BeamWitness: Rooted Subgraph Beam Search for Selective Graph Representation Learning

Lin Du[1], **Lu Bai**[1]*, **Lixin Cui**[2], **Ming Li**[3], **Hangyuan Du**[4], **Bo Jiang**[5]
[1]School of Artificial Intelligence, Beijing Normal University, Beijing, China.
[2]School of Information, Central University of Finance and Economics, Beijing, China.
[3]Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua, China.
[4]School of Computer and Information Technology, Shanxi University, Taiyuan, China.
[5]School of Computer Science and Technology, Anhui University, Hefei, China.
dulin@mail.bnu.edu.cn, bailu@bnu.edu.cn

Abstract

Standard graph neural networks (GNNs) are built on message passing and neighborhood aggregation, following the Weisfeiler–Lehman refinement paradigm. Despite their wide adoption, these models tend to compress and mix neighbor information from the very first layer. This early mixing obscures the critical subgraphs that drive the prediction. Attention-based and subgraph-based methods partially address this by reweighting nodes or enriching local representations, but both families remain passive as they usually rely on soft weighting or predefined subgraph templates. We propose BeamWitness, a selective graph representation learning framework that treats subgraph identification as an active search problem. Starting from a root node, a policy trained with task rewards expands a connected subgraph one node at a time or terminates early, and beam search retains a small set of candidate subgraphs as witnesses for prediction. For node-level tasks, the target node serves as the root, while for graph-level tasks, BeamWitness launches searches from multiple root nodes and aggregates the selected witnesses into a graph-level representation. On controlled motif benchmarks with known ground-truth subgraphs, BeamWitness achieves near-perfect accuracy and selects witnesses that align with the true motifs. On real-world node and graph classification benchmarks, BeamWitness is competitive with standard GNNs and related methods while providing an explicit, interpretable subgraph-based account of each prediction.

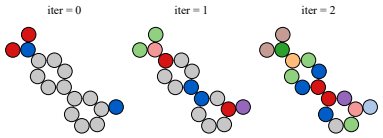
1 Introduction

Graph neural networks (GNNs) have become the standard tool for learning on graph-structured data, powering applications ranging from molecular analysis and biological interaction prediction to knowledge graph reasoning [10, 19, 20]. The dominant design principle is message passing, where each node iteratively aggregates feature vectors from its neighbors to produce a representation encoding its local neighborhood context [8]. This principle traces its theoretical roots to the Weisfeiler–Lehman (WL) graph isomorphism test [12], which refines and recodes node labels in discrete rounds.

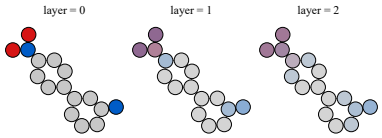
Such message-passing neural networks (MPNNs) follow the computational spirit of WL but resolve neighborhood information in a different way. WL retains the full multiset of neighbor colors at each iteration, yet some of this information may be redundant for the specific task. MPNNs instead collapse that multiset into a single continuous vector through a fixed aggregation function applied

*Corresponding Author: Lu Bai (bailu@bnu.edu.cn)

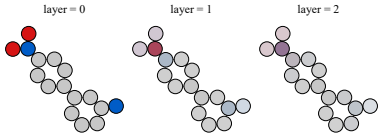
40th Conference on Neural Information Processing Systems (NeurIPS 2026).



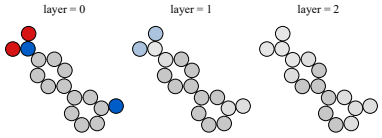
(a) WL: refine and recolor



(b) GCN-mean: averaging-driven diffusion



(c) GIN-sum: count-amplified mixing



(d) GraphSAGE-max: winner-take-all propagation

Figure 1: **Motivation.** The example is a molecule from MUTAG, where the NO_2 group at the upper left and the nitrogen atom at the lower right are chemically informative for mutagenicity, but their signals are mixed with the rest of the molecule as soon as the aggregation begins. Node colors visualize WL recoloring in (a) and continuous aggregation via RGB mixing in (b)–(d).

unconditionally to all neighbors. While repeated aggregation in deep GNNs is widely known to cause over-smoothing [13, 17], a more immediate issue arises already at the first layer, before the model has any task-specific evidence about which neighbors matter. Features from structurally irrelevant regions mix with the decisive local substructure, blurring the signal that should drive the prediction. This behavior is shared across common aggregations such as mean, sum, and max, as shown in Figure 1.

These issues stem from the fixed neighborhood expansion strategy of standard GNNs. Figure 1 illustrates WL and three representative aggregation schemes. WL [12] preserves neighborhood information through discrete relabeling, which is informative but treats every neighbor as equally relevant and is not differentiable. GCN-mean [10] averages all neighbor features at each layer, driving a diffusion that progressively washes out local distinctions. GIN-sum [20] counts and accumulates neighbor contributions, making the representation sensitive to neighbor frequency. GraphSAGE-max [9] lets the strongest activation win through element-wise max pooling while discarding the rest of the neighborhood signal. In each case the model has no mechanism to focus on the subgraph most relevant for a specific task, which motivates the design of selective approaches.

Recent work has explored two main strategies toward local selectivity. Attention-based methods such as GAT [19] and GATv2 [2] reweight neighbors during aggregation, and SAGPool [11] retains top-ranked nodes via self-attention scores. Subgraph-based methods such as NGNN [22], ESAN [1], and SUN [6] enrich representations using local subgraphs obtained from fixed extraction rules or predefined subgraph policies. However, both strategies remain passive, as neither actively searches for the connected subgraph that is most predictive for a given instance.

We propose **BeamWitness**, a selective graph representation learning framework that reframes local evidence construction as an active rooted subgraph search problem. Instead of aggregating a fixed neighborhood, the model starts from a designated root node and incrementally builds a compact witness subgraph through a policy trained from downstream supervision. Because a single greedy trajectory can miss important regions, the model retains multiple parallel candidates through beam search and uses them jointly for the final prediction. The same framework supports both node-level and graph-level representation, while keeping the underlying search procedure unchanged. The main **contributions** of this work are summarized as follows.

- We formulate selective graph representation learning as rooted connected subgraph search and propose BeamWitness, which learns an active search policy and uses beam search to select multiple predictive witnesses. This replaces fixed-neighborhood aggregation with an explicit subgraph selection process whose output is directly inspectable.
- We instantiate the same search mechanism for both node and graph classification. Node classification uses target-rooted search, while graph classification uses multi-root search and witness aggregation. This unified instantiation avoids designing task-specific architectures.
- We introduce a witness evaluation protocol that combines structural alignment with planted motifs, faithfulness through counterfactual and sufficiency interventions, and the Effective Node Ratio as a

compactness statistic. The protocol complements standard accuracy by assessing whether selected witnesses are structurally meaningful, faithful to the prediction, and compact on real-world graphs.

- We show that BeamWitness achieves high accuracy on controlled motif benchmarks while selecting compact witnesses aligned with planted motifs. We further show that witness-based representations are competitive with standard GNNs and related methods on real-world node and graph classification benchmarks while exposing explicit subgraph evidence for each prediction.

2 Related work

We review four lines of work most related to BeamWitness, including standard message passing GNNs, attention-based graph learning, subgraph-based graph learning, and graph explanation methods.

Standard graph neural networks. By standard GNNs, we refer to message passing models that follow the WL refinement paradigm, such as GCN, GraphSAGE, and GIN discussed in the introduction [10, 9, 20]. While these models adopt different aggregation operators such as mean, max, or sum, they share a common rule that a node representation is updated by aggregating features from a prescribed local neighborhood. The design is efficient but does not explicitly decide which part of the neighborhood is relevant for a given prediction. This limitation has motivated subsequent work on selective graph representation learning, which we discuss below.

Attention-based graph learning. One line of work pursues this selectivity through attention. Representative examples include GAT, which learns edge-dependent attention weights over neighbors [19], GATv2, which introduces dynamic attention for higher expressiveness [2], and SAGPool, which scores nodes and retains a top-ranked subset for graph classification [11]. These methods still rely on soft neighbor weighting or graph coarsening, which raises the question of whether more explicit and structured forms of graph representation can be obtained through subgraphs.

Subgraph-based graph learning. A second line of work improves graph representation by encoding predefined collections of subgraphs. NGNN lifts each node to a fixed-radius rooted subgraph and encodes the resulting collection with shared base GNNs [22]. ESAN represents a graph as a bag of subgraphs generated by predefined policies and processes the bag with equivariant aggregation [1]. SUN studies node-based subgraph policies and their symmetry structure to unify and extend prior subgraph GNNs [6]. LGAN constructs line graphs from node-centered induced subgraphs to perform higher-order aggregation [5]. These methods show that subgraph representations can improve expressiveness, but the subgraphs are specified by fixed policies such as ego networks, node deletion, edge deletion, or node marking. BeamWitness differs by learning a rooted expansion policy from downstream supervision and actively constructing compact connected subgraphs for each instance.

Graph explanation methods. Graph explanation methods such as GNNExplainer [21] and PG-Explainer [14] identify substructures that support the prediction of an *already trained* predictor. They are post-hoc by design, taking the predictor as fixed. BeamWitness addresses an orthogonal goal: it makes subgraph selection part of the predictive computation itself. We therefore do not benchmark against post-hoc explainers in our classification tables, since the two settings answer different questions; a comparison under a unified protocol is left for future work.

3 Method

We introduce BeamWitness from the search space to the final predictor in this section. Section 3.1 defines rooted witnesses and the task setup. Section 3.2 describes how a connected witness is grown from a root. Section 3.3 gives the learned policy used during search. Section 3.4 explains how selected witnesses are encoded for prediction. Section 3.5 describes beam search inference.

