides retain only 19% of edges. Graph normalization reduces the across-slide standard deviation of retention from 0.18 to 0.06, and adding density matching further tightens it to 0.04 while improving accuracy by 2 points. This supports the need for WSI-specific density control rather than applying generic ℓ_0 sparsification directly to variable-size slide graphs.

Setting	**Acc (%)**	**Mean $\bar{\pi}$ (%)**	**Std($\bar{\pi}$)**
Unnormalized (size-dependent)			
Overall	$86.4_{\pm 1.8}$	$31_{\pm 8}$	0.18
Small (<1K)	$84.7_{\pm 2.1}$	$42_{\pm 6}$	0.12
Medium (1K–5K)	$86.6_{\pm 1.9}$	$31_{\pm 5}$	0.09
Large (>5K)	$83.6_{\pm 2.3}$	$19_{\pm 4}$	0.08
Size-normalized			
$\mathcal{L}_{\text{task}} + \lambda \bar{\pi}_{\mathcal{G}}$	$87.4_{\pm 1.5}$	$22_{\pm 3}$	0.06
$+\lambda_\rho(\bar{\pi}_{\mathcal{G}} - \rho_e)^2$ (ours)	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{28}_{\pm 2}$	**0.04**

Fixed λ causes premature commitment and misses the target retention (Table H.7, Figure H.2). Warmup schedules improve stability but remain one-directional. The adaptive schedule provides closed-loop correction, achieving a $3\times$ smaller retention standard deviation and $2\times$ faster convergence than linear warmup.

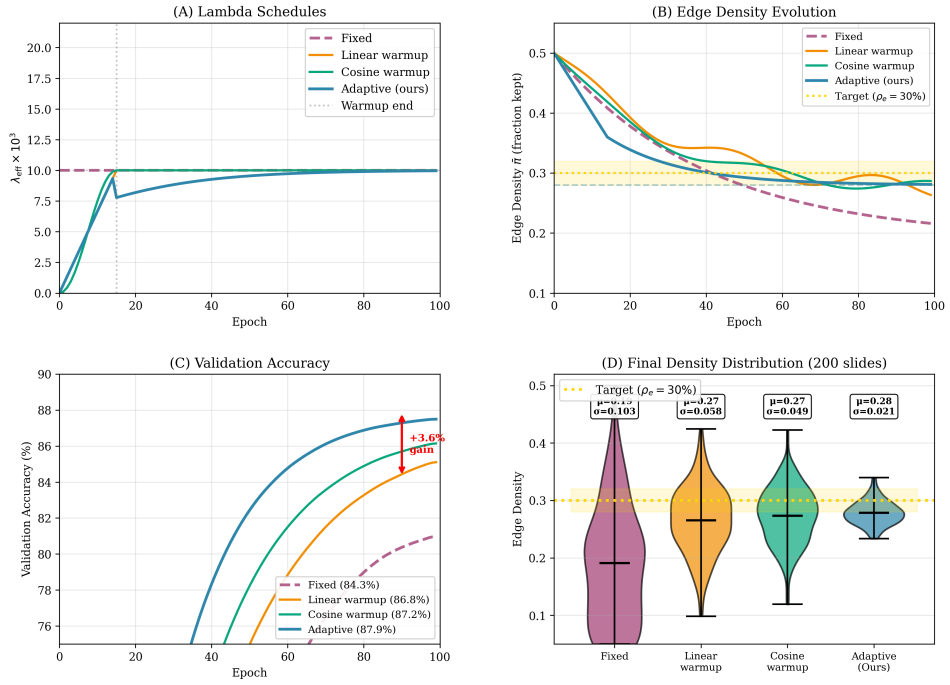

Figure H.2: **Lambda-scheduling dynamics for density-controlled ℓ_0 sparsification on CPTAC-Lung.** (A) Effective sparsity weight over training for fixed, warmup, and adaptive schedules. (B) Edge-density convergence toward the target retention ρ_e=0.30; the gold band marks the desired $\pm 2\%$ range. (C) Validation accuracy over training, showing whether each schedule reaches stable performance. (D) Final density distribution across 200 slides, measuring slide-to-slide retention stability. Together with Table H.7, the figure shows that the adaptive schedule provides closed-loop correction: it reaches $28 \pm 2\%$ retention, reduces the retention standard deviation to 0.04, and converges in 35 ± 4 epochs.

Table H.7: **Lambda-scheduling ablation for density-controlled ℓ_0 sparsification on CPTAC-Lung with `UNI2-h` features.** All variants use the same edge scorer, backbone, target density $\rho_e = 0.30$, and Hard-Concrete gates; only the sparsity-weight schedule changes. Fixed λ commits too early and misses the target retention, yielding $19 \pm 11\%$ retained edges and unstable density. Linear and cosine warmup improve stability but remain one-directional, converging in 68 ± 8 and 52 ± 6 epochs. The adaptive schedule provides closed-loop correction, reaches $28 \pm 2\%$ retention, reduces the retention standard deviation to 0.04, converges in 35 ± 4 epochs, and gives the best accuracy 88.4%. This supports adaptive density control as a necessary component for stable sparsification across variable-size WSI graphs.

Schedule	Acc (%)	$\bar{\pi}$ (%)	Std	Conv.	Stability
Fixed λ_0	$84.8_{\pm 2.1}$	$19_{\pm 11}$	0.21	>100	Unstable
Linear warmup	$87.3_{\pm 1.6}$	$26_{\pm 6}$	0.12	$68_{\pm 8}$	Moderate
Cosine warmup	$87.7_{\pm 1.5}$	$27_{\pm 5}$	0.10	$52_{\pm 6}$	Good
Adaptive (ours)	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{28}_{\pm 2}$	**0.04**	$\mathbf{35}_{\pm 4}$	Excellent

H.8 Target Density Sensitivity

Table H.8 and Figure H.3 characterize the full sparsity spectrum. This ablation tests whether the target density is a fragile hyperparameter or whether **`LENS`** admits a stable operating range. Very aggressive pruning can remove useful tissue interfaces, whereas high retention restores unnecessary dense-graph computation and weakens localization. The goal is therefore not maximum sparsity, but a controlled middle regime that preserves predictive tissue relations while reducing compute and maintaining compact topology-derived heatmaps. The 25–35% range gives the best accuracy–efficiency–interpretability trade-off: accuracy remains 87.9–88.4% at 17–21 GFLOPs with spatially concentrated topology-derived heatmaps, i.e., 7–9$\times$ less compute than dense **`GTP`**. The default 30% target lies in the center of this stable regime, while reducing compute from 103.7 GFLOPs (dense GCN) to 19.2 GFLOPs ($5.4\times$).

Overall, the density sweep shows that sparsity is useful only when controlled. Very low retention removes predictive tissue interfaces, while high retention approaches the dense-graph regime and weakens the interpretability benefit. The 25–35% range provides a stable operating region in which **`LENS`** preserves accuracy, reduces compute, and maintains compact topology-derived explanations.

Table H.8: **Target-density sensitivity for ℓ_0-controlled sparsification on CPTAC-Lung with `UNI2-h` features** (N=3,048 nodes). The table sweeps the requested edge-retention target from aggressive pruning to dense inference and reports accuracy, AUROC, realized retention $\bar{\pi}$, GFLOPs, and qualitative interpretability. Very low targets, 10–20%, produce excellent sparsity but lose accuracy because useful tissue interactions are removed. The 25–35% range gives the most stable accuracy–efficiency–interpretability trade-off, with 87.9–88.4% accuracy at 17.3–21.5 GFLOPs and spatially concentrated heatmaps. Higher densities recover more edges but provide little accuracy gain while increasing compute and degrading interpretability. The default target $\rho_e = 0.30$ lies in this stable middle regime, yielding $28 \pm 2\%$ realized retention, 88.4% accuracy, 96.3% AUROC, and 19.2 GFLOPs.

Config.	Acc (%)	AUC (%)	$\bar{\pi}$ (%)	GFLOPs	Interp.
Aggressive pruning					
10%	$83.2_{\pm 2.3}$	$91.6_{\pm 1.6}$	$10_{\pm 2}$	12.8	Excellent
15%	$85.6_{\pm 2.0}$	$94.2_{\pm 1.2}$	$15_{\pm 2}$	14.2	Excellent
20%	$87.3_{\pm 1.7}$	$95.3_{\pm 1.0}$	$20_{\pm 2}$	15.8	Excellent
Optimal range					
25%	$87.9_{\pm 1.5}$	$95.8_{\pm 0.9}$	$24_{\pm 2}$	17.3	Very good
30%	$\mathbf{88.4}_{\pm 1.4}$	$\mathbf{96.3}_{\pm 0.8}$	$28_{\pm 2}$	19.2	Very good
35%	$88.1_{\pm 1.5}$	$96.1_{\pm 0.9}$	$31_{\pm 2}$	21.5	Good
40%	$87.4_{\pm 1.7}$	$95.2_{\pm 1.1}$	$32_{\pm 3}$	24.1	Good
Dense regime					
50%	$88.2_{\pm 1.9}$	$93.9_{\pm 1.3}$	$44_{\pm 4}$	29.8	Moderate
70%	$87.7_{\pm 2.2}$	$93.3_{\pm 1.5}$	$76_{\pm 5}$	42.3	Poor
100%	$86.1_{\pm 2.4}$	$94.5_{\pm 1.9}$	100	103.7	Poor

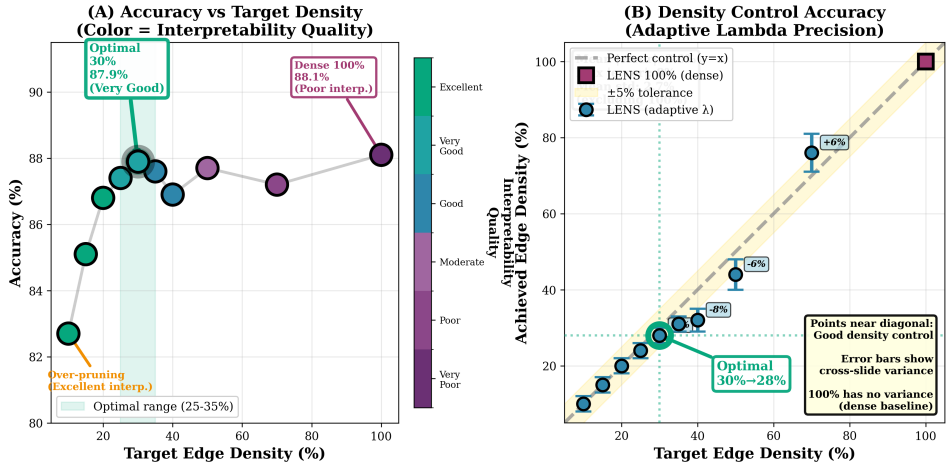

Figure H.3: **Target-density sensitivity for ℓ_0-controlled sparsification on CPTAC-Lung.** (A) Accuracy versus realized edge retention, with points colored by interpretability quality; the shaded band marks the stable 25–35% operating range where accuracy remains high and explanations remain spatially concentrated. (B) Achieved versus target density; realized retention follows the target closely between 10% and 35%, and deviates more at higher targets (e.g., 40% → 32%, 70% → 76%). Together with Table H.8, the figure shows that the default target $\rho_e = 0.30$ lies in a stable regime rather than being an arbitrary sparsity choice.