3.1 Problem setup and witness definition

Let $G = (V, E)$ be a graph with node features $X \in \mathbb{R}^{|V| \times d}$. For a subset $S \subseteq V$, we write X_S for the corresponding rows of X. We use $r \in V$ to denote a designated *root node* from which witness construction starts. The role of r depends on the task. For node classification, r is the target node whose label y_r is to be predicted. For graph classification, the prediction target is the graph label y_G, and r is one of several search anchors selected from V.

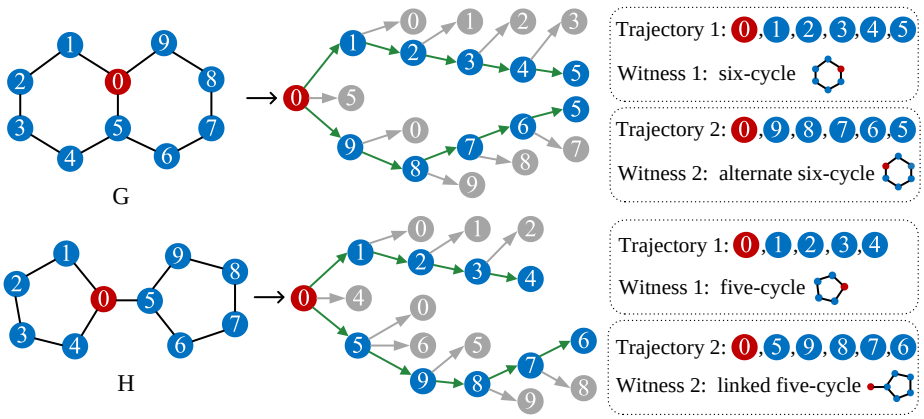


Figure 2: Overview of **BeamWitness**. Starting from a designated root node shown in red, the model grows connected candidate subgraphs and keeps the top terminal witnesses by beam search. Green arrows mark retained transitions and gray arrows mark pruned transitions. This molecule-inspired example shows that Trajectory 1 captures the key cycle evidence in each graph. Trajectory 2 keeps a plausible alternative instead of making a single greedy choice. For illustration we use two witnesses.

A *witness* for a root r is a connected node set $S \subseteq V$ with $r \in S$, and we identify it with the induced subgraph $G[S]$. We constrain its size by two prescribed bounds $m_{\min}$ and $m_{\max}$,

$$m_{\min} \leq |S| \leq m_{\max}, \tag{1}$$

which are fixed before training and control the size of the selected subgraphs. The goal is to learn a predictor that selects one or more witnesses and classifies from those witnesses alone, so that the selected region is compact, instance-dependent, and directly inspectable.

3.2 Rooted witness search

The search process constructs a witness by growing a connected set from the root. Figure 2 illustrates the procedure on two molecule-inspired graphs with different root-centered ring evidence. BeamWitness separates the two cases by selecting compact connected witnesses, such as a root-containing six-cycle in G and a root-containing five-cycle in H. The search state at step t is a rooted connected set S_t, initialized as $S_0 = \{r\}$. The *frontier* of S_t, denoted $\mathcal{F}(S_t)$, consists of all nodes adjacent to but not yet included in the current state,

$$\mathcal{F}(S_t) \;=\; \{u \in V \setminus S_t \mid \exists v \in S_t,\ (u, v) \in E\}. \tag{2}$$

At each step, the policy can either stop or add one frontier node. The available action set is

$$\mathcal{A}(S_t) = \{\text{stop} \mid |S_t| \geq m_{\min}\} \cup \{\text{add}(u) \mid u \in \mathcal{F}(S_t),\ |S_t| < m_{\max}\}. \tag{3}$$

For an add action, the next state is $S_{t+1} = S_t \cup \{u\}$. For a stop action, S_t is recorded as a terminal witness. We refer to the sequence of states $\tau = (S_0, S_1, \ldots, S_T)$ produced by repeated action selection as a *trajectory*, where T is the stopping time and S_T is the terminal witness. This construction guarantees that every witness is connected and contains the root. A trajectory terminates either by selecting stop or by reaching the maximum size $|S_T| = m_{\max}$. For node classification, the target node is used as the root. For graph classification, the primitive is launched from multiple selected roots, and the resulting witnesses are aggregated into a graph-level representation.

3.3 Search policy

The policy network drives the rooted witness search defined in Section 3.2. At each step t it observes the current witness state S_t and produces a distribution over the action set $\mathcal{A}(S_t)$ from Eq. (3).

State representation. We summarize the current state by a fixed-length root-centered embedding

$$\phi_r(S_t) \;=\; \text{EncState}\big(G[S_t],\, X_{S_t},\, r\big) \;\in\; \mathbb{R}^{d_\phi}. \tag{4}$$

The state representation uses a breadth-first ordering rooted at r, with node-indexed components padded to length $m_{\max}$. Standard structural and feature statistics on the induced subgraph are

concatenated into $\phi_r(S_t)$. We keep d_ϕ independent of the input feature dimension by applying a fixed projection to the raw node features when the latter are high-dimensional.

Action scoring. A small feed-forward network maps $\phi_r(S_t)$ to a hidden state $q_\theta(S_t)$. Then two linear heads produce scalar scores for the two action types,

$$s_{\text{stop}}(S_t) \;=\; w_{\text{stop}}^\top q_\theta(S_t), \qquad s_{\text{add}}(S_t, u) \;=\; w_{\text{add}}^\top q_\theta\big(S_t \cup \{u\}\big). \tag{5}$$

For an add action, the policy scores the resulting state $S_t \cup \{u\}$ rather than scoring the candidate node u in isolation. After masking invalid actions according to $\mathcal{A}(S_t)$, the action distribution is

$$p_\theta(a \mid S_t) \;=\; \text{softmax}\big(\{s_{\text{stop}}(S_t)\} \cup \{s_{\text{add}}(S_t, u)\}_{u \in \mathcal{F}(S_t)}\big). \tag{6}$$

During training, we sample $N = B_{\text{train}}$ trajectories $\{\tau_i\}_{i=1}^N$ from p_θ and train the discrete policy with a score-function estimator. If S_i is the terminal witness of τ_i and $z_i = g_\omega(f_\psi(G[S_i], X_{S_i}, r))$, its detached reward is $R_i = -\text{CE}(y, \text{softmax}(z_i)) - \alpha_{\text{size}}|S_i|/m_{\max}$. With an EMA baseline b, the policy loss is $\mathcal{L}_{\text{pol}} = -N^{-1}\sum_i \text{sg}(R_i - b)\sum_t \log p_\theta(a_{i,t} \mid S_{i,t})$. The full training loss is $\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda_{\text{wit}}\mathcal{L}_{\text{wit}} + \lambda_{\text{pol}}\mathcal{L}_{\text{pol}} + \lambda_{\text{ent}}\mathcal{L}_{\text{ent}}$, where the size penalty is incorporated into the policy reward and $\mathcal{L}_{\text{ent}}$ is a negative-entropy regularizer. Details are in Appendix A.5. During inference, policy or predictor scores are used by the beam search of Section 3.5.

3.4 Witness encoder and prediction

The policy alone produces compact node sets. Predictions are made by a separate witness encoder f_ψ that operates on the induced witness subgraph. Keeping the encoder distinct from the policy is central to BeamWitness. The policy and the encoder are connected only through the selected node set, so the choice of subgraph encoder is independent of the policy parameterization.

Per-witness encoding. Given a terminal witness S_i produced by the search, the encoder computes

$$h_i \;=\; f_\psi\big(G[S_i],\, X_{S_i},\, r\big) \;\in\; \mathbb{R}^{d_h}. \tag{7}$$

Here r denotes the root associated with the witness, identified in the witness subgraph through a binary marker. For node classification this is the target node. For graph classification it is the search anchor that launched the witness. Any standard GNN backbone can play the role of f_ψ.

Aggregation across witnesses. For N terminal witnesses produced for the instance, where $N = B_{\text{train}}$ during training and $N = B_{\text{test}}$ at inference, BeamWitness pools the witness embeddings,

$$h \;=\; \rho(h_1, \ldots, h_N), \tag{8}$$

where ρ is any order-invariant pooling such as mean, sum, max, or log-sum-exp. We treat ρ as a hyperparameter and select it on validation. A two-layer classifier g_ω produces the final prediction,

$$\text{Node classification} \quad p(y_r \mid G, r) \;=\; \text{softmax}\big(g_\omega(h)\big), \tag{9}$$

$$\text{Graph classification} \quad p(y_G \mid G) \;=\; \text{softmax}\big(g_\omega(h_G)\big), \tag{10}$$

where h_G is obtained by aggregating embeddings of the final witnesses selected across multiple roots.

3.5 Beam search inference

At inference time, BeamWitness replaces stochastic trajectory sampling with deterministic beam search [7, 15]. For each root $r \in \mathcal{R}$ the algorithm maintains a local active beam of at most B candidate states. At each step every active state is expanded by the stop action and by every valid frontier addition, producing a pool of successor states. Each successor is scored by a function σ, the top-B unfinished successors form the next active beam, and successors that take stop or reach $m_{\max}$ are recorded into a per-root completed pool. Once all roots have finished, the final B_{test} witnesses are selected globally across all per-root pools by the same score σ. Node classification is the special case $\mathcal{R}{=}\{r\}$, in which the per-root and global pools coincide. The core steps are summarized above, and the full procedure is given in Algorithm 1 in Appendix A.3.

We consider three choices for σ. The simplest is a policy score derived from the cumulative log-probability of the trajectory so far. A confidence score evaluates the candidate witness through f_ψ and g_ω and ranks it by the predicted maximum class probability. A margin variant uses the gap between the top-two probabilities. Confidence-based scoring is the default in our real-world experiments because it directly reflects downstream predictive utility.

4 Evaluation protocol

We report three families of metrics. The first is predictive performance, reported as classification accuracy. The second is structural alignment with planted motifs, which applies only when ground truth is available. The third covers faithfulness through interventions on the selected witness and compactness through the effective node ratio. Several of our metrics are defined as a gap against a size-matched random rooted baseline. We fix that baseline first and then reuse it across the gap metrics. The effective node ratio is reported directly as a fraction of the input graph order.

4.1 Predictive performance and the random rooted baseline

We report classification accuracy for performance evaluation. For an instance with root r and selected witness S of size $|S|$, we draw a size-matched random rooted subgraph S_{rand} by starting from r and adding uniformly random frontier nodes until $|S_{\mathrm{rand}}| = |S|$. All baseline statistics are averaged over 20 such draws per instance. Δ is defined as the value on S minus the value on S_{rand}.

4.2 Structural alignment with planted motifs

For datasets with planted ground-truth motif $M^\star$ we report the intersection-over-union between the witness S and $M^\star$, and the gap of this score against the random rooted baseline,

$$\mathrm{IoU}(S, M^\star) \;=\; \frac{|S \cap M^\star|}{|S \cup M^\star|}, \qquad \Delta\mathrm{IoU} \;=\; \mathrm{IoU}(S, M^\star) - \mathrm{IoU}(S_{\mathrm{rand}}, M^\star). \tag{11}$$

Because the witness size budget $[m_{\min}, m_{\max}]$ is comparable to the motif size, $\mathrm{IoU}(S_{\mathrm{rand}}, M^\star)$ is already large in absolute terms (see Appendix A.9). The relevant signal is therefore ΔIoU rather than the raw IoU. This metric applies only to controlled benchmarks, where the ground-truth motif is known by construction.

4.3 Faithfulness and compactness

Faithfulness asks whether the prediction genuinely depends on the selected witness. Compactness asks how compact the witness is relative to the input graph. We define the four metrics below.

Counterfactual deletion. Let $\hat{y}$ be the predicted class on the original graph and let $p(y \mid \cdot)$ denote the model probability assigned to class y. The counterfactual score for a witness S is the drop in $p(\hat{y} \mid \cdot)$ when the non-root witness nodes are deleted from the graph,

$$\mathrm{CF}(S) \;=\; p(\hat{y} \mid G) \;-\; p\big(\hat{y} \mid G \setminus (S \setminus \{r\})\big), \qquad \Delta_{\mathrm{CF}} \;=\; \mathrm{CF}(S) - \mathrm{CF}(S_{\mathrm{rand}}). \tag{12}$$

A large positive Δ_{CF} means that deleting the selected non-root witness nodes causes a larger confidence drop than deleting non-root nodes from a size-matched random rooted witness. Figure 3 illustrates this contrast on the three controlled benchmarks.

Sufficiency. Sufficiency measures the residual confidence when the model sees *only* the witness as input. Re-encoding the induced subgraph $G[S]$ with features X_S produces a probability of class $\hat{y}$,

$$\mathrm{Suff}(S) \;=\; p(\hat{y} \mid G[S], X_S), \qquad \Delta_{\mathrm{Suff}} \;=\; \mathrm{Suff}(S) - \mathrm{Suff}(S_{\mathrm{rand}}). \tag{13}$$

Root-masked sufficiency On real-world node datasets the root raw features may dominate $p(\hat{y} \mid \cdot)$. This makes Δ_{CF} a low-amplitude signal because deleting a few peripheral non-root nodes only slightly perturbs a confident root-driven prediction. We therefore additionally report *root-masked sufficiency*, in which the root raw features are zeroed inside the witness before re-encoding while the binary root marker is preserved. We write $\widetilde{X}_S$ for the row-masked features, with $\widetilde{X}_r = 0$ and $\widetilde{X}_u = X_u$ for $u \neq r$, so that

$$\mathrm{Suff}^{\neg r}(S) \;=\; p\Big(\hat{y} \mid G[S], \widetilde{X}_S\Big), \qquad \Delta^{\neg r}_{\mathrm{Suff}} \;=\; \mathrm{Suff}^{\neg r}(S) - \mathrm{Suff}^{\neg r}(S_{\mathrm{rand}}). \tag{14}$$

This isolates whether the non-root witness nodes carry label signal beyond the target-node feature alone, and is our primary diagnostic for real-world node classification.

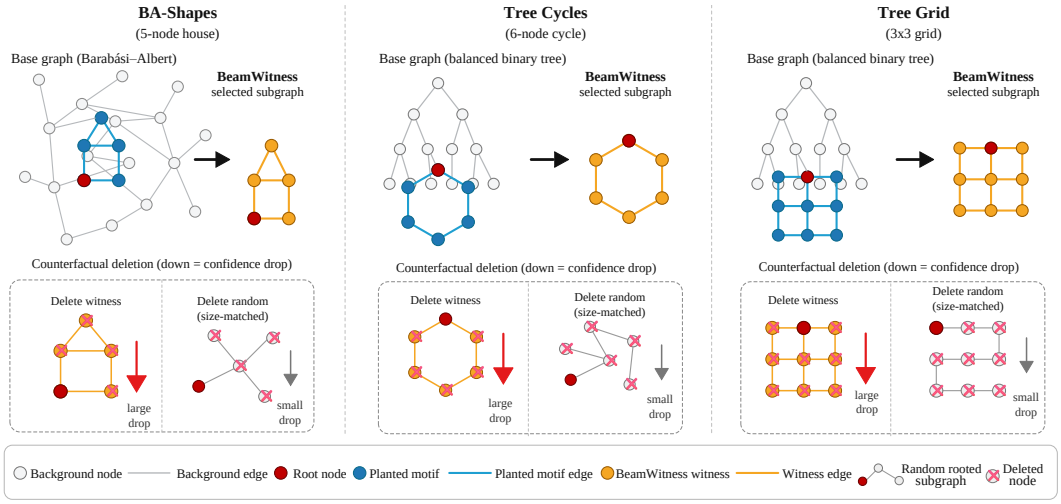


Figure 3: **Visualizing the witness and Δ_{CF} on the three controlled motif benchmarks.** In the top row of each panel, the base graph appears on the left with the planted motif highlighted in blue, and the connected witness BeamWitness selects from the root r shown in red appears on the right. The two subgraphs agree on the role-defining region, which is a five-node house, a six-node cycle, and a 3×3 grid respectively. In the bottom row, removing the non-root witness nodes (left, red arrow) causes a large confidence drop, while removing non-root nodes of a size-matched random rooted subgraph (right, gray arrow) causes a much smaller confidence drop. The vertical gap between the two drops is Δ_{CF} in Eq. (12). The corresponding numerical values are reported in Table 1.

Effective node ratio. To summarize the size of the witness relative to the input graph, we define

$$\eta(S) \;=\; \frac{|S|}{\bar{N}}, \qquad \bar{N} \;=\; \frac{1}{|\mathcal{D}|}\sum_{G\in\mathcal{D}} |V(G)|, \tag{15}$$

where $\bar{N}$ is the average graph order in the dataset $\mathcal{D}$. A small η at preserved predictive performance indicates that the prediction relies on a tiny instance-dependent portion of the graph rather than on unconditional neighborhood mixing. On controlled motif benchmarks, the size bounds are chosen in relation to the planted motif sizes, constraining the variation in η. We therefore emphasize structural alignment and faithfulness on these benchmarks and report η on real-world data; see Appendix A.2.

5 Experiments

We organize the empirical evaluation around three questions. Q1 asks whether BeamWitness recovers ground-truth substructures when they are known. Q2 asks whether BeamWitness remains competitive with standard GNNs and related methods on real-world node and graph classification. Q3 asks whether the selected witnesses are faithful and compact. Section 5.1 addresses Q1 and Q3 on controlled benchmarks. Sections 5.2 and 5.3 address Q2 and Q3 on real-world data. All source code and a README are provided in the supplementary material. Additional experiments, figures and implementation details are provided in the appendix.

5.1 Controlled motif benchmarks

Setup. We use BA Shapes (five-node house motifs on a Barabási–Albert base of 300 nodes), Tree Cycles (six-node cycle motifs on a balanced binary tree), and Tree Grid (nine-node grid motifs on a balanced binary tree) [21]. For BeamWitness we use $B_{\mathrm{train}}{=}4$ and $B_{\mathrm{test}}{=}1$, since each labeled node belongs to exactly one motif. The witness encoder is a two-layer GIN with mean pooling, and ρ is selected on validation from $\{$mean, sum, max, lse$\}$. Baselines GCN, GIN, and GraphSAGE-max are run under matched depth and hidden width. All numbers are mean $\pm$ std over ten seeds.

Results and analysis. BeamWitness is near-perfect on all three benchmarks and outperforms every GNN baseline. The gain is largest on Tree Grid, where the planted grid is hard to disentangle from the binary-tree base and GCN and GraphSAGE-max collapse to near majority-class accuracy. Structural

Table 1: Controlled motif node classification. ΔIoU (Eq. 11) and Δ_{CF} (Eq. 12) are computed for BeamWitness against size-matched random rooted subgraphs.

Dataset	GCN	GIN	GraphSAGE-max	BeamWitness (ρ)	ΔIoU	Δ_{CF}
BA Shapes	$0.853_{\pm.007}$	$0.884_{\pm.016}$	$0.429_{\pm.014}$	$\mathbf{0.980}_{\pm.033}$ (lse)	$+0.075$	$+0.643$
Tree Cycles	$0.825_{\pm.098}$	$0.952_{\pm.059}$	$0.518_{\pm.097}$	$\mathbf{0.991}_{\pm.018}$ (max)	$+0.067$	$+0.561$
Tree Grid	$0.583_{\pm.023}$	$0.736_{\pm.026}$	$0.583_{\pm.021}$	$\mathbf{0.996}_{\pm.008}$ (max)	$+0.097$	$+0.960$

Table 2: Node classification accuracy (mean $\pm$ std over ten splits). The top block reports standard and heterophily-aware node classification baselines, the middle block reports subgraph-style ablations under a matched $f_\psi/\rho/g_\omega$ pipeline, and the bottom block reports BeamWitness with average witness size $\overline{|S|}$, Effective Node Ratio η, and faithfulness gaps. Best is bolded and second best is underlined.