I Input Representation Ablation

Both the feature encoder and the candidate graph are fixed before training. This appendix tests three alternative explanations for the gains of **`LENS`**: a stronger feature encoder, a favorable hand-chosen

candidate graph, or patch features alone without spatial topology. We vary the encoder, the k-NN candidate pool, and the patch coordinates one at a time.

I.1 Feature Encoder

The main paper uses UNI2-h as the primary encoder. We repeat all experiments with two additional encoders to confirm that **LENS**'s advantages are architectural, not feature-driven: **CTransPath** (Swin-T, 768-dim, WSI-aware contrastive pretraining on 15M patches (Wang et al., 2022)) and **SimCLR** (ResNet-18, 512-dim, contrastive SSL on lung histopathology). All other settings are fixed: 8-NN graph, ρ_e=0.30, identical splits and baselines. Only the input projection dimension is adapted per encoder. All eight baselines from the main paper are included in both tables.

Table I.1: **Encoder ablation using `CTransPath` features.** The table repeats the main evaluation protocol with a 768-dimensional `CTransPath` encoder while keeping the same splits, 8-NN candidate graphs, target density $\rho_e = 0.30$, baselines, and model-selection procedure. The purpose is to test whether `LENS` depends on the `UNI2-h` foundation encoder. `LENS+` remains best or second-best across all four benchmarks, showing that the topology-learning advantage is preserved under a different pathology encoder. **Bold** indicates best and underline indicates second-best.

	CPTAC-Lung			TCGA-LUNG		TCGA-RCC		TCGA-BRCA	
Method	Acc	AUC	GFLOPs	Acc	AUC	Acc	AUC	Acc	AUC
Multiple Instance Learning									
ABMIL	$84.6_{\pm 1.7}$	$93.1_{\pm 1.1}$	25.4	$77.2_{\pm 1.8}$	$86.4_{\pm 1.4}$	$86.9_{\pm 1.6}$	$94.6_{\pm 1.0}$	$82.8_{\pm 1.7}$	$92.1_{\pm 1.2}$
TransMIL	$85.4_{\pm 1.5}$	$94.1_{\pm 1.0}$	36.4	$78.4_{\pm 1.6}$	$87.6_{\pm 1.3}$	$87.8_{\pm 1.4}$	$95.4_{\pm 0.9}$	$83.6_{\pm 1.5}$	$92.8_{\pm 1.1}$
CLAM-SB	$86.3_{\pm 1.6}$	$95.1_{\pm 0.8}$	30.7	$78.9_{\pm 1.5}$	$88.4_{\pm 1.1}$	$88.6_{\pm 1.3}$	$95.8_{\pm 0.8}$	$84.4_{\pm 1.4}$	$93.2_{\pm 1.0}$
CLAM-MB	$87.1_{\pm 1.4}$	$\underline{95.9}_{\pm 0.7}$	31.2	$79.8_{\pm 1.4}$	$89.1_{\pm 1.0}$	$89.4_{\pm 1.2}$	$96.4_{\pm 0.7}$	$\mathbf{85.6}_{\pm 1.3}$	$\mathbf{93.9}_{\pm 0.9}$
Dense Graph Neural Networks									
GCN	$81.4_{\pm 2.1}$	$89.9_{\pm 1.6}$	102.1	$75.3_{\pm 2.2}$	$84.6_{\pm 1.8}$	$83.7_{\pm 2.0}$	$92.4_{\pm 1.5}$	$78.9_{\pm 2.1}$	$88.7_{\pm 1.6}$
Patch-GCN	$84.1_{\pm 1.9}$	$91.8_{\pm 1.5}$	173.5	$77.6_{\pm 1.9}$	$86.2_{\pm 1.6}$	$85.4_{\pm 1.8}$	$94.1_{\pm 1.3}$	$80.9_{\pm 1.9}$	$90.4_{\pm 1.5}$
GTP	$\underline{88.4}_{\pm 1.4}$	$95.8_{\pm 0.8}$	148.6	$\underline{80.8}_{\pm 1.4}$	$\underline{89.6}_{\pm 1.1}$	$88.1_{\pm 1.4}$	$96.1_{\pm 0.8}$	$84.6_{\pm 1.4}$	$93.4_{\pm 1.0}$
Sparse Graph Neural Networks									
GraphLSurv	$82.9_{\pm 1.9}$	$91.0_{\pm 1.3}$	25.5	$76.4_{\pm 2.0}$	$85.7_{\pm 1.6}$	$84.3_{\pm 1.9}$	$93.4_{\pm 1.3}$	$77.8_{\pm 2.2}$	$88.1_{\pm 1.6}$
LENS	$87.1_{\pm 1.5}$	$95.4_{\pm 0.9}$	**19.1**	$80.1_{\pm 1.7}$	$89.4_{\pm 1.3}$	$\underline{90.1}_{\pm 1.6}$	$\underline{97.1}_{\pm 0.9}$	$81.2_{\pm 1.8}$	$91.0_{\pm 1.4}$
LENS+	$\mathbf{89.8}_{\pm 1.3}$	$\mathbf{96.9}_{\pm 0.6}$	$\underline{24.0}$	$\mathbf{83.1}_{\pm 1.5}$	$\mathbf{91.4}_{\pm 1.1}$	$\mathbf{91.2}_{\pm 1.0}$	$\mathbf{97.8}_{\pm 0.6}$	$\underline{84.8}_{\pm 1.2}$	$\underline{93.6}_{\pm 0.9}$

Observations. (1) Rank ordering is largely preserved across encoders. `LENS+` ranks first or second on every dataset under both **`CTransPath`** and **`SimCLR`**, matching the **`UNI2-h`** pattern in Table 1. The advantage of topology learning therefore does not depend on one particular encoder.

(2) On CPTAC-Lung, `LENS+` matches or surpasses `GTP` under each encoder at $\sim 6\times$ lower compute. With **`SimCLR`**, **`LENS+`** reaches 90.7% against 89.9% for **`GTP`**, using 23.8 vs. 142.3 GFLOPs. With **`CTransPath`**, **`LENS+`** reaches 89.8% against 88.4% for **`GTP`** with the same encoder.

(3) `LENS+` depends less on the encoder than MIL does. CLAM-MB improves by $+3.4$ accuracy points from SimCLR to **`UNI2-h`**; **`LENS+`** improves by only $+0.5$ points yet maintains a higher baseline at every encoder level (CPTAC-Lung: $+5.8$ pts over CLAM-MB under SimCLR, $+2.9$ pts under **`UNI2-h`**). The gap narrows as encoders improve, because MIL benefits more from stronger patch features, but it does not close: edge structure adds information that better patch features alone do not provide.

(4) Sparsity is encoder-agnostic. Mean edge retention is 28.4%, 28.1%, and 28.2% under SimCLR, CTransPath, and **`UNI2-h`** respectively, all within $\pm 3\%$ of the ρ_e=0.30 target. The adaptive λ mechanism stabilizes density regardless of the feature distribution.

I.2 Candidate Graph Construction (k-NN)

LENS learns to gate edges from a fixed candidate graph built before training. We ablate $k \in \{4, 8, 16, 32\}$ to establish whether there is a coverage floor below which missing candidate edges

Table I.2: **Encoder ablation using `SimCLR` ResNet-18 features.** The table repeats the main evaluation protocol with a 512-dimensional `SimCLR` encoder while keeping the same splits, 8-NN candidate graphs, target density $\rho_e = 0.30$, baselines, and model-selection procedure. This tests whether `LENS` depends on high-capacity foundation-model features. Despite using a smaller self-supervised encoder, `LENS+` remains competitive across all four benchmarks and reaches 90.7% accuracy / 95.6% AUROC on CPTAC-Lung at 23.8 GFLOPs. Together with Table I.1, this shows that the topology-learning advantage is preserved across feature encoders rather than being specific to `UNI2-h`. **Bold** indicates best and underline indicates second-best.

	CPTAC-Lung			TCGA-LUNG		TCGA-RCC		TCGA-BRCA	
Method	Acc	AUC	GFLOPs	Acc	AUC	Acc	AUC	Acc	AUC
Multiple Instance Learning									
ABMIL	$81.3_{\pm 1.9}$	$90.4_{\pm 1.4}$	24.9	$74.8_{\pm 2.0}$	$83.9_{\pm 1.6}$	$85.1_{\pm 1.8}$	$93.2_{\pm 1.2}$	$80.6_{\pm 1.9}$	$90.4_{\pm 1.4}$
TransMIL	$83.2_{\pm 1.8}$	$91.8_{\pm 1.3}$	32.6	$76.1_{\pm 1.8}$	$85.4_{\pm 1.4}$	$86.3_{\pm 1.6}$	$94.1_{\pm 1.1}$	$81.8_{\pm 1.7}$	$91.2_{\pm 1.3}$
CLAM-SB	$84.1_{\pm 1.7}$	$93.2_{\pm 1.1}$	29.5	$76.8_{\pm 1.7}$	$86.2_{\pm 1.3}$	$87.0_{\pm 1.5}$	$94.6_{\pm 1.0}$	$82.6_{\pm 1.6}$	$91.8_{\pm 1.2}$
CLAM-MB	$84.9_{\pm 1.6}$	$\underline{93.8}_{\pm 1.0}$	30.9	$77.4_{\pm 1.6}$	$87.1_{\pm 1.2}$	$87.6_{\pm 1.4}$	$95.2_{\pm 0.8}$	$\mathbf{83.4}_{\pm 1.5}$	$\mathbf{92.6}_{\pm 1.0}$
Dense Graph Neural Networks									
GCN	$79.1_{\pm 2.2}$	$88.2_{\pm 1.7}$	103.7	$72.8_{\pm 2.3}$	$81.9_{\pm 1.9}$	$82.4_{\pm 2.1}$	$91.3_{\pm 1.5}$	$77.2_{\pm 2.2}$	$87.1_{\pm 1.7}$
Patch-GCN	$82.3_{\pm 2.0}$	$90.6_{\pm 1.6}$	178.4	$75.4_{\pm 2.0}$	$84.7_{\pm 1.6}$	$84.3_{\pm 1.9}$	$93.2_{\pm 1.4}$	$79.6_{\pm 2.0}$	$89.4_{\pm 1.6}$
GTP	$\underline{89.9}_{\pm 1.4}$	$93.4_{\pm 1.0}$	142.3	$\underline{80.6}_{\pm 1.5}$	$\underline{90.4}_{\pm 1.2}$	$86.2_{\pm 1.6}$	$94.8_{\pm 1.0}$	$82.3_{\pm 1.6}$	$91.6_{\pm 1.2}$
Sparse Graph Neural Networks									
GraphLSurv	$80.1_{\pm 2.1}$	$88.6_{\pm 1.5}$	24.1	$74.1_{\pm 2.2}$	$83.2_{\pm 1.7}$	$82.8_{\pm 2.1}$	$92.1_{\pm 1.5}$	$76.9_{\pm 2.3}$	$87.2_{\pm 1.7}$
`LENS`	$87.9_{\pm 1.6}$	$93.2_{\pm 1.0}$	**18.9**	$78.3_{\pm 1.8}$	$87.6_{\pm 1.4}$	$\underline{88.1}_{\pm 1.7}$	$\underline{95.8}_{\pm 1.1}$	$79.4_{\pm 1.9}$	$89.2_{\pm 1.5}$
`LENS+`	$\mathbf{90.7}_{\pm 1.4}$	$\mathbf{95.6}_{\pm 0.8}$	$\underline{23.8}$	$\mathbf{81.4}_{\pm 1.6}$	$\mathbf{92.8}_{\pm 1.2}$	$\mathbf{89.6}_{\pm 1.2}$	$\mathbf{96.4}_{\pm 0.8}$	$\underline{83.1}_{\pm 1.4}$	$\underline{91.8}_{\pm 1.0}$