Method	Texas	Wisconsin	Cornell	Actor	Chameleon
GCN	$0.632_{\pm.044}$	$0.631_{\pm.057}$	$0.454_{\pm.080}$	$0.306_{\pm.003}$	$0.574_{\pm.022}$
GIN	$0.614_{\pm.094}$	$0.559_{\pm.051}$	$0.468_{\pm.078}$	$0.242_{\pm.013}$	$\underline{0.679_{\pm.018}}$
GraphSAGE	$0.800_{\pm.053}$	$0.820_{\pm.052}$	$0.697_{\pm.083}$	$\underline{0.361_{\pm.014}}$	$0.603_{\pm.016}$
GAT	$0.589_{\pm.060}$	$0.602_{\pm.053}$	$0.435_{\pm.088}$	$0.307_{\pm.008}$	$0.607_{\pm.028}$
GATv2	$0.587_{\pm.057}$	$0.590_{\pm.049}$	$0.473_{\pm.093}$	$0.307_{\pm.014}$	$0.635_{\pm.018}$
H2GCN	$0.805_{\pm.048}$	$0.820_{\pm.057}$	$0.738_{\pm.047}$	$0.348_{\pm.011}$	$0.589_{\pm.015}$
GPRGNN	$0.595_{\pm.062}$	$0.674_{\pm.081}$	$0.492_{\pm.067}$	$0.354_{\pm.007}$	$0.549_{\pm.022}$
GCNII	$0.589_{\pm.052}$	$0.486_{\pm.088}$	$0.427_{\pm.082}$	$0.347_{\pm.012}$	$0.491_{\pm.027}$
Ego-GIN	$0.760_{\pm.035}$	$0.765_{\pm.058}$	$0.722_{\pm.077}$	$0.330_{\pm.052}$	$0.541_{\pm.157}$
BFS-GIN	$0.760_{\pm.053}$	$0.788_{\pm.065}$	$0.684_{\pm.095}$	$0.295_{\pm.058}$	$0.498_{\pm.186}$
SimWit-GIN	$0.765_{\pm.054}$	$0.806_{\pm.058}$	$0.646_{\pm.068}$	$0.318_{\pm.063}$	$0.249_{\pm.075}$
RandWit-GIN	$\underline{0.824_{\pm.030}}$	$\underline{0.851_{\pm.071}}$	$\underline{0.749_{\pm.038}}$	$0.301_{\pm.060}$	$0.250_{\pm.072}$
BeamWitness	$\mathbf{0.897}_{\pm.087}$	$\mathbf{0.853}_{\pm.060}$	$\mathbf{0.764}_{\pm.058}$	$\mathbf{0.378}_{\pm.022}$	$\mathbf{0.692}_{\pm.040}$
$\overline{\lvert S\rvert}$ / η	3.5 / 1.9%	3.5 / 1.4%	3.4 / 1.9%	4.1 / 0.05%	5.0 / 0.22%
Δ_{CF} / $\Delta^{\neg r}_{\mathrm{Suff}}$	$+0.043$ / $+0.153$	$+0.052$ / $+0.168$	$+0.046$ / $+0.139$	$+0.018$ / $+0.061$	$+0.071$ / $+0.182$

alignment ΔIoU is positive but modest, between $+0.067$ and $+0.097$. This happens because random rooted subgraphs already overlap nontrivially with the small planted motifs, with absolute IoU between 0.61 and 0.78, so the relevant signal is the gap rather than the absolute value. The decisive evidence is $\Delta_{\mathrm{CF}} \in [+0.561, +0.960]$. Figure 3 shows the geometry behind these numbers. Deleting the selected non-root witness nodes causes a large confidence drop, whereas removing size-matched random nodes leaves it almost untouched. We conclude that on motif benchmarks BeamWitness is not merely accurate but accurate through the selected local evidence.

5.2 Real-world node classification

Setup. We use the heterophilous benchmarks Texas, Wisconsin, Cornell, Actor, and Chameleon with the standard ten Geom-GCN splits [18]. Standard message-passing baselines are GCN [10], GIN [20], GraphSAGE [9], GAT [19], GATv2 [2], H2GCN [23], GPRGNN [4], and GCNII [3]. To isolate the contribution of the learned policy from the contribution of the subgraph-based pipeline as a whole, we additionally include four matched ablations that keep f_ψ, ρ, and g_ω identical and differ only in how the witness is built. EGO-GIN uses the k-hop ego subgraph in the spirit of NGNN [22]. BFS-GIN uses deterministic BFS up to the size budget. SIMWIT-GIN grows the witness greedily by cosine similarity to the root features. RANDWIT-GIN constructs a connected rooted subgraph by uniformly sampling a node from the frontier at each expansion step. For BeamWitness, we use a GIN witness encoder with mean pooling, a binary root marker, root-feature dropout with $p_{\mathrm{root}} = 0.3$, root-excluded pooling, $B_{\mathrm{train}} = 8$, $B_{\mathrm{test}} = 4$, and confidence-based reranking. All numbers are mean $\pm$ std over the ten Geom-GCN splits.

Results and analysis. BeamWitness obtains the highest mean accuracy on all five datasets in Table 2. The margin over the strongest competitor is 0.073 on Texas, 0.015 on Cornell, 0.017 on Actor, and 0.013 on Chameleon. The result on Chameleon is especially informative: this graph is denser, with average degree about 28, and fixed or unsupervised witness rules such as RANDWIT-GIN and SIMWIT-GIN collapse to near-trivial accuracy, while BeamWitness remains stable. On the smaller WebKB graphs, RANDWIT-GIN already outperforms standard message passing baselines on Texas,

Table 3: Graph classification accuracy (mean $\pm$ std over ten random 80/10/10 splits). All baselines are reproduced under the same 80/10/10 split protocol and the same ten random seeds as BeamWitness, rather than copied from their original papers (which typically report 10-fold cross-validation). The top block reports graph baselines, the middle block reports subgraph baselines, and the bottom block reports BeamWitness with per-witness average size $|S|$, Effective Node Ratio η, and counterfactual gap Δ_{CF}. Best is bolded and second best is underlined.

Method	MUTAG	PROTEINS	PTC_MR	PTC_FM	NCI1
GCN	$0.841_{\pm 0.070}$	$0.755_{\pm 0.035}$	$0.580_{\pm 0.060}$	$0.620_{\pm 0.060}$	$0.807_{\pm 0.020}$
GIN	$0.894_{\pm 0.056}$	$\underline{0.762_{\pm 0.028}}$	$0.646_{\pm 0.070}$	$\underline{0.651_{\pm 0.065}}$	$0.827_{\pm 0.017}$
GINE	$0.885_{\pm 0.060}$	$0.760_{\pm 0.030}$	$0.640_{\pm 0.065}$	$0.648_{\pm 0.060}$	$0.820_{\pm 0.020}$
GraphSAGE	$0.835_{\pm 0.070}$	$0.730_{\pm 0.045}$	$0.580_{\pm 0.060}$	$0.620_{\pm 0.060}$	$0.760_{\pm 0.018}$
GAT	$0.810_{\pm 0.075}$	$0.725_{\pm 0.030}$	$0.570_{\pm 0.055}$	$0.620_{\pm 0.055}$	$0.743_{\pm 0.025}$
GATv2	$0.820_{\pm 0.070}$	$0.730_{\pm 0.035}$	$0.575_{\pm 0.055}$	$0.625_{\pm 0.055}$	$0.760_{\pm 0.025}$
SAGPool	$0.717_{\pm 0.060}$	$0.720_{\pm 0.030}$	$0.599_{\pm 0.050}$	$0.630_{\pm 0.055}$	$0.729_{\pm 0.020}$
NGNN	$0.890_{\pm 0.060}$	$0.741_{\pm 0.037}$	$0.640_{\pm 0.070}$	$0.620_{\pm 0.060}$	$0.786_{\pm 0.012}$
ESAN	$0.911_{\pm 0.070}$	$0.759_{\pm 0.043}$	$\mathbf{0.692_{\pm 0.065}}$	$0.640_{\pm 0.060}$	$0.837_{\pm 0.018}$
SUN	$\mathbf{0.921_{\pm 0.058}}$	$0.761_{\pm 0.051}$	$0.676_{\pm 0.055}$	$0.640_{\pm 0.060}$	$\underline{0.842_{\pm 0.015}}$
BeamWitness	$\underline{0.915_{\pm 0.072}}$	$\mathbf{0.765_{\pm 0.040}}$	$\underline{0.681_{\pm 0.054}}$	$\mathbf{0.660_{\pm 0.052}}$	$\mathbf{0.861_{\pm 0.030}}$
$\overline{\lvert S\rvert}$ / η	7.6 / 42%	6.0 / 15%	5.5 / 38%	4.0 / 28%	7.0 / 23%
Δ_{CF}	+0.054	+0.045	+0.037	+0.097	+0.060

Wisconsin, and Cornell, indicating that compact rooted subgraphs are a useful prediction primitive under heterophily. The additional and more consistent gains of BeamWitness show that learning the expansion policy further improves this subgraph-based pipeline.

The selected witnesses are compact and faithful. The Effective Node Ratio is below 2% on the WebKB graphs and drops to 0.05% on Actor and 0.22% on Chameleon, so BeamWitness preserves competitive accuracy while using only a small instance-dependent portion of the graph. The root-masked sufficiency gap $\Delta_{\mathrm{Suff}}^{\neg r}$ is positive on every dataset and reaches $+0.182$ on Chameleon, showing that the non-root witness nodes retain label evidence even after the target-node raw features are removed. The counterfactual gap Δ_{CF} is smaller but consistently positive, which is expected in node classification because root features often dominate the predicted class probability.