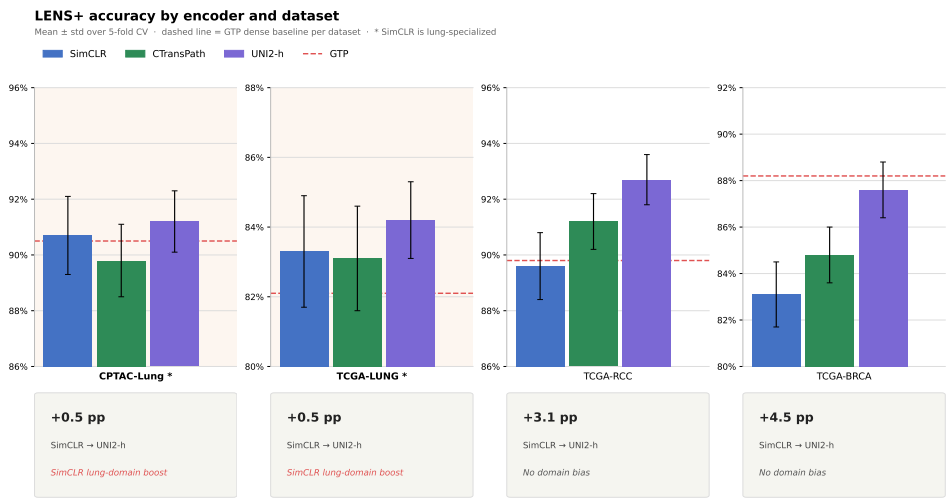

Figure I.1: **Encoder robustness of `LENS+` across four benchmarks.** The figure compares **`LENS+`** accuracy using three frozen feature encoders: **`SimCLR`**, **`CTransPath`**, and **`UNI2-h`**. All runs use the same splits, 8-NN candidate graph, target density $\rho_e = 0.30$, and model-selection protocol; only the feature encoder changes. **`LENS+`** remains best or second-best under all three encoders across all four datasets, showing that the topology-learning advantage is not specific to **`UNI2-h`**. On lung cohorts, **`SimCLR`** remains close to **`UNI2-h`** because of lung-specific pretraining; on out-of-domain cohorts, the encoder gap widens. Sparsity remains stable at 28–29% retention across encoders, indicating that density control is robust to feature representation. Dashed lines show the dense **`GTP`** baseline for each dataset.

hurt performance regardless of scoring quality, and a ceiling above which extra candidates are simply gated away.

Setup. All experiments use **UNI2-h** on CPTAC-Lung (n=2,071), 5-fold CV, identical scorer g_φ and adaptive λ scheduler with ρ_e=0.30. For 4-NN we use cardinal ("+") connectivity: each patch connects to its four axis-aligned neighbors, without diagonals.

Table I.3: **Candidate graph construction ablation on CPTAC-Lung with UNI2-h features.** The table varies the k-NN graph used as the candidate edge pool before **LENS** gating, while keeping the same edge scorer, Hard-Concrete gates, target density $\rho_e = 0.30$, and training protocol. The 4-NN graph is cheaper but creates a coverage floor: some diagnostically relevant tissue interfaces are absent from the candidate pool and cannot be recovered by the scorer. The 8-NN graph achieves the best observed accuracy–compute trade-off, reaching 88.4% accuracy and 96.3% AUROC at $28 \pm 2\%$ retention and 19.2 GFLOPs. Increasing to 16-NN or 32-NN adds candidates and compute without improving accuracy, because the learned gates remove many redundant longer-range edges. This supports using 8-NN as a sufficiently rich but not overly dense candidate graph. **Bold** indicates best and underline indicates second-best.

Config	Acc (%)	AUROC (%)	Ret. (%)	GFLOPs
4-NN (+)	$85.8_{\pm 1.8}$	$94.1_{\pm 1.1}$	$29_{\pm 3}$	16.8
8-NN (ours)	$\mathbf{88.4}_{\pm 1.3}$	$\mathbf{96.3}_{\pm 0.7}$	$\mathbf{28}_{\pm \mathbf{2}}$	<u>19.2</u>
16-NN	<u>87.6</u>$_{\pm 1.5}$	<u>95.7</u>$_{\pm 0.9}$	$28_{\pm 2}$	23.4
32-NN	$86.8_{\pm 1.7}$	$95.1_{\pm 1.1}$	$28_{\pm 2}$	31.7

Observations.

(1) There is a coverage floor at k=4. The drop from k=8 to k=4 (-2.6 points) is $1.6\times$ larger than the drop from k=8 to k=32 (-1.6 points). The asymmetry is structural: at 4-NN, some diagnostically critical tissue interfaces have *no candidate edge crossing them* in the first place. The scorer cannot retain what the candidate pool does not contain. This is confirmed by connectivity: k=4 produces 4.2 mean connected components per slide after gate freezing vs. 2.1 at k=8, with 1.8% isolated nodes vs. 0.4%.

(2) Performance is stable above the floor ($k \in \{8, 16, 32\}$). Accuracy varies by only 1.6 points across this range, and relative retention stays at $28 \pm 2\%$ throughout. When k doubles from 8 to 16, **LENS** gates out the additional candidates rather than exploiting them: the Jaccard similarity of retained edge sets between these two conditions is 0.71 ± 0.09.

(3) k=8 gives the best observed accuracy–compute–interpretability trade-off. It sits above the coverage floor, achieves highest accuracy and AUROC, and costs 39% fewer GFLOPs than k=32. The 12% efficiency gain of k=4 (16.8 vs. 19.2 GFLOPs) does not justify its accuracy cost when k=8 is already $7.9\times$ cheaper than the dense GTP baseline.

Thus, the candidate graph must be rich enough to include diagnostic tissue interfaces, but adding many extra neighbors mainly increases compute because the learned gates remove redundant candidates.

I.3 Spatial Structure: Topology vs. Features

The main paper reports binary permutation (0% vs. 100%). This section presents the full analysis: connectivity statistics on the learned sparse graph, hop-reachability under original vs. permuted coordinates, and a partial permutation gradient that identifies the percolation threshold at which spatial structure collapses.

Setup. All experiments use **LENS** with **UNI2-h** features on CPTAC-Lung (n=2,071, 5-fold CV). For the partial permutation, a fraction p of patch coordinates is randomly shuffled while features and tiling are held fixed, then the 8-NN graph is rebuilt from the resulting mixed coordinates. This preserves the feature distribution and graph density while progressively destroying spatial coherence.

The exponential growth in reachable nodes from 1-hop to 4-hop under the original graph ($5.3 \rightarrow 270$) shows that the learned sparse topology forms large, well-connected diagnostic regions rather than isolated patch pairs. Under permutation, reachability collapses ($1.3 \rightarrow 5.5$ across four hops), consistent with the fragmentation observed in Table I.4.

Observations.

Table I.4: **Connectivity statistics for the learned sparse graph under original versus permuted tissue geometry.** Statistics are averaged over 50 CPTAC-Lung test slides after gate freezing. The original spatial graph produces connected diagnostic regions: RoI node degree is 5.3, average connected component size is 847 patches, and only 0.4% of nodes are isolated. After coordinate permutation, the learned topology fragments: RoI node degree drops to 1.3, average component size falls to 56 patches, largest-component tissue coverage decreases from 17% to 3%, and isolated nodes increase to 8.2%. This shows that **LENS** relies on spatial tissue geometry, not patch features alone, to form prediction-relevant message-passing structure.

Metric	Original graph	Permuted graph
Avg. node degree in RoI regions	5.3	1.3
Avg. connected component size	847 patches	56 patches
Largest CC coverage (% of tissue)	17%	3%
Isolated nodes (%)	0.4	8.2

Table I.5: **Multi-hop reachability in the learned sparse graph under original versus permuted tissue geometry.** The table reports the average number of distinct nodes reachable within 1–4 hops after gate freezing, averaged over the same 50 CPTAC-Lung test slides as Table I.4. Under the original spatial graph, reachability expands from 5.3 nodes at 1-hop to 270 nodes at 4-hop, showing that the learned sparse topology forms extended diagnostic regions rather than isolated patch pairs. After coordinate permutation, reachability collapses from 1.3 to only 5.5 nodes across four hops, indicating fragmented, weakly structured message-passing paths. Together with Table I.4, this supports the conclusion that **LENS** relies on spatial tissue geometry to build prediction-relevant sparse topology.

Hop distance	Original graph	Permuted graph
1-hop	5.3	1.3
2-hop	24.0	2.4
3-hop	95.2	3.8
4-hop	270	5.5

(1) Accuracy falls steadily as spatial structure is destroyed. Accuracy drops by 2.3 points up to 25% permutation and by a further 6.9 points from 25% to 100%, at a roughly constant rate of about one point per 10% of shuffled coordinates. Mean component size shrinks in step, from 388 to 56 patches.

(2) Gate bimodality tracks spatial coherence. Bimodality at 25% permutation (0.78) remains close to the unperturbed baseline (0.84), indicating that the scorer still produces decisive keep/drop decisions when $\sim$75% of spatial relationships are intact. Beyond 50% permutation, bimodality falls further alongside component size, indicating that the scoring mechanism itself degrades when it can no longer find coherent tissue boundaries to contrast.

(3) Most of the 9.2-point gap is lost before full permutation. About 7.1 points are lost up to 75% permutation and 2.1 points between 75% and 100%. **LENS** therefore depends on local spatial structure throughout, not only on global connectivity.