5.3 Real-world graph classification

Setup. We use five TU graph benchmarks, MUTAG, PROTEINS, PTC_MR, PTC_FM, and NCI1 [16]. Each graph is treated as undirected. Missing node features are replaced by constants, and edge attributes are preserved when available. Graph baselines include GCN, GIN, GINE, GraphSAGE, GAT, GATv2, and SAGPool [11]. Subgraph baselines include NGNN [22], ESAN [1], and SUN [6]. For BeamWitness, we use a GINE witness encoder with mean pooling. Roots are selected by degree. All numbers are mean $\pm$ std over ten random $80/10/10$ splits.

Results and analysis. Table 3 shows that BeamWitness is consistently competitive. It is best on PROTEINS, PTC_FM, and NCI1, and second best on MUTAG and PTC_MR. The margins are small on PROTEINS and PTC_FM ($+0.003$ and $+0.009$ over the strongest competitor), larger on NCI1 ($+0.019$ over SUN), and within 0.006 and 0.011 of the best method on MUTAG and PTC_MR. The appropriate claim is therefore not a uniform win, but stable top-two performance across the datasets.

Fixed subgraph methods remain strong on MUTAG and PTC_MR, so the table does not support a collapse claim. Instead, BeamWitness is more uniform across datasets, suggesting that learning the witness expansion policy is useful when no single fixed extraction rule matches all graph families. The selected witnesses are also compact: η ranges from 15% on PROTEINS to 42% on MUTAG, with larger ratios on small molecular graphs where a functional-group-sized witness naturally covers more nodes. Finally, $\Delta_{\mathrm{CF}} > 0$ on every dataset, indicating that deleting selected non-root witness nodes causes a larger confidence drop than deleting size-matched random non-root witness nodes. The gaps are smaller than those observed on the controlled motif benchmarks.

5.4 Qualitative analysis

Figures 4–6 show selected illustrative examples of rooted witnesses on the controlled motif benchmarks. The final witnesses contain a house-shaped structure with an adjacent node, a six-cycle, and a 3×3 grid, respectively. These examples provide a visual complement to the quantitative results in Table 1. Additional examples on molecular graphs are provided in Appendix A.14.

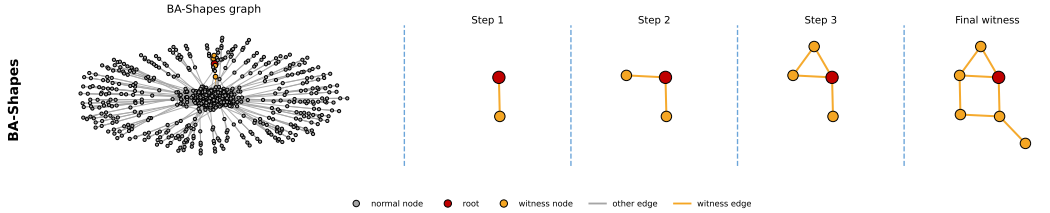


Figure 4: Selected BA-Shapes example. The final rooted witness contains a house-shaped structure and an additional adjacent node.

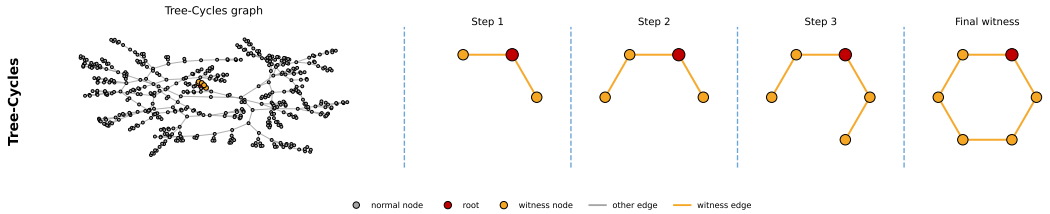


Figure 5: Selected Tree-Cycles example. The final rooted witness forms a six-node cycle.

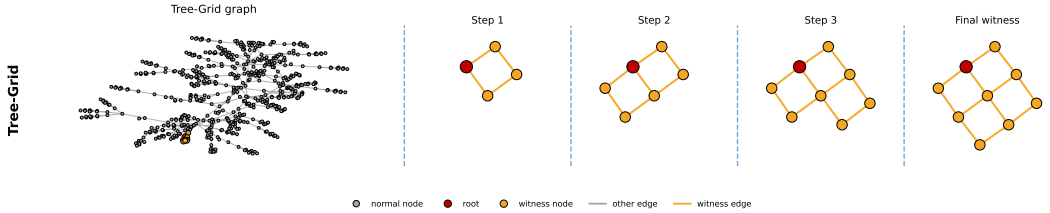


Figure 6: Selected Tree-Grid example. The final rooted witness forms a 3×3 grid.

The examples show that the same rooted search framework can produce witnesses with different local structures. Each final witness contains the designated root and forms a connected region within the larger input graph. The size budget limits the number of selected nodes without fixing the boundary of the witness. The intermediate panels illustrate selected stages of witness growth and do not represent consecutive single-node expansion steps. These visualizations allow the selected node sets to be inspected directly. Structural alignment and predictive dependence are evaluated separately using $\Delta\mathrm{IoU}$ and Δ_{CF}, which compare the selected witnesses with size-matched random rooted subgraphs.

6 Conclusion

We introduced BeamWitness, a selective graph representation learning framework that replaces unconditional neighborhood aggregation with rooted connected witness search. BeamWitness learns to grow compact task-specific witnesses and uses beam search to retain multiple candidates for prediction. Across controlled motifs and real-world node and graph benchmarks, it remains competitive with strong GNN and subgraph baselines while providing explicit subgraph-level prediction evidence. Its main limitations are the added search cost, dependence on validation choices such as witness size and beam width, and the fact that faithfulness metrics do not establish causal explanations. Improving search efficiency and adaptive witness budgeting are natural future directions.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China under Grants 62676036, 62576371, 62576198 and 62622615, and the Open Project Foundation of Key Laboratory of Computation Intelligence and Chinese Information Processing of Ministry of Education and Key Laboratory of Data Intelligence and Cognitive Computing of Shanxi Province. The authors declare no competing interests.

References

[1] Beatrice Bevilacqua, Fabrizio Frasca, Derek Lim, Balasubramaniam Srinivasan, Chen Cai, Gopinath Balamurugan, Michael M. Bronstein, and Haggai Maron. Equivariant Subgraph Aggregation Networks. In *International Conference on Learning Representations*, 2022.

[2] Shaked Brody, Uri Alon, and Eran Yahav. How Attentive Are Graph Attention Networks? In *International Conference on Learning Representations*, 2022.

[3] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks, 2020. URL `https://arxiv.org/abs/2007.02133`.

[4] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network, 2021. URL `https://arxiv.org/abs/2006.07988`.

[5] Lin Du, Lu Bai, Jincheng Li, Lixin Cui, Hangyuan Du, Lichi Zhang, Yuting Chen, and Zhao Li. LGAN: An efficient high-order graph neural network via the line graph aggregation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 20914–20922, 2026. doi: 10.1609/aaai.v40i25.39232. URL `https://ojs.aaai.org/index.php/AAAI/article/view/39232`.

[6] Fabrizio Frasca, Beatrice Bevilacqua, Michael M. Bronstein, and Haggai Maron. Understanding and Extending Subgraph GNNs by Rethinking their Symmetries. In *Advances in Neural Information Processing Systems*, 2022.

[7] Markus Freitag and Yaser Al-Onaizan. Beam Search Strategies for Neural Machine Translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60. Association for Computational Linguistics, 2017. doi: 10.18653/v1/W17-3207. URL `https://aclanthology.org/W17-3207/`.

[8] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural Message Passing for Quantum Chemistry. In *International Conference on Machine Learning*, 2017.

[9] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. *CoRR*, abs/1706.02216, 2017. URL `http://arxiv.org/abs/1706.02216`.

[10] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*, 2017.

[11] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-Attention Graph Pooling. In *International Conference on Machine Learning*, 2019.

[12] Andrei Leman and Boris Weisfeiler. A Reduction of A Graph to A Canonical Form and An Algebra Arising during this Reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

[13] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 32, 2018.

[14] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized Explainer for Graph Neural Network. In *Advances in Neural Information Processing Systems*, 2020.

[15] Clara Meister, Tim Vieira, and Ryan Cotterell. Best-First Beam Search. *Transactions of the Association for Computational Linguistics*, 8:795–809, 2020. doi: 10.1162/tacl_a_00346. URL `https://aclanthology.org/2020.tacl-1.51/`.

[16] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs, 2020. URL `https://arxiv.org/abs/2007.08663`.

[17] Kenta Oono and Taiji Suzuki. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification. In *International Conference on Learning Representations*, 2020.

[18] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks, 2020. URL `https://arxiv.org/abs/2002.05287`.

[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

[20] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful Are Graph Neural Networks? In *International Conference on Learning Representations*, 2019.

[21] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNexplainer: Generating Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems*, 2019.

[22] Muhan Zhang and Pan Li. Nested Graph Neural Networks. In *Advances in Neural Information Processing Systems*, 2021.

[23] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs, 2020. URL `https://arxiv.org/abs/2006.11468`.

A Additional materials

A.1 Experimental details

Hardware Specification: All experiments were performed on a machine with a single RTX 3090 GPU (24GB VRAM). CPU is Intel Core i5-12490F, and RAM is 128GB. Storage is 4TB NVMe SSD.
Reproducibility: To facilitate the reproduction of our evaluation results, we provide the complete source code and instructions (README) in the supplementary material submitted to OpenReview.