Qualitative attribution under permutation. Figure I.2 shows representative CPTAC-Lung slides under original and fully permuted coordinates. Under the original spatial graph, **LENS** produces compact, focused attribution on coherent diagnostic regions consistent with expert annotations, with most tissue receiving near-zero weight. Under permutation, attribution becomes diffuse across the slide with no region accumulating high edge mass, consistent with the connectivity fragmentation in Table I.4.

Overall, Appendix I shows that **LENS** does not depend on a single feature encoder or a hand-picked candidate graph; instead, its gains are tied to meaningful spatial tissue geometry. Performance remains stable across encoders, 8-NN provides the best accuracy–compute trade-off among candidate graphs, and spatial permutation degrades both accuracy and heatmap structure. These results support the conclusion that **LENS** gains come from task-relevant topology learning rather than encoder choice, graph density, or patch features alone. This behavior is expected for a topology-learning

Table I.6: **Partial spatial-permutation ablation for `LENS` on CPTAC-Lung with `UNI2-h` features.** A fraction p of patch coordinates is randomly shuffled while patch features are held fixed, and the 8-NN graph is rebuilt from the perturbed coordinates. This preserves the feature distribution and graph density while progressively destroying spatial tissue organization. Accuracy degrades mildly up to 25% permutation, from 88.4% to 86.1%, but drops more sharply once 50% or more of the coordinates are permuted, reaching 79.2% under full permutation. Mean component size falls from 388 to 56 patches and gate bimodality drops from 0.84 to 0.58, showing that disrupted spatial geometry fragments the learned sparse graph and weakens decisive keep/drop edge selection. This table supports the conclusion that **`LENS`** relies on spatial tissue organization in addition to patch-level features.

Permuted (%)	Acc (%)	AUROC (%)	Mean Comp. Size	Gate Bimodality
0	$88.4_{\pm 1.3}$	$96.3_{\pm 0.7}$	$388_{\pm 142}$	$0.84_{\pm 0.06}$
10	$87.6_{\pm 1.5}$	$95.8_{\pm 0.9}$	$341_{\pm 138}$	$0.82_{\pm 0.07}$
25	$86.1_{\pm 1.7}$	$94.7_{\pm 1.1}$	$287_{\pm 129}$	$0.78_{\pm 0.08}$
50	$83.6_{\pm 2.0}$	$92.6_{\pm 1.4}$	$198_{\pm 112}$	$0.71_{\pm 0.10}$
75	$81.3_{\pm 2.2}$	$90.9_{\pm 1.6}$	$124_{\pm 87}$	$0.64_{\pm 0.12}$
100	$79.2_{\pm 2.3}$	$89.1_{\pm 1.8}$	$56_{\pm 41}$	$0.58_{\pm 0.14}$

method: when tissue geometry is destroyed, the candidate graph no longer encodes meaningful tissue interfaces to select.

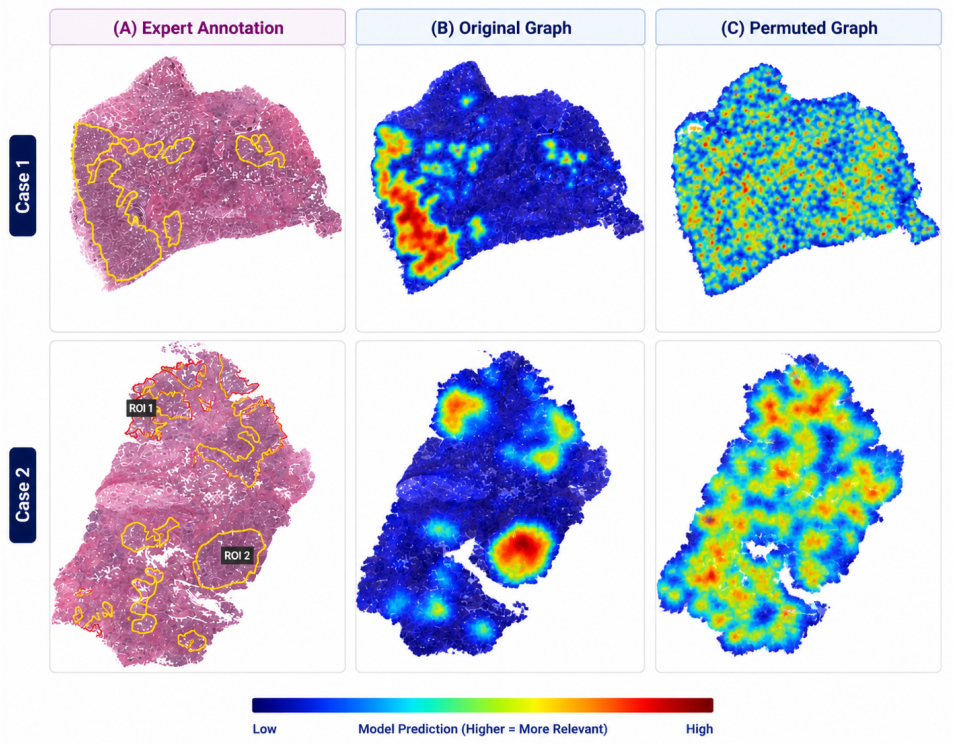

Figure I.2: **Effect of spatial permutation on topology-derived heatmaps for two CPTAC-Lung WSIs.** The top row shows a LUAD slide and the bottom row shows an LSCC slide. Each row contains three panels: (A) the original WSI with expert-annotated RoIs shown as yellow contours, (B) the **LENS** topology-derived heatmap under the original spatial geometry, and (C) the **LENS** heatmap after full coordinate permutation. Warmer heatmap colors indicate higher retained edge mass. Patch features are kept fixed, patch coordinates are fully permuted, and the 8-NN candidate graph is rebuilt before applying **LENS**. Under the original geometry, retained edge mass concentrates near the expert-annotated diagnostic regions. Under full permutation, retained edge mass disperses across the slide instead of forming a coherent diagnostic region, and the heatmap no longer aligns with the expert RoIs. This isolates spatial topology, not patch features, as the disrupted factor. This qualitative collapse matches the connectivity and reachability statistics in Tables I.4–I.5 and supports the conclusion that **LENS** relies on spatial tissue organization in addition to patch-level features.

J Interpretability and Quantitative Evaluation

A board-certified pathologist (15+ years experience) annotated 10 CPTAC-Lung WSIs (5 LUAD, 5 LSCC) while blinded to model predictions, following standard criteria: glandular/acinar and lepidic growth for LUAD; solid growth with squamous differentiation for LSCC. The same pathologist also drew tumor contours on representative TCGA-BRCA and TCGA-RCC slides; these are used for qualitative figures only (Figs. J.5–J.6). All model outputs were converted to slide-level heatmaps at the same patch resolution and evaluated against the same expert RoIs. For quantitative comparison, each heatmap was normalized per slide and binarized with the same rule for all methods. Precision, recall, IoU, Dice, and false-positive area (FP, highlighted area outside the expert RoI as a percentage of tissue area) were then computed on the same binary masks for every method. Aggregate alignment is reported in the main text (Table 3); Table J.1 reports the subtype-stratified breakdown. Figures J.1–J.3 provide case-level and multi-slide qualitative comparisons. Figures J.4–J.6 show RoI localization on CPTAC-Lung, TCGA-BRCA, and TCGA-RCC.

Why a shared binarization rule matters. Because every heatmap goes through the same rule and the same metrics, the differences in Table J.1 and Table 3 come from where each method places its evidence, not from post-processing. Reporting precision, recall, and FP together also shows whether a method gains precision by highlighting too little or gains recall by highlighting too much.

Table J.1: **Subtype-stratified expert-RoI alignment on CPTAC-Lung ($n = 5$ LUAD, $n = 5$ LSCC).** Same blinded protocol as Table 3; mean±std over 5 slides per subtype. Pooling the two subtypes gives the values in Table 3. **LENS** keeps high precision and recall on both subtypes, with 97.8% precision on LUAD and 96.4% on LSCC. **GTP** shows high variability on LUAD, while **CLAM** and **GTP** produce higher false-positive area than **LENS** in both subtypes. This supports the conclusion that the learned sparse topology aligns with subtype-relevant tissue regions rather than applying a uniform pruning pattern.

	Subtype	Prec. (%) ↑	Rec. (%) ↑	IoU ↑	Dice ↑	FP (%) ↓
LENS	LUAD	$\mathbf{97.8}_{\pm 2.2}$	$\mathbf{91.8}_{\pm 4.6}$	$\mathbf{0.91}_{\pm .04}$	$\mathbf{0.95}_{\pm .02}$	$\mathbf{2.2}_{\pm 1.7}$
	LSCC	$\mathbf{96.4}_{\pm 3.3}$	$\mathbf{93.1}_{\pm 3.9}$	$\mathbf{0.89}_{\pm .04}$	$\mathbf{0.94}_{\pm .02}$	$\mathbf{3.6}_{\pm 2.4}$
GTP	LUAD	$67.2_{\pm 13.0}$	$84.3_{\pm 10.6}$	$0.58_{\pm .07}$	$0.73_{\pm .05}$	$15.4_{\pm 5.2}$
	LSCC	$89.6_{\pm 6.2}$	$87.2_{\pm 9.4}$	$0.76_{\pm .05}$	$0.86_{\pm .04}$	$6.2_{\pm 2.9}$
CLAM	LUAD	$72.2_{\pm 8.8}$	$82.6_{\pm 11.2}$	$0.62_{\pm .12}$	$0.77_{\pm .09}$	$13.8_{\pm 5.8}$
	LSCC	$80.2_{\pm 7.4}$	$85.2_{\pm 10.2}$	$0.68_{\pm .14}$	$0.81_{\pm .10}$	$10.4_{\pm 4.6}$

Interpretation of scope. This study evaluates structural alignment, not clinical deployment readiness: the single-expert, ten-slide design tests whether the learned sparse topology aligns with pathology-relevant RoIs under a controlled head-to-head protocol. Larger multi-reader, multi-institution studies are required to assess inter-reader variability, site/scanner/stain robustness, and clinical utility.

J.1 Comparison with Existing Interpretability Evaluations

Existing interpretability evaluations in the WSI setting are often limited in quantitative scope. For the methods compared here, **CLAM** (Lu et al., 2021) reports qualitative attention heatmaps but does not provide quantitative RoI-alignment metrics, expert-annotated RoIs, or statistical testing. **GTP** (Zheng et al., 2022) reports threshold-maximized IoU on a subset of slides, but does not report the full precision–recall–false-positive profile, blinded expert annotation, or subtype-stratified analysis.

In contrast, we evaluate **LENS**, **CLAM**, and **GTP** under the same blinded protocol described above (pathologist-annotated RoIs on 10 CPTAC-Lung WSIs; one binarization rule for all heatmaps; five alignment metrics on identical masks), and additionally report:

- **Statistical testing:** paired t-tests over the 10 slides compare each baseline with **LENS**; precision, Dice, and FP are significantly better for **LENS** against both baselines ($p<0.01$, Table 3).
- **Subtype stratification:** results are reported separately for LUAD and LSCC (Table J.1).
- **Qualitative consistency:** topology-derived heatmaps are visualized on three datasets, including case-level RoI overlap on CPTAC-Lung (Figures J.1–J.4) and representative slides from TCGA-BRCA and TCGA-RCC (Figures J.5–J.6).

Applying one binarization rule and the same metrics to all methods makes localization errors directly comparable across aggregate, subtype, and case-level analyses.

Overall, this section shows that **LENS** produces compact topology-derived heatmaps that align with blinded expert RoIs, with lower false-positive area than the compared baselines.

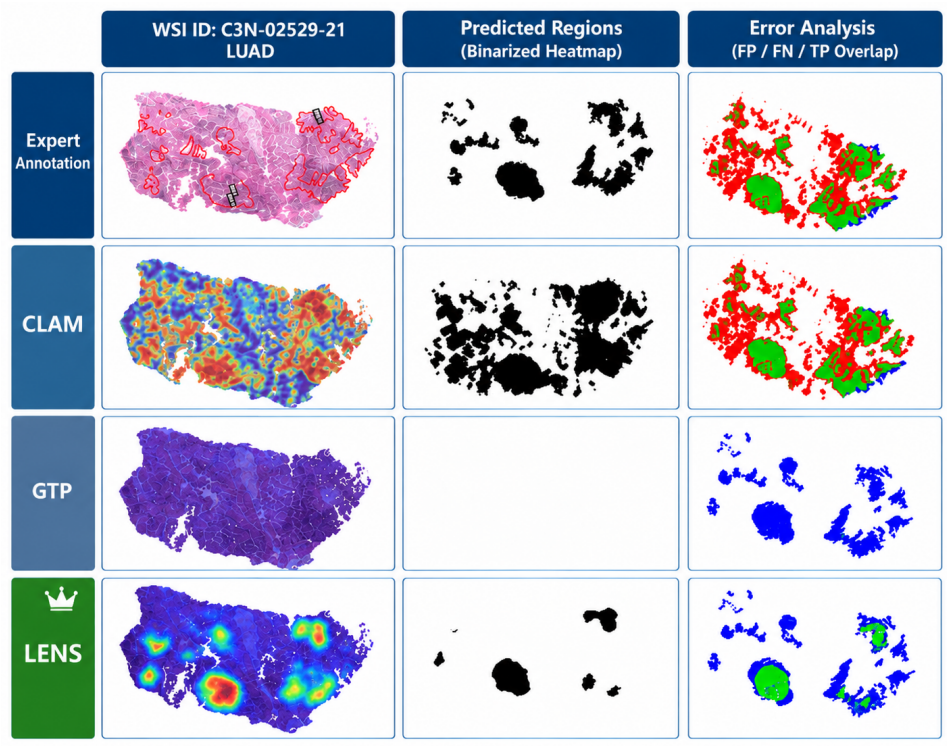

Figure J.1: **Case-level expert-RoI overlap analysis for a representative LUAD slide from CPTAC-Lung (C3N-02529-21).** Columns show the WSI with expert RoI, the binarized model mask, and the overlap map; rows compare the expert reference, **`CLAM`**, **`GTP`**, and **`LENS`**. In the overlap maps, green denotes true positive area shared by the model and expert RoI, red denotes false positive model-highlighted area outside the RoI, and blue denotes false negative expert RoI area missed by the model. **`LENS`** concentrates on glandular tumor structures, achieving 98.1% precision with only 2.3% false-positive area. **`CLAM`** over-activates into surrounding stroma, producing 21.6% false-positive area, while **`GTP`** misses 24% of the annotated RoI and has 22.4% false-positive area. This case illustrates the aggregate pattern in Table 3 and Table J.1: **`LENS`** aligns its topology-derived mask with the expert RoI, giving compact RoI-centered localization with substantially fewer false positives.

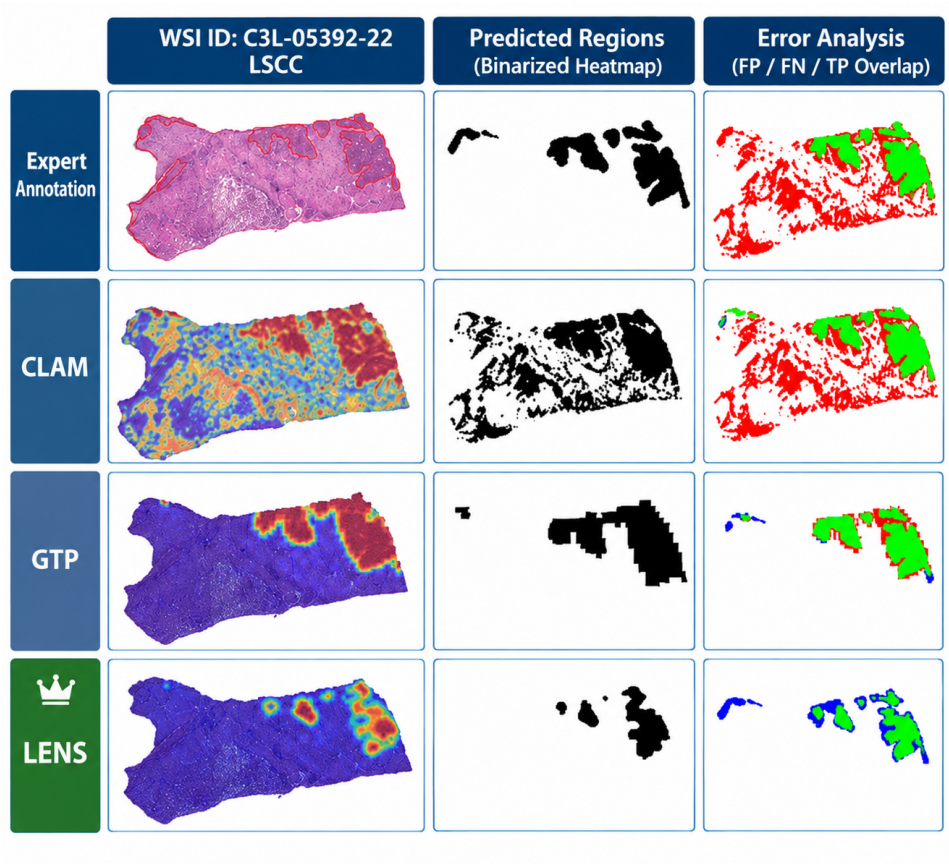

Figure J.2: **Case-level expert-RoI overlap analysis for a representative LSCC slide from CPTAC-Lung (C3L-05392-22).** Layout and overlap-map color coding follow Fig. J.1. **`LENS`** localizes solid squamous tumor growth, achieving 95.9% precision with only 3.6% false-positive area. **`CLAM`** shows diffuse off-RoI activation with 16.8% false-positive area, while **`GTP`** produces coarser boundaries with 10.2% false-positive area. This LSCC case illustrates the same pattern as the LUAD case in Fig. J.1: **`LENS`** aligns its topology-derived mask with the expert RoI, giving compact subtype-relevant localization with fewer false positives.

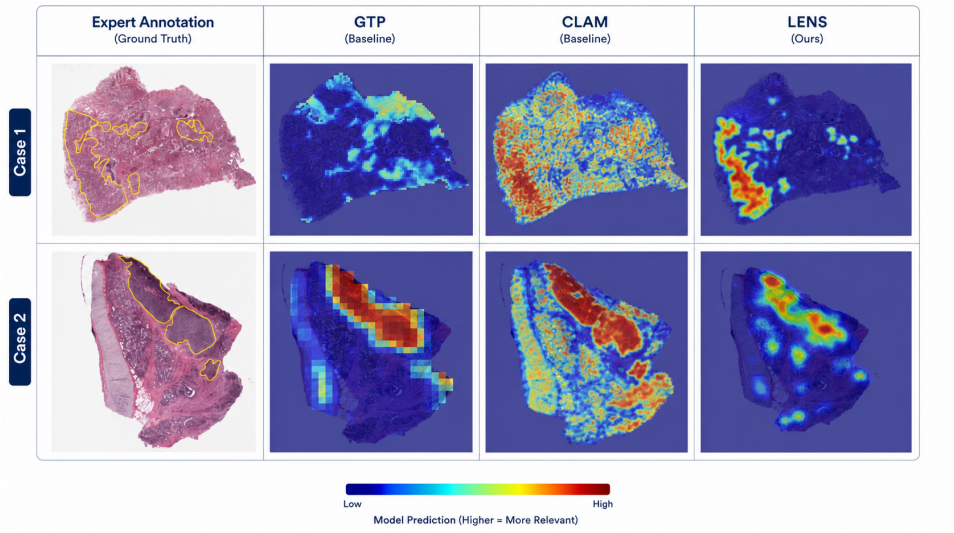

Figure J.3: **Multi-slide qualitative comparison of expert RoIs and model heatmaps on CPTAC-Lung.** Each row shows a representative CPTAC-Lung slide, and columns compare the expert annotation, **GTP** GraphCAM attribution, **CLAM** attention heatmap, and **LENS** topology-derived heatmap. Yellow contours mark expert RoIs, and warmer heatmap colors indicate stronger model-highlighted evidence. **GTP** often produces coarse or unstable localization, while **CLAM** tends to generate diffuse halos beyond the expert RoI. In contrast, **LENS** produces compact subtype-aligned heatmaps that co-localize with glandular/lepidic structures in LUAD and solid growth regions in LSCC, with limited background activation. This multi-slide comparison visually supports the quantitative expert-alignment results in Table 3 and Table J.1.