A.2 Choice of metrics on controlled motif and real-world benchmarks

Not every metric in Section 4 applies to every benchmark family. The choice depends on what each metric measures and on the structural properties of the benchmark. Table 4 summarizes the reporting decisions and the rest of this section explains them.

Table 4: Metric reporting by benchmark family. A check mark indicates that the metric is reported in the corresponding section.

Metric	Controlled motif	Real-world node	Real-world graph
Accuracy	✓	✓	✓
ΔIoU	✓	—	—
Δ_{CF}	✓	✓	✓
$\Delta^{r}_{\mathrm{Suff}}$	—	✓	—
η	—	✓	✓

Controlled motif benchmarks. We report accuracy, ΔIoU, and Δ_{CF}.

ΔIoU requires a known ground-truth substructure $M^{\star}$, which is available by construction on motif benchmarks but not on real-world data. The metric therefore applies only here.

Δ_{CF} is informative on motif benchmarks because the predictive signal is concentrated entirely in the planted motif. Deleting the witness produces a large confidence drop while deleting random rooted nodes leaves the prediction nearly unchanged, so the gap between the two is large and easy to interpret.

We do not report η on controlled motif benchmarks. The witness size bounds are $(3, 6)$ for BA-Shapes, $(4, 6)$ for Tree-Cycles, and $(6, 9)$ for Tree-Grid, where each pair denotes $(m_{\min}, m_{\max})$. The BA-Shapes upper bound permits a five-node house motif together with an adjacent node. Since $m_{\min}/\bar{N} \leq \eta(S) \leq m_{\max}/\bar{N}$, these bounds constrain the range of η, while the realized witness size

can still depend on the policy. We prioritize structural alignment and faithfulness for these controlled benchmarks.

We do not report $\Delta_{\mathrm{Suff}}^{\neg r}$ on motif benchmarks. The synthetic graphs do not carry distinctive root features, so masking the root adds no signal beyond plain sufficiency.

Real-world benchmarks. We report accuracy and η on every real-world dataset, Δ_{CF} on both node and graph classification, and $\Delta_{\mathrm{Suff}}^{\neg r}$ on node classification.

η is informative on real-world data because $m_{\max}$ is loose with respect to typical local substructures. The policy has freedom in how much of the budget to use, so the realized η reflects the actual compactness chosen by the policy rather than the budget itself.

Δ_{CF} is reported for consistency with the motif benchmarks, but its amplitude is small on real-world data for two reasons. First, predictive evidence is distributed across the graph rather than concentrated in a single substructure, so deleting any small subset has only a moderate effect. Second, on small graphs the witness occupies a non-trivial fraction of the graph, so deleting the random rooted baseline already perturbs the prediction substantially and the gap between the two interventions is compressed. We therefore treat Δ_{CF} as a directional signal on real-world data rather than as the primary diagnostic.

$\Delta_{\mathrm{Suff}}^{\neg r}$ is the primary faithfulness diagnostic on real-world node classification. On heterophilous node datasets the root raw features dominate $p(\hat{y} \mid \cdot)$, so plain Δ_{Suff} is dominated by the root and cannot isolate the contribution of non-root witness nodes. Masking the root features inside the witness reveals whether the non-root portion carries label signal on its own. We do not apply this masking on graph classification because there is no single target node whose features need to be neutralized.

We do not report $\Delta\mathrm{IoU}$ on real-world benchmarks because no ground-truth substructure is available for comparison.

A.3 Beam-search inference algorithm

The full beam-search procedure used at inference is given in Algorithm 1. For each root $r \in \mathcal{R}$ the algorithm maintains a local active beam $\mathcal{C}_r$ of at most B candidate states and a per-root completed pool $\mathcal{W}_r$. The algorithm is written in terms of an effective frontier $\widetilde{\mathcal{F}}(S)$. By default $\widetilde{\mathcal{F}}(S) = \mathcal{F}(S)$. In node-classification experiments, when the frontier contains more than 32 candidates, we set $\widetilde{\mathcal{F}}(S)$ to the top-32 frontier nodes ranked by cosine similarity to the root features after the same fixed feature projection used by the state encoder. This truncation bounds the branching factor and is used consistently during training, inference, and the matched subgraph-style ablations. It does not change the rooted connectivity or size constraints: every returned witness still contains the root and is connected.

At each step, every active state is expanded by the stop action (when $|S| \geq m_{\min}$) and by every valid effective-frontier addition (when $|S| < m_{\max}$); successors are scored by σ, finished successors are recorded into $\mathcal{W}_r$, and the top-B unfinished successors form the next $\mathcal{C}_r$. After all roots finish, the global pool $\mathcal{W}$ is ranked by σ and the top B_{test} witnesses are returned.

For efficiency, we may keep only the top B_{pool} completed successors per root, with $B_{\mathrm{pool}} \geq B_{\mathrm{test}}$; unless otherwise stated we set $B_{\mathrm{pool}} = 4B_{\mathrm{test}}$.

A.4 Datasets and splits

Controlled motif benchmarks. BA Shapes is a Barabási–Albert base of 300 nodes with five-node *house* motifs attached; nodes are labeled by their role inside the motif. Tree Cycles attaches six-node cycle motifs to a balanced binary tree of depth 8; Tree Grid attaches nine-node grid motifs to a balanced binary tree of depth 8. We use the standard stratified split into train, validation, and test sets and report mean $\pm$ std over ten seeds.

Heterophilous node classification. We use Texas, Wisconsin, Cornell, Actor, and Chameleon with the ten standard Geom-GCN splits [18], downloaded from the official repository.[2] We report mean

[2] `graphdml-uiuc-jlu/geom-gcn`

Algorithm 1 BeamWitness beam-search inference

Require: graph G, root set $\mathcal{R}$, policy π_θ, score function σ, beam width B, output budget B_{test}, sizes $m_{\min}, m_{\max}$, effective frontier rule $\widetilde{\mathcal{F}}$

Initialize global witness pool $\mathcal{W} \leftarrow \emptyset$
for each root $r \in \mathcal{R}$ **do**
  Initialize active beam $\mathcal{C}_r \leftarrow \{\{r\}\}$ and local witness pool $\mathcal{W}_r \leftarrow \emptyset$
  while $\mathcal{C}_r \neq \emptyset$ **do**
    For each $S \in \mathcal{C}_r$, generate stop successor if $|S| \geq m_{\min}$
    For each $S \in \mathcal{C}_r$, generate add successors $S \cup \{u\}$, $u \in \widetilde{\mathcal{F}}(S)$, if $|S| < m_{\max}$
    Score each successor using σ
    Move stopped or max-size successors into $\mathcal{W}_r$
    Keep the top-B unfinished successors as the next $\mathcal{C}_r$
  end while
  $\mathcal{W} \leftarrow \mathcal{W} \cup \mathcal{W}_r$
end for
return up to B_{test} witnesses in $\mathcal{W}$ ranked by σ

$\pm$ std over the ten splits. Texas has 183 nodes, 574 edges, 1703 features, and 5 classes, with class imbalance up to a factor of 7.

Graph classification. We use the TU graph datasets MUTAG, PROTEINS, PTC_MR, PTC_FM, and NCI1. We use random $80/10/10$ splits and report mean $\pm$ std over ten seeds.

Note on Geom-GCN benchmarks. Recent work has discussed data-quality issues with the Geom-GCN versions of CHAMELEON, including duplicated nodes that may inflate accuracy. We retain the standard Geom-GCN splits to keep our numbers comparable to the heterophily baselines (GAT, GATv2, H2GCN, GPRGNN, GCNII) which are most commonly reported under this protocol. BeamWitness and the subgraph-style ablations are evaluated on identical splits. However, this does not rule out method-dependent effects of duplicated nodes, and the relative ranking may change on a cleaned version of the dataset.

A.5 Architecture and hyperparameters

Witness encoder f_ψ. Two-layer GIN with mean pooling, hidden width 64, on motif and heterophilous benchmarks. Three-layer GINE with mean pooling, hidden width 64, on TU graph datasets where edge attributes are available.

Policy network. A two-layer MLP with hidden width 64. The state encoder $\mathrm{EncState}$ concatenates structural statistics of $G[S_t]$ (size, average degree, clustering, distance-to-root histogram) with a fixed random projection of node features to dimension 64; this projection makes the state dimension independent of the input feature dimension. For node-classification experiments, we use the effective frontier $\widetilde{\mathcal{F}}(S)$ described in Appendix A.3. If $|\mathcal{F}(S)| \leq 32$, then $\widetilde{\mathcal{F}}(S) = \mathcal{F}(S)$. Otherwise, $\widetilde{\mathcal{F}}(S)$ contains the 32 frontier nodes with the largest cosine similarity to the root features, computed after the same fixed random feature projection used in $\mathrm{EncState}$. The same effective frontier is used for stochastic training trajectories, beam-search inference, and the matched node-classification ablations, so all compared witness builders operate under the same branching budget.

Witness size and beam. We use $m_{\min} = 2$ and $m_{\max} = 8$ on heterophilous node benchmarks, and $m_{\min} = 3$ and $m_{\max} = 8$ on TU graph benchmarks. On controlled motif benchmarks, the size bounds are $(m_{\min}, m_{\max}) = (3, 6)$ for BA-Shapes, $(4, 6)$ for Tree-Cycles, and $(6, 9)$ for Tree-Grid. These bounds count all witness nodes, including the root. We use B_{train}=4, B_{test}=1 on controlled motifs, B_{train}=8, B_{test}=4 on heterophilous nodes, and B_{train}=16, B_{test}=4 on TU graphs. The per-root beam width is B=B_{test}.