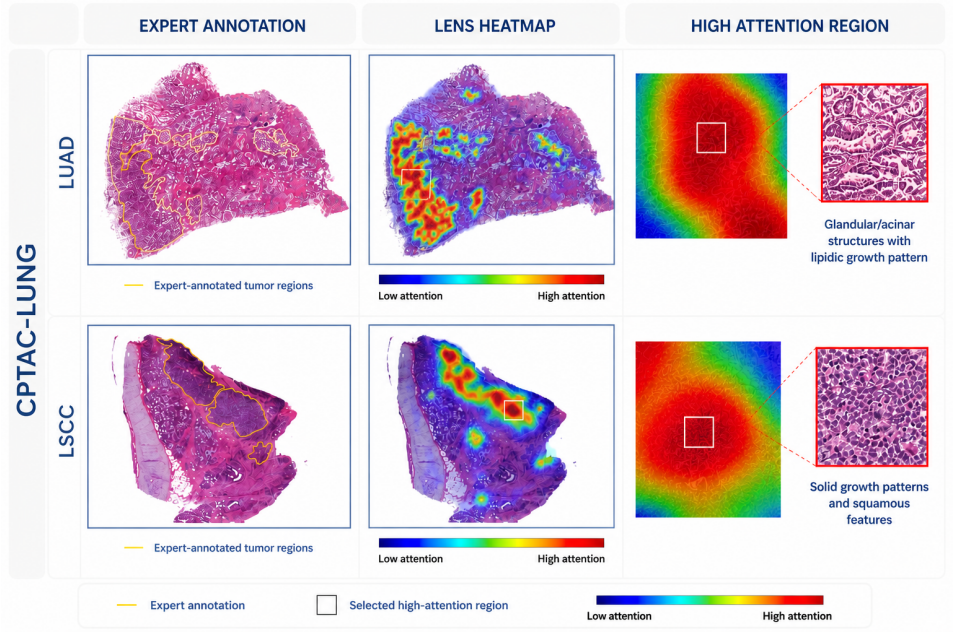

Figure J.4: **CPTAC-Lung topology-derived heatmaps for representative LUAD and LSCC slides.** Each row shows one subtype example: LUAD in the top row and LSCC in the bottom row. Columns show, from left to right: expert-annotated RoIs, the **LENS** topology-derived heatmap, and a zoomed histology patch from the highlighted region. Yellow contours mark expert RoIs, and warmer heatmap colors indicate higher retained edge mass. **LENS** concentrates retained edge mass on glandular/lepidic tumor structures in LUAD and solid squamous growth in LSCC, matching subtype-relevant morphology without region-level supervision. This supports the claim that the learned sparse topology captures pathology-relevant tissue organization from slide-level labels alone.

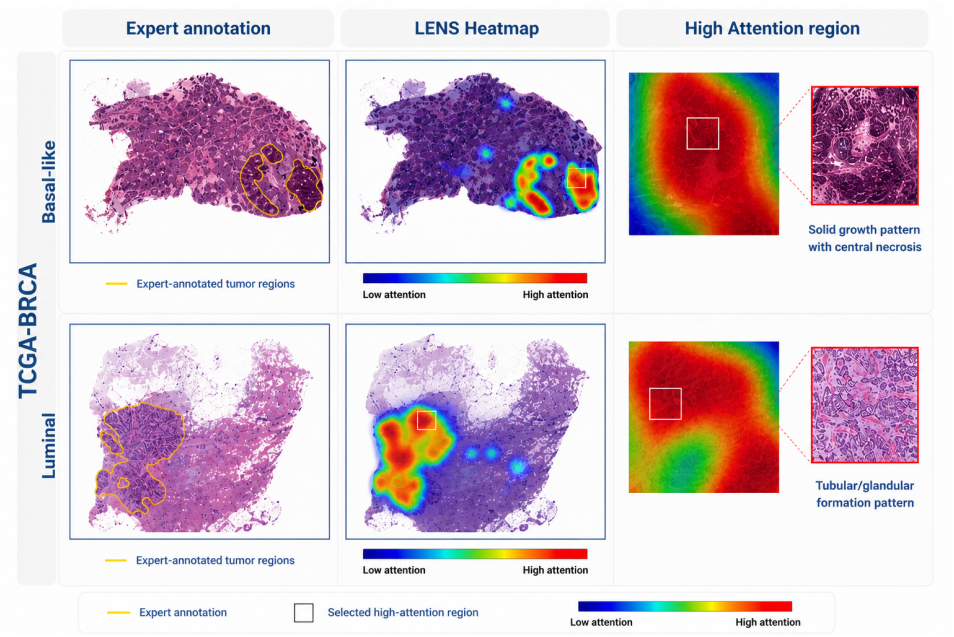

Figure J.5: **TCGA-BRCA topology-derived heatmaps for representative basal-like and luminal slides.** Layout and color coding follow Fig. J.4. **LENS** concentrates retained edge mass within tumor-rich breast tissue regions rather than distributing activation broadly across the slide. Tumor contours were drawn by the same board-certified pathologist, blinded to model outputs, and serve as qualitative reference only; they are not part of the quantitative study. This supports the cross-dataset consistency of the topology-derived explanation outside the CPTAC-Lung expert-alignment cohort.

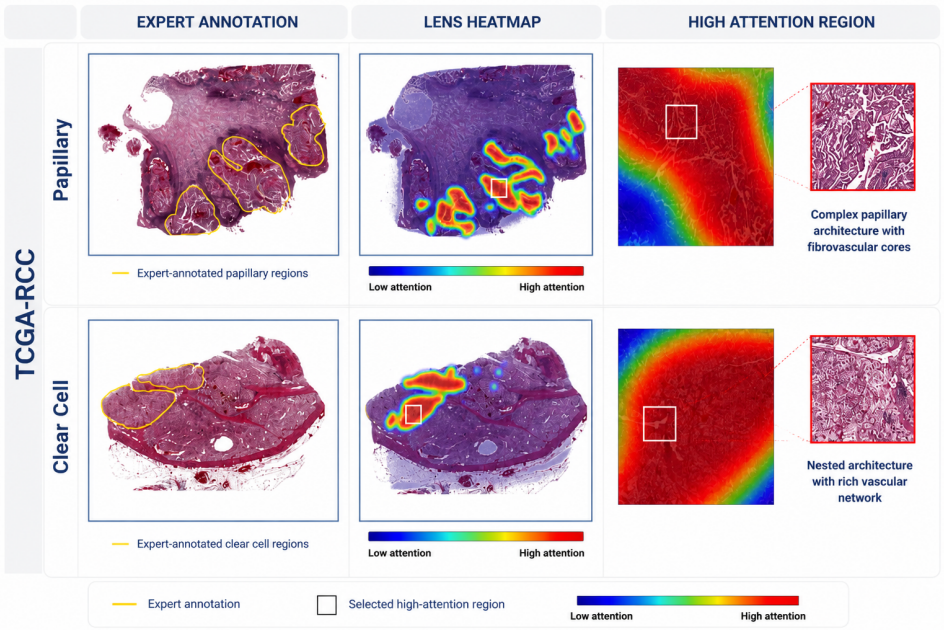

Figure J.6: **TCGA-RCC topology-derived heatmaps for representative papillary and clear-cell slides.** Layout and color coding follow Fig. J.4. **LENS** concentrates retained edge mass in tumor-rich renal tissue regions instead of distributing activation broadly across non-diagnostic tissue. As for TCGA-BRCA, contours were drawn by the same pathologist and are qualitative reference only. The learned sparse graph stays centered on tumor regions on TCGA-RCC, where compartmentalized renal morphology gives strong spatial structure.

K Failure Mode Analysis

Examining misclassified cases through the learned sparse graph reveals three recurring failure modes. They separate three sources of error: morphology that H&E alone cannot resolve, focal evidence outweighed by the global slide architecture, and cases that are ambiguous even for pathologists. Several match known diagnostic limits of H&E and point to where molecular data, hierarchical aggregation, or uncertainty estimates would help. Figure K.1 shows one representative case per mode.

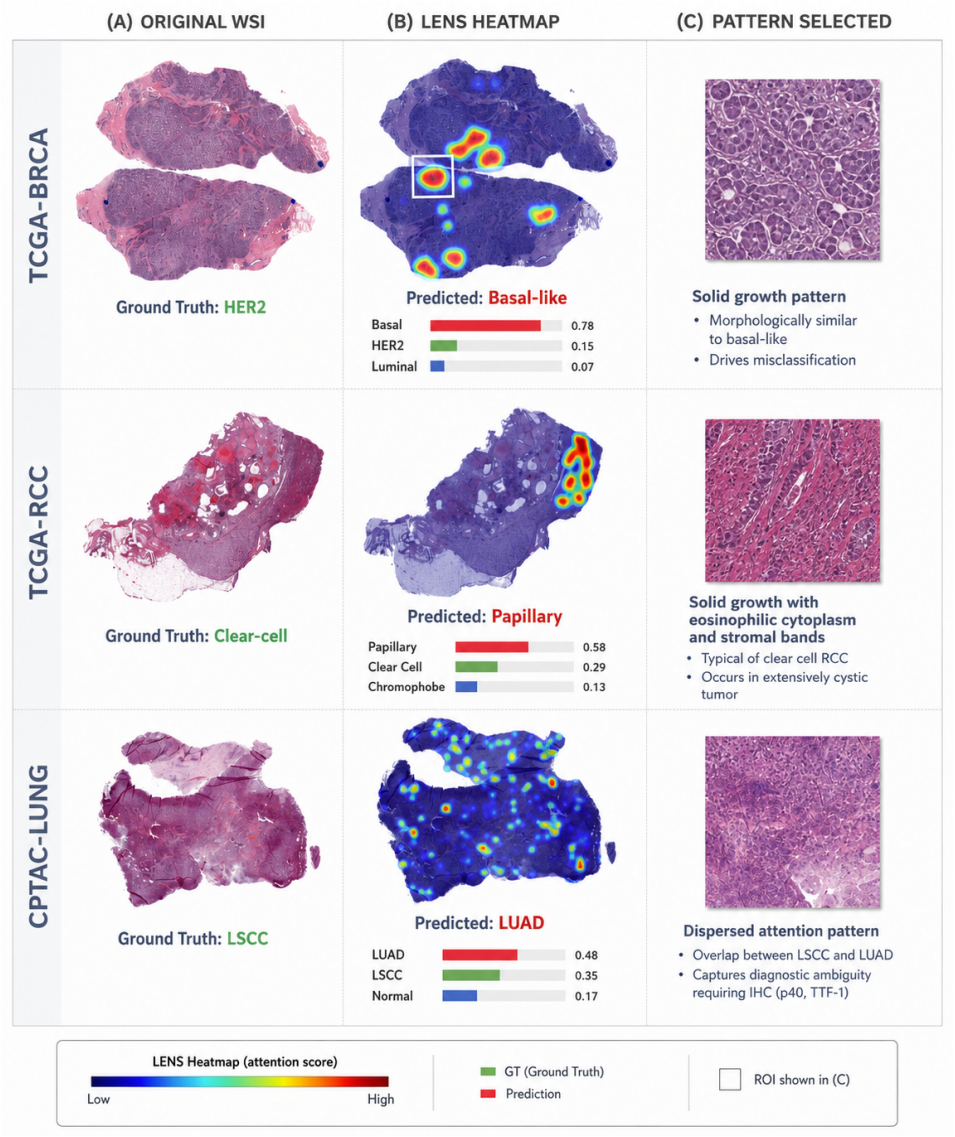

Figure K.1: **Representative failure modes revealed by the learned sparse graph.** Each row shows the WSI, the **LENS** topology-derived heatmap (warmer colors indicate higher retained edge mass), and a zoomed histology region. (A) TCGA-BRCA, HER2-enriched tumor predicted as basal-like: **LENS** localizes the tumor, but H&E morphology alone cannot resolve the molecular subtype. (B) TCGA-RCC, clear-cell RCC predicted as papillary: focal solid tumor receives high edge mass, but more than 60% of the tissue is cystic and dominates graph-level aggregation. (C) CPTAC-Lung, poorly differentiated LSCC predicted as LUAD with low confidence (LUAD 0.48, LSCC 0.35) and diffuse edge mass, reflecting genuine morphologic ambiguity rather than confident focus on a wrong region.

K.1 Molecular subtype not resolvable from H&E (BRCA HER2-enriched)

High-grade solid growth is shared by HER2-enriched, basal-like, and some luminal B tumors. In this case, **LENS** concentrates on the tumor core, but the morphology does not determine the molecular subtype. This matches clinical practice: HER2 status in invasive breast cancer is assessed by IHC or FISH, not by H&E morphology (Wolff et al., 2018).

Practical implication. When a slide shows solid high-grade morphology and two subtype probabilities are close, molecular testing is more informative than a morphology-only prediction. **LENS** can point to the tumor region to sample, but the subtype itself requires IHC, FISH, or genomic data.

K.2 Local evidence overridden by global architecture (RCC cystic change)

LENS assigns high edge mass to small solid tumor regions with clear-cell architecture, yet classifies the slide as papillary RCC. This points to an aggregation failure: when more than 60% of the tissue is cystic, graph-level pooling gives the focal diagnostic regions too little weight relative to the dominant pattern. Pathologists instead prioritize the most diagnostic region, however small.

Practical implication. Edge mass concentrated on a small subregion together with low confidence can flag slides where focal diagnostic tissue is underweighted. Hierarchical pooling that can up-weight small high-quality regions is a natural extension (Pati et al., 2022).

K.3 Intrinsic morphologic ambiguity (poorly differentiated NSCLC)

The prediction is uncertain (LUAD 0.48, LSCC 0.35), and the retained edges are spread across the slide instead of concentrating on one region. The model therefore does not find a strong discriminative pattern, which is consistent with the case: the WHO classification recommends IHC (p40, TTF-1) for poorly differentiated NSCLC because H&E alone is often insufficient (Travis et al., 2015).

Practical implication. Low prediction confidence together with diffuse edge mass can flag slides that need IHC or expert review. Because **LENS** gates are point estimates, a calibrated uncertainty estimate should accompany the topology-derived explanation in future work (see Limitations).

L Algorithms

LENS and **LENS+** share the same sparsification front-end: an edge scorer assigns gate values to the candidate edges and produces a slide-specific sparse weighted adjacency $\tilde{\boldsymbol{A}}$. They differ only in how they use this graph. **LENS** aggregates features once over $\tilde{\boldsymbol{A}}$, applies a single linear layer, and pools patches with softmax weights over their retained edge mass, which adds no learned parameters. **LENS+** applies three residual GCN layers over $\tilde{\boldsymbol{A}}$ and pools with multi-head attention from a learnable `[CLS]` token restricted to active nodes. Both variants use the same learned graph for prediction and for the heatmap.

Training and inference. During training, gates are sampled stochastically from the Hard-Concrete distribution. At inference, they are computed deterministically and frozen, and the resulting sparse adjacency is used both for prediction and for the heatmap.

Algorithms L.1–L.4 give pseudocode for Hard-Concrete gating, edge scoring, the **LENS** loss, and the full training and inference pipeline. Code and trained models are available at https://anonymous.4open.science/r/LENS-014E/.

Algorithm L.1: Hard-Concrete ℓ_0 Gating

Input: Edge logits $\{\log \alpha_{ij}\}_{(i,j)\in\mathcal{E}}$, temperature τ, stretch bounds $[\gamma, \zeta]$
Output: Edge gates $\{z_{ij}\}_{(i,j)\in\mathcal{E}}$

if training **then**
  // Stochastic gating with reparameterization trick
  foreach *edge* $(i,j) \in \mathcal{E}$ **do**
    $u \sim \text{Uniform}(0,1)$
    $g \leftarrow \log u - \log(1-u)$ // Logistic noise
    $s_{ij} \leftarrow \sigma\left((\log \alpha_{ij} + g)/\tau\right)$ // Concrete relaxation
    $\bar{s}_{ij} \leftarrow \gamma + (\zeta - \gamma) s_{ij}$ // Stretch to $[\gamma, \zeta]$
    $z_{ij} \leftarrow \text{clip}_{[0,1]}(\bar{s}_{ij})$ // Hard-Concrete gate
else
  // Deterministic gating at inference (no sampling), with $\tau{=}\tau_{\min}$
  foreach *edge* $(i,j) \in \mathcal{E}$ **do**
    $s_{ij} \leftarrow \sigma\left((\log \alpha_{ij})/\tau_{\min}\right)$ // Deterministic sigmoid
    $\bar{s}_{ij} \leftarrow \gamma + (\zeta - \gamma) s_{ij}$ // Stretch to $[\gamma, \zeta]$
    $z_{ij} \leftarrow \text{clip}_{[0,1]}(\bar{s}_{ij})$ // Frozen graded gate value
return $\{z_{ij}\}_{(i,j)\in\mathcal{E}}$

Algorithm L.2: Edge Scoring Network g_φ

Input: Node features $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, adjacency $\boldsymbol{A} \in \{0,1\}^{n\times n}$
Output: Edge log-alpha parameters $\{\log \alpha_{ij}\}_{(i,j)\in\mathcal{E}}$

$\mathcal{E} \leftarrow \{(i,j) \mid A_{ij} = 1\}$ // Extract candidate edges from 8-NN graph

// Feature Normalization:
$\hat{\boldsymbol{x}}_i \leftarrow \boldsymbol{x}_i / \|\boldsymbol{x}_i\|_2$ for all $i \in \{1, \ldots, n\}$

// Feature-Importance Attention:
foreach *edge* $(i,j) \in \mathcal{E}$ **do**
  $\boldsymbol{a}_{[i,j]} \leftarrow \text{softmax}\left(\text{MLP}_\phi([\hat{\boldsymbol{x}}_i \| \hat{\boldsymbol{x}}_j])\right) \in \mathbb{R}^d$ // Channel-wise attention

// Relation-Aware Edge Feature Construction:
foreach *edge* $(i,j) \in \mathcal{E}$ **do**
  $\Delta_{ij} \leftarrow |\hat{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_j| \odot \boldsymbol{a}_{[i,j]}$ // Attended pairwise contrast
  $\boldsymbol{e}_{ij} \leftarrow [\hat{\boldsymbol{x}}_i \parallel \hat{\boldsymbol{x}}_j \parallel \Delta_{ij}] \in \mathbb{R}^{3d}$ // Edge descriptor
  $\log \alpha_{ij} \leftarrow \text{MLP}_\psi(\boldsymbol{e}_{ij})$ // Edge importance logit
return $\{\log \alpha_{ij}\}_{(i,j)\in\mathcal{E}}$

Algorithm L.3: **LENS** Loss Function

Input: Predictions $\{\hat{\boldsymbol{y}}_b\}_{b \in \mathcal{B}}$, labels $\{\boldsymbol{y}_b\}_{b \in \mathcal{B}}$, edge logits $\{\log \alpha_{b,ij}\}$, Hard-Concrete β and stretch bounds $[\gamma, \zeta]$, target density ρ_e, scheduled weight $\lambda(t)$, density weight λ_ρ, $\alpha_{\min}$=0.2, $\alpha_{\max}$=2.0

Output: Total loss $\mathcal{L}$

// Classification loss:

$\mathcal{L}_{\text{cls}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \ell_{\text{CE}}(\boldsymbol{y}_b, \hat{\boldsymbol{y}}_b)$ *// Cross-entropy*

// Expected ℓ_0 sparsity:

foreach *graph* $b \in \mathcal{B}$ **do**

  $c \leftarrow \beta \log\left(-\frac{\gamma}{\zeta}\right)$ *// Hard-Concrete constant*

  $\pi_{b,ij} \leftarrow \sigma(\log \alpha_{b,ij} - c)$ for all $(i,j) \in \mathcal{E}_b$ *// Keep probabilities, Eq. (3.3)*

  $\bar{\pi}_b \leftarrow \frac{1}{|\mathcal{E}_b|} \sum_{(i,j) \in \mathcal{E}_b} \pi_{b,ij}$ *// Graph-level density*

$\mathcal{L}_{\ell_0} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \bar{\pi}_b$ *// Size-normalized edge count*

// Density deviation:

$\mathcal{L}_\rho \leftarrow \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} (\bar{\pi}_b - \rho_e)^2$ *// Target compliance, Eq. (3.7)*

// Adaptive scaling:

$\bar{\pi}_{\mathcal{B}} \leftarrow \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \bar{\pi}_b$ *// Batch density (one slide per batch)*

$\lambda_{\text{eff}} \leftarrow \lambda(t) \cdot \text{clip}\big(1 + (\bar{\pi}_{\mathcal{B}} - \rho_e),\ \alpha_{\min},\ \alpha_{\max}\big)$ *// Eq. (3.8)*

// Total loss:

$\mathcal{L} \leftarrow \mathcal{L}_{\text{cls}} + \lambda_{\text{eff}} \cdot \mathcal{L}_{\ell_0} + \lambda_\rho \cdot \mathcal{L}_\rho$

return $\mathcal{L}$

Algorithm L.4: LENS: Learnable Edge Network Sparsification for WSI Classification

Input: WSI graphs $\{\mathcal{G}_b = (\boldsymbol{X}_b, \boldsymbol{A}_b)\}_{b=1}^{B}$ (undirected 8-NN), labels $\{y_b\}_{b=1}^{B}$, epochs T, warm-up epochs T_w, ramp epochs T_r, annealing epochs T_{anneal}, target density ρ_e, Hard-Concrete parameters τ_{init}, $\tau_{\min}$, β, $[\gamma, \zeta]$, regularization λ_0, λ_ρ, adaptive bounds $[\alpha_{\min}, \alpha_{\max}]$

Output: Trained backbone θ and edge scorer φ

// Initialization:

Initialize backbone parameters $\theta = \{\theta_{\text{GNN}}, \theta_{\text{pool}}, \theta_{\text{cls}}\}$

Initialize edge scorer parameters $\varphi = \{\phi, \psi\}$ (output bias 1.0)

// Training loop:

for $t = 1, \ldots, T$ **do**

  $\tau \leftarrow \tau_{\min} + (\tau_{\text{init}} - \tau_{\min}) \cdot \frac{1+\cos(\pi \cdot \min(1, \max(0, t-T_w)/T_{\text{anneal}}))}{2}$ *// Cosine annealing*

  $\lambda(t) \leftarrow \begin{cases} 0 & \text{if } t \le T_w \\ \lambda_0 \cdot \min\left(1, (t - T_w)/T_r\right) & \text{otherwise} \end{cases}$ *// Warm-up, then linear ramp*

  foreach *mini-batch* $\mathcal{B}$ *(one slide)* **do**

    foreach *graph* $b \in \mathcal{B}$ **do**

      $\{\log \alpha_{b,ij}\}_{(i,j) \in \mathcal{E}_b} \leftarrow \text{EDGESCORING}_\varphi(\boldsymbol{X}_b, \boldsymbol{A}_b)$ *// Algorithm L.2*