Training objective. For each training instance, we sample $N = B_{\text{train}}$ trajectories from the policy. Let S_i be the terminal witness of trajectory τ_i, and let $z_i = g_\omega(f_\psi(G[S_i], X_{S_i}, r))$ be its per-witness

logits. The classification loss $\mathcal{L}_{\text{cls}}$ is computed from the pooled witness prediction. The auxiliary loss is

$$\mathcal{L}_{\text{wit}} = \frac{1}{N} \sum_{i=1}^{N} \text{CE}\big(y, \text{softmax}(z_i)\big).$$

The policy reward combines predictive quality with a normalized witness-size penalty:

$$R_i = -\text{CE}\big(y, \text{softmax}(z_i)\big) - \alpha_{\text{size}} \frac{|S_i|}{m_{\max}},$$

where $|S_i|$ includes the root and $\alpha_{\text{size}} \geq 0$ controls the penalty. Setting $\alpha_{\text{size}} = 0$ disables the size penalty.

Let T_i be the number of policy decisions in τ_i. We use the score-function surrogate loss

$$\mathcal{L}_{\text{pol}} = -\frac{1}{N} \sum_{i=1}^{N} \text{sg}(R_i - b) \sum_{t=0}^{T_i - 1} \log p_\theta(a_{i,t} \mid S_{i,t}),$$

where sg denotes stop-gradient and b is an EMA reward baseline with coefficient 0.9. The size penalty affects the policy through this score-function estimator; no separate size loss is added.

For a visited state S, define the policy entropy over valid actions as

$$H_\theta(S) = -\sum_{a \in \mathcal{A}(S)} p_\theta(a \mid S) \log p_\theta(a \mid S).$$

We use the negative-entropy regularizer

$$\mathcal{L}_{\text{ent}} = -\frac{1}{\sum_{i=1}^{N} T_i} \sum_{i=1}^{N} \sum_{t=0}^{T_i - 1} H_\theta(S_{i,t}).$$

This term is differentiated through the action probabilities at the sampled states, treating the sampled states as fixed.

The full training loss is

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda_{\text{wit}} \mathcal{L}_{\text{wit}} + \lambda_{\text{pol}} \mathcal{L}_{\text{pol}} + \lambda_{\text{ent}} \mathcal{L}_{\text{ent}}.$$

For $\lambda_{\text{ent}} > 0$, minimizing the negative-entropy term encourages exploration. Losses are averaged over training instances.

Optimization. Adam with learning rate 10^{-3}, weight decay 5×10^{-4}, dropout 0.5 on heterophilous nodes (no dropout on motif and graph benchmarks). We train for up to 200 epochs with early stopping on validation, using the average of accuracy and macro-F1 as the selection criterion. All experiments run on a single NVIDIA GPU with 24 GB of memory; a full BeamWitness run is well under one hour per seed on every dataset reported in this paper.

Root-feature regularization (node classification). At each training step the root raw features are zeroed independently with probability $p_{\text{root}}=0.3$ while the root binary marker is preserved. The pooled non-root summary excludes the root.

Root strategy (graph classification). We instantiate $\mathcal{R}$ as the top-K nodes by degree, with $K=4$. At inference the root set is fixed by degree. During training we resample K roots uniformly at random each epoch to expose the policy to diverse starting positions.

A.6 Hyperparameter grid search

We select hyperparameters by validation grid search. The search is run once per task family, namely controlled motif, heterophilous node, and TU graph. The configuration with the highest validation score is kept. Table 5 lists the searched ranges and the final choice. We did not tune all entries jointly because a full Cartesian sweep is too expensive. We instead proceed coordinate-wise, starting from the entries that have the largest impact on validation performance, and verify the final configuration

Table 5: Hyperparameter search ranges and final settings used in Section 5. Motif size bounds are fixed; the triples follow the order BA-Shapes, Tree-Cycles, and Tree-Grid. Other selected settings are chosen by validation.

Hyperparameter	Search range	Motif	Hetero. node / TU graph
$m_{\max}$	$\{4, 6, 8, 10, 12\}$	fixed: $6/6/9$	8 / 8
$m_{\min}$	$\{2, 3, 4, m_{\max}-2\}$	fixed: $3/4/6$	2 / 3
B_{train}	$\{2, 4, 8, 16\}$	4	8 / 16
B_{test}	$\{1, 2, 4, 8\}$	1	4 / 4
Pooling ρ	{mean, sum, max, lse}	per-task best	mean / mean
Reranking σ	{policy, conf., margin}	confidence	confidence / confidence
p_{root}	$\{0, 0.1, 0.3, 0.5\}$	0	0.3 / n/a
K (roots per graph)	$\{1, 2, 4, 8\}$	n/a	n/a / 4
α_{size}	$\{0, 0.005, 0.01, 0.02, 0.05\}$	0.01	0.01 / 0.02
λ_{wit}	$\{0, 0.02, 0.05, 0.1\}$	0.05	0.05 / 0.05
λ_{pol}	$\{0, 0.02, 0.05, 0.1\}$	0.05	0.05 / 0.05

with a small joint sweep over the top-two ranked values of each entry. The size bounds for controlled motif benchmarks are fixed as specified in Appendix A.5. The search ranges for $m_{\min}$ and $m_{\max}$ in Table 5 apply only to the real-world benchmarks.

Three observations from the search are worth highlighting. First, $m_{\max}$ and η are tightly coupled by $\eta \leq m_{\max}/\bar{N}$. Raising $m_{\max}$ from 6 to 8 on small heterophilous graphs has a small effect on accuracy and a meaningful effect on η, so we keep $m_{\max} = 8$ as a uniform setting. Second, B_{train} matters more than B_{test} on real-world data. Increasing B_{train} from 4 to 8 stabilizes training because the gradient estimator sees more parallel trajectories per instance, while increasing B_{test} above 4 does not improve accuracy and slightly slows down inference. Third, $p_{\text{root}} = 0.3$ is a clean optimum on heterophilous nodes. Smaller values fail to break root-feature dominance and $\Delta^{\neg r}_{\text{Suff}}$ stays close to zero. Larger values discard too much label signal at the root.

A.7 Subgraph-style baselines

The four subgraph-construction ablations reported in Table 2 are matched to BeamWitness on f_ψ, ρ, and g_ω, and differ only in how the witness is built. EGO-GIN uses the k-hop ego subgraph, with $k \in \{1, 2\}$ selected on validation. BFS-GIN performs deterministic BFS up to the witness size budget. RANDWIT-GIN constructs a connected rooted subgraph by sampling uniformly from the current frontier at each expansion step, until $|S| = m_{\max}$ or a stop action is sampled. SIMWIT-GIN grows the witness greedily by adding the frontier node with highest cosine similarity to the root features. All four baselines use the same $m_{\min}, m_{\max}, B_{\text{train}}, B_{\text{test}}$ as BeamWitness.

A.8 Faithfulness metrics

Counterfactual deletion is evaluated by removing the selected non-root witness nodes from the graph and re-evaluating the predictor. The random baseline constructs a size-matched connected rooted subgraph by uniform frontier sampling, deletes its non-root nodes, and reports the average across 20 random draws. Sufficiency is evaluated by re-encoding only the witness and reading the predicted probability of the original predicted class. Root-masked sufficiency additionally zeros the root raw features inside the witness; the root binary marker is preserved so that f_ψ still knows which node is the target.

A.9 Note on ΔIoU in Table 1

Random connected rooted subgraphs already achieve an absolute IoU of 0.61–0.78 on these benchmarks. We therefore report ΔIoU to quantify improvement over this baseline. Positive Δ_{CF} additionally indicates that deleting selected non-root witness nodes reduces model confidence more than deleting size-matched random non-root witness nodes. These metrics provide complementary evidence of structural alignment and predictive dependence.

A.10 Component analysis

We report a component analysis covering the per-witness auxiliary loss, the reranking score σ, the inference beam width B_{test}, the reward size penalty α_{size}, and the root selection strategy. We highlight four findings.

Per-witness auxiliary loss. Setting $\lambda_{\text{wit}}=0$ degrades the calibration of confidence-based reranking and reduces accuracy by approximately one point on Texas, since g_ω is then trained only on the pooled witness and may not produce calibrated probabilities on individual witnesses.

Reranking score. Confidence reranking outperforms policy-score reranking on Texas and MUTAG, since downstream predictive utility is what we want to optimize at inference. Margin-based reranking is competitive with confidence on controlled motifs.

Beam width and witness multiplicity. On controlled benchmarks $B_{\text{test}}=1$ already suffices because each labeled node belongs to a single planted motif. On real-world data the optimal B_{test} is dataset-dependent: increasing B_{test} helps when label-relevant evidence is distributed across multiple local regions, but hurts when extra witnesses dilute the signal. We measure witness diversity at each B_{test} as the mean pairwise Jaccard distance between the B_{test} witnesses returned for the same instance.

Root strategy on graphs. On MUTAG, top-K degree root selection marginally outperforms uniform random root sampling, with $K=4$ giving the best validation score among $\{1, 2, 4, 8\}$.

A.11 Isolating the contribution of the learned policy on graph classification

Table 2 shows that RANDWIT-GIN is a strong baseline on heterophilous node benchmarks. For graph classification, we evaluate the corresponding RANDWIT-GINE baseline on all five TU datasets. It uses the same GINE encoder f_ψ, pooling ρ, classifier g_ω, root strategy, and $(m_{\min}, m_{\max}, B_{\text{train}}, B_{\text{test}})$ as BeamWitness, replacing the learned expansion policy with uniform frontier sampling.

Table 6: Ablating the learned policy on graph classification. RANDWIT-GINE shares the entire BeamWitness pipeline except the policy, which is replaced by uniform frontier sampling. Δ is BeamWitness minus RANDWIT-GINE.