      $\{z_{b,ij}\} \leftarrow \text{HARDCONCRETEGATING}(\{\log \alpha_{b,ij}\}, \tau, [\gamma, \zeta])$ *// Algorithm L.1*

      $\tilde{\boldsymbol{A}}_b \leftarrow \boldsymbol{Z}_b \odot \boldsymbol{A}_b$; $\hat{\boldsymbol{A}}_b \leftarrow \text{row_norm}(\tilde{\boldsymbol{A}}_b + \boldsymbol{I})$ *// Gated, normalized adjacency*

      if ***LENS*** **then**

        $\boldsymbol{H}_b \leftarrow \hat{\boldsymbol{A}}_b \boldsymbol{X}_b \boldsymbol{W}$ *// One aggregation + linear layer*

        $\boldsymbol{h}_{\text{slide},b} \leftarrow \sum_i \text{softmax}_i(w_{b,i})\, \boldsymbol{h}_{b,i}$, $w_{b,i} = \sum_j z_{b,ij}$ *// Edge-mass pooling*

      else *// **LENS+***

        $\boldsymbol{H}_b^{(0)} \leftarrow \text{LN}(\boldsymbol{X}_b \boldsymbol{W}_{\text{in}})$

        for $\ell = 1, 2, 3$ **do**

          $\boldsymbol{H}_b^{(\ell)} \leftarrow \boldsymbol{H}_b^{(\ell-1)} + \text{ReLU}\left(\text{LN}\left(\hat{\boldsymbol{A}}_b \boldsymbol{H}_b^{(\ell-1)} \boldsymbol{W}^{(\ell)}\right)\right)$ *// Residual GCN*

        $\boldsymbol{h}_{\text{slide},b} \leftarrow \text{MHA}_{[\text{CLS}]}(\boldsymbol{H}_b^{(3)};\, m_{b,i} = \mathbb{I}[w_{b,i} > 0])$ *// Masked [CLS] pooling*

      $\hat{y}_b \leftarrow \text{MLP}_{\theta_{\text{cls}}}(\boldsymbol{h}_{\text{slide},b})$ *// Prediction*

    $\mathcal{L} \leftarrow \text{LENSLOSS}(\{\hat{y}_b\}, \{y_b\}, \{\log \alpha_{b,ij}\}, \beta, [\gamma, \zeta], \rho_e, \lambda(t), \lambda_\rho, [\alpha_{\min}, \alpha_{\max}])$ *// Algorithm L.3*

    Update θ, φ with AdamW using $\nabla_{\theta,\varphi} \mathcal{L}$

  Keep the checkpoint with the best validation accuracy

// Inference on an unseen slide:

$\{\log \alpha_{ij}\} \leftarrow \text{EDGESCORING}_\varphi(\boldsymbol{X}, \boldsymbol{A})$ on both orientations of every edge

$z_{ij}^{\text{inf}} \leftarrow \text{HARDCONCRETEGATING}$ in deterministic mode ($\tau = \tau_{\min}$)

$z_{ij}^{\text{inf}} \leftarrow z_{ji}^{\text{inf}} \leftarrow \frac{1}{2}(z_{ij}^{\text{inf}} + z_{ji}^{\text{inf}})$ *// Undirected symmetrization*

Predict with the frozen sparse adjacency $\tilde{\boldsymbol{A}}^{\text{inf}} = \boldsymbol{Z}^{\text{inf}} \odot \boldsymbol{A}$; heatmap $w_i = \sum_j z_{ij}^{\text{inf}}$

return θ, φ

M Limitations and Future Work

We group the main limitations by design axis: the patch representation, the candidate graph, the supervision signal, and the reliability of the explanations.

1) Encoder spectrum and topology stability: The current evaluation spans **SimCLR**, **CTransPath**, and **UNI2-h**, with stable 28–29% retention and largely preserved rank ordering across encoders. Encoders with different inductive biases, such as **Virchow** (Vorontsov et al., 2024) or **CONCH** (Lu et al., 2024), use slide-level or language-aligned pretraining that may change what patch-pair differences encode, and thus the learned topology.

2) Single magnification: The current study uses $20\times$ patches, which capture tissue architecture but miss nuclear detail at $40\times$ and broader growth patterns at $5\times$. Hierarchical multi-scale graphs with cross-resolution sparsification could model both while keeping edge-level interpretability (Guo et al., 2023; Shi et al., 2023).

3) Limited survival evidence: The survival extension (Appendix N) covers two cohorts. **LENS+** gives the highest C-index on TCGA-BRCA, but on TCGA-BLCA all methods stay close to chance. Larger prognostic cohorts are needed to judge how well task-supervised topology transfers to censored time-to-event supervision.

4) Interpretability evaluation scale: The expert-alignment study uses one board-certified pathologist and ten CPTAC-Lung slides, so it shows structural alignment, not clinical validity. Multi-reader, multi-institution studies are needed to measure inter-reader variability, robustness to site, scanner, and stain differences, and clinical utility. Such a clinical validation study is beyond the scope of this work, which focuses on the method and its structural evaluation.

5) Calibrated uncertainty: Edge gates and heatmaps are deterministic point estimates without calibrated uncertainty. Predictive and gate-level uncertainty would help separate genuinely ambiguous morphology from confidently learned structure (see the failure cases in Appendix K).

6) Multimodal integration: Sparse WSI graphs could combine morphology with genomic, clinical, or IHC data through cross-modal edge gating, testing whether topology learning still yields interpretable structure in multimodal prediction (Hemker et al., 2024).

7) Progressive graph construction: The model sparsifies a fixed 8-NN candidate graph. Updating the candidate edges during training could improve topology quality, provided density control is preserved (Chen et al., 2020).

Together, these directions position learned sparse topology as a framework for controllable, interpretable WSI modeling across encoders, scales, objectives, and modalities.

N Extension to Survival Prediction

Patch-GCN and **GraphLSurv** were built for survival prediction, and in the main evaluation we adapted them to classification. This section removes that asymmetry. We move **LENS** to survival and compare it with both baselines in their native survival form, so the baselines now run on the task they were designed for. The question is whether task-supervised sparsification still works when the signal comes from censored time-to-event labels instead of class labels.

Cohorts. We use two TCGA cohorts: Urothelial Bladder Carcinoma (BLCA, n=436) and Breast Invasive Carcinoma (BRCA, n=1,021).

Adapting LENS. The architecture is unchanged; only the head and the loss change. We discretize follow-up time into K intervals $\{I_1, \ldots, I_K\}$. For slide g the network outputs a per-interval hazard

$$h_k(g) = \sigma\big(\boldsymbol{w}_k^\top \boldsymbol{h}_{\text{slide}}(g) + b_k\big), \qquad k = 1, \ldots, K, \tag{N.1}$$

where $\boldsymbol{h}_{\text{slide}}(g)$ is the pooled sparse-graph embedding and σ is the sigmoid. The survival function follows as $S_k(g) = \prod_{j \le k}\big(1 - h_j(g)\big)$. With event or censoring time in interval k_i and event indicator

δ_i (δ_i=1 for an observed event), the loss is the discrete-time negative log-likelihood

$$\mathcal{L}_{\text{surv}} = -\sum_i \left[\delta_i \Big(\log h_{k_i}(g_i) + \sum_{j<k_i} \log\big(1 - h_j(g_i)\big) \Big) + (1-\delta_i) \sum_{j \le k_i} \log\big(1 - h_j(g_i)\big) \right]. \quad \text{(N.2)}$$

An observed event contributes its hazard in interval k_i and survival through the earlier intervals. A censored slide contributes survival through interval k_i. This is a per-slide loss. It needs no risk set, so it stays compatible with the batch-size-one, variable-size graph regime. The full objective keeps the **LENS** gating and density terms unchanged:

$$\mathcal{L} = \mathcal{L}_{\text{surv}} + \lambda_{\text{eff}} \cdot \mathcal{L}_{\ell_0} + \lambda_\rho \cdot \mathcal{L}_\rho. \quad \text{(N.3)}$$

All other hyperparameters, including ρ_e=0.30, are the same as for classification.

Table N.1: **Survival prediction on TCGA-BRCA and TCGA-BLCA (UNI2-h features).** C-index (mean±std over 5 patient-stratified folds) and edge retention. All graph methods use the same 8-NN candidate graph, and **Patch-GCN** and **GraphLSurv** run in their native survival form. Bold marks the best result and underline the second best. On BRCA, **LENS+** gives the highest mean C-index and **LENS** the smallest std. On BLCA, all methods stay close to chance with overlapping intervals.

Method	BRCA C-index ↑	BLCA C-index ↑	Edge Ret. (%) ↓
Dense Graph Neural Networks			
Patch-GCN	$0.539_{\pm 0.045}$	$\mathbf{0.564_{\pm 0.102}}$	100
Sparse Graph Neural Networks			
GraphLSurv	$0.636_{\pm 0.066}$	$0.525_{\pm 0.097}$	∼45
LENS	$\underline{0.639_{\pm 0.012}}$	$0.536_{\pm 0.040}$	∼30
LENS+	$\mathbf{0.653_{\pm 0.018}}$	$\underline{0.542_{\pm 0.081}}$	∼29

Three observations follow from Table N.1.

1) Stability across folds. On BRCA, the std of **LENS** is 4–5× smaller than that of either baseline (0.012 against 0.045 and 0.066). On BLCA, it is about 2.5× smaller (0.040 against 0.102 and 0.097). Censored labels are a noisy training signal. A possible explanation is that the density controller keeps each slide near the same operating point, so the read-out sees a similar amount of context in every fold. With five folds this is an observation, not a proof, but it suggests that sparsity can also reduce variance, not only compute.

2) Same accuracy with a sparser graph. **LENS** matches **GraphLSurv** in mean C-index on BRCA (0.639 against 0.636) while keeping ∼30% of edges instead of ∼45%. The difference comes from how density is set. **GraphLSurv** learns a projection but keeps a fixed threshold on a bounded cosine similarity, so its final density depends on how the features are distributed. **LENS** scores each edge with an unbounded logit $\log \alpha_{ij}$ and steers the kept fraction toward a chosen target ρ_e. It therefore reaches a chosen operating point whatever the feature scale.

3) BLCA stays close to chance. All methods reach a mean C-index between 0.525 and 0.564, with fold std up to 0.10, so no method is reliably better than the others. The nominal best, **Patch-GCN**, keeps all edges. A likely reason is that the BLCA survival signal is too weak for the censored gradient to guide edge selection. In that regime a task-supervised sparsifier has little to select from, which marks a limit of the approach rather than a failure specific to **LENS**.