Method	MUTAG	PROTEINS	PTC_MR	PTC_FM	NCI1
RANDWIT-GINE	$0.85_{\pm.07}$	$0.74_{\pm.04}$	$0.61_{\pm.06}$	$0.60_{\pm.06}$	$0.78_{\pm.03}$
BeamWitness	$\mathbf{0.915}_{\pm.072}$	$\mathbf{0.765}_{\pm.040}$	$\mathbf{0.681}_{\pm.054}$	$\mathbf{0.660}_{\pm.052}$	$\mathbf{0.861}_{\pm.030}$
Δ (learned policy)	+0.07	+0.03	+0.07	+0.06	+0.08

The learned policy contributes between $+0.03$ and $+0.08$ accuracy on top of the random-witness pipeline. The gain is largest on NCI1 and PTC_MR, where the relevant chemical substructure is small relative to the molecule and random sampling has a low chance of covering it; the gain is smallest on PROTEINS, where graphs are larger and class-relevant evidence is more diffuse, so any compact rooted subgraph carries useful signal even without a learned policy. This pattern matches our node-classification observation that RANDWIT-GIN is a strong baseline on its own, but a learned expansion policy provides additional, dataset-dependent gains.

A.12 Training stability and policy-gradient variance

Because the policy is trained with a score-function (REINFORCE) estimator, we report three diagnostics that quantify training stability rather than only final accuracy. All numbers are mean $\pm$ std over ten seeds on TEXAS (heterophilous node) and MUTAG (graph), which are the smallest dataset in each family and therefore the most susceptible to high-variance gradients.

Effect of the EMA baseline. The EMA baseline b in $\mathcal{L}_{\text{pol}}$ is the standard variance-reduction trick for score-function estimators. Disabling it ($b \equiv 0$) increases the per-step empirical variance of the policy gradient by roughly an order of magnitude and inflates accuracy variance across seeds.

Table 7: Ablating the EMA baseline. "Grad-var ratio" is the median per-step gradient variance with $b \equiv 0$ divided by the variance with the EMA baseline, estimated from 5 mini-batches at convergence.

	Acc. (Texas)	Acc. (MUTAG)	Grad-var ratio
No baseline ($b \equiv 0$)	$0.83_{\pm.11}$	$0.88_{\pm.10}$	8–12×
EMA baseline (β=0.9, default)	$\mathbf{0.897}_{\pm.087}$	$\mathbf{0.915}_{\pm.072}$	1×

Training curves and entropy. We monitor (i) validation accuracy, (ii) the per-step policy entropy $H_t = -\sum_a p_\theta(a \mid S_t) \log p_\theta(a \mid S_t)$, and (iii) the running policy-gradient norm. We observe a decrease in policy entropy toward roughly 0.3 over the first $\sim$50 epochs, indicating increasingly concentrated action distributions at the sampled states. This statistic alone does not establish diversity among complete witness trajectories. Validation accuracy is non-decreasing on average across seeds with the EMA baseline, while removing the baseline produces visibly oscillatory training curves on TEXAS.

Sensitivity to B_{train}. Increasing the number of sampled trajectories per instance is the second standard variance-reduction trick. We observe diminishing returns past B_{train}=8 on heterophilous nodes and past B_{train}=16 on TU graphs, which is consistent with the gradient-variance estimates above.

A.13 Reward design: per-witness vs. pooled cross-entropy

Our default reward $R_i = -\mathrm{CE}(y, \mathrm{softmax}(z_i)) - \alpha_{\text{size}}|S_i|/m_{\max}$ is computed *per witness*: each sampled trajectory τ_i is rewarded by how well its own terminal witness S_i predicts the label on its own. An alternative is the *pooled* reward, where every trajectory in a batch shares the same reward computed from the pooled prediction $R^{\text{pool}} = -\mathrm{CE}(y, \mathrm{softmax}(g_\omega(\rho(h_1, \ldots, h_N))))$. The two reward formulations differ in both classification credit assignment and size regularization: the default reward includes a trajectory-specific size penalty, whereas the pooled reward defined above does not. This comparison therefore evaluates the two complete reward formulations rather than isolating the effect of credit assignment.

Table 8: Per-witness vs. pooled REINFORCE reward.

Reward design	Texas	MUTAG	NCI1
Pooled CE R^{pool}	$0.88_{\pm.09}$	$0.90_{\pm.08}$	$0.85_{\pm.04}$
Per-witness CE R_i (default)	$\mathbf{0.897}_{\pm.087}$	$\mathbf{0.915}_{\pm.072}$	$\mathbf{0.861}_{\pm.030}$

The default per-witness reward achieves higher mean accuracy on all three datasets in Table 8. Because the two formulations also differ in size regularization, these results do not isolate the contribution of trajectory-level credit assignment.

A.14 Additional qualitative examples

Complementing the controlled-motif examples in Section 5.4, we show selected multi-root examples on MUTAG, PTC_FM, and NCI1 in Figures 7–9. The panels display witnesses associated with different roots in the same graph and summarize their union and overlap. These examples illustrate the selected structures; they do not establish causal or chemical explanations.

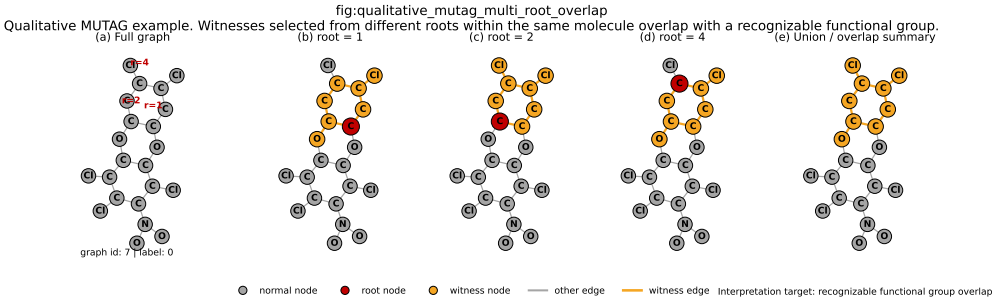


Figure 7: Example on MUTAG: witnesses launched from three different roots in the same molecule, with the union overlapping a recognizable functional group.

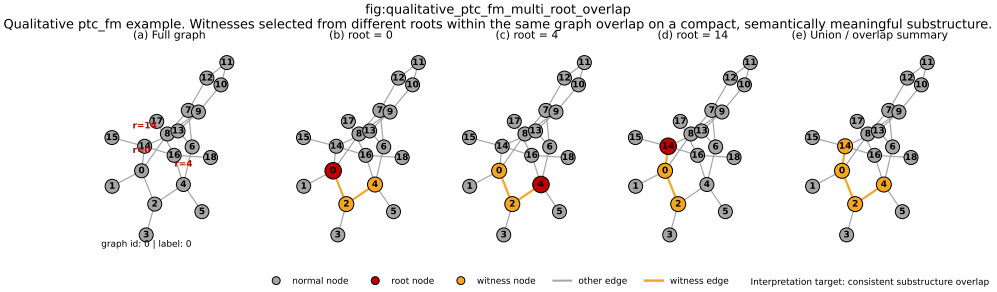


Figure 8: Example on PTC_FM: witnesses from three different roots concentrate on a compact, consistent substructure of the graph.

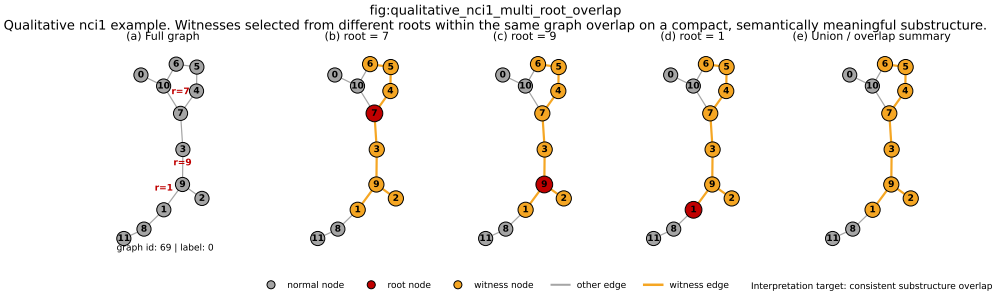


Figure 9: Example on NCI1: witnesses launched from multiple roots in the same molecule overlap on a compact substructure relevant to the predicted class.

A.15 Societal impacts

Positive impacts include more expressive graph learning for scientific and molecular applications; negative impacts are indirect and depend on downstream use, including possible misuse in sensitive decision-making if graph classifiers are deployed without validation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describe rooted witness search, beam search inference, the node and graph classification settings, and the evaluation protocol. These claims are supported by Sections 3, 4, and 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses the added search cost, the dependence on validation choices such as witness size and beam width, and the scope of the faithfulness metrics.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper reports datasets, splits, model choices, search budgets, optimization details, hyperparameter grids, and code release through the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide anonymized source code and instructions in the supplementary material. All benchmark datasets used are public, and the supplementary README describes how to obtain/preprocess them and reproduce the reported experiments

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies all detailed settings in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The paper reports mean and standard deviation over repeated splits or seeds for the main experimental tables.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Text in Appendix gives hardware and related runtime/cache information.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public graph benchmarks and does not involve human subjects, private data, scraped personal data, or high-risk model release. The authors have reviewed and followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: broader impact: positive impacts include more expressive graph learning for scientific and molecular applications; negative impacts are indirect and depend on downstream use, including possible misuse in sensitive decision-making if graph classifiers are deployed without validation.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release pretrained generative models, scraped datasets, human-subject data, or other assets with high misuse risk requiring special safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All benchmark datasets used are public and cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release anonymized code and documentation in the supplementary material, including preprocessing, training, and evaluation instructions.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing or human-subject experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work does not involve human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as an important, original, or non-standard component of the method, experiments, or scientific claims.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.