Overcoming Shortcut Learning in Graph Neural Networks through Active Explanation Guidance

Taraneh Younesian[1] (✉), Steve Azzolin[2], Antonio Longa[3], Francesco Ferrini[2], Vincenzo Marco De Luca[2], and Stefano Teso[2]

[1] VU Amsterdam, Netherlands t.younesian@vu.nl
[2] University of Trento, Italy
steve.azzolin,francesco.ferrini,vincenzomarco.deluca,stefano.teso@unitn.it
[3] UiT The Arctic University of Norway, Norway
antonio.longa@uit.no

Abstract. Graph Neural Networks (GNNs) can solve prediction tasks by unintentionally exploiting *shortcuts*—that is, edges, nodes, and features that correlate with but are not causal for the prediction—which compromise their reliability in out-of-distribution tasks. We introduce `XIGL` (eXplanatory Interactive Graph shortcut unLearning), an architecture-agnostic human-in-the-loop strategy for removing such shortcuts from GNNs. Our key insight is twofold. On the one hand, reliance on shortcuts can be detected by inspecting GNN explanations. On the other hand, once made aware of such shortcuts, sufficiently expert users can provide tailored corrective feedback, which helps deconfound the model. `XIGL` supports any query strategy; however, since corrective feedback can be expensive to acquire, we develop an active learning strategy for prioritizing explanations that are more likely to display shortcut behavior, lowering annotation and cognitive costs. We showcase the effectiveness of `XIGL`, including both existing and proposed explanation-based strategies, on several GNN architectures. Our implementation is available online. [4]

Keywords: Graph Neural Networks · Explainable AI · Active Learning · Shortcut Learning · Deconfounding.

1 Introduction

Graph Neural Network (GNN) classifiers can pick up on *shortcuts*, also called *confounders*, patterns that happen to correlate with the label in-distribution, allowing the model to achieve high accuracy but not generalize outside of it. Reliance on such shortcuts—including watermarks [11], metadata [6], and simple input statistics [16]—can prevent models from behaving properly upon deployment when the data distribution changes. Since providing more (observational) training data is insufficient to resolve shortcuts, existing works address them by employing other kinds of data, such as examples from multiple domains [1, 10], that are not always available.

[4] https://github.com/TYounesian/xilgraph.git

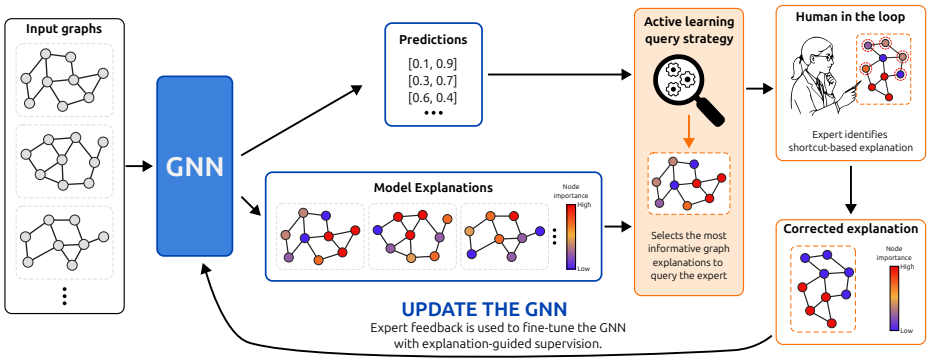


Fig. 1: **Overview of** `XIGL`. We first extract explanations from the GNN being trained over a small subset of the dataset; an active learning strategy then selects the most informative explanations for an expert to correct by indicating which nodes the model should not rely on and which nodes it should rely on instead; the corrections then fine-tune the GNN via an explanation-guided loss that removes shortcut dependence. The process iterates until the annotation budget is reached.

We take a different route. Specifically, we exploit the fact that, by unveiling the GNN's reasoning process, GNN *explanations* can naturally expose its reliance on shortcuts [2, 5, 6, 11, 16, 18]. We then design `XIGL` (eXplanatory Interactive Graph shortcut unLearning), an active learning pipeline, illustrated in Figure 1, that iteratively *i*) employs an intuitive selection strategy to identify those instances whose explanations are more likely to reveal shortcut behavior; *ii*) requests a human annotator to correct these explanations by indicating what nodes of the input graph the GNN should *not* rely on; and *iii*) adapts the GNN based on the user's corrections through an end-to-end differentiable loss function geared toward shortcut removal [18, 23].

Compared to existing work on explanation-based deconfounding of GNNs [28], which assumes that explanatory supervision is available for the entire training set, `XIGL` actively prioritizes annotating a subset of the data with those explanations that better capture dependency on shortcuts. This avoids the need to annotate examples where the model is behaving properly, while focusing the annotator's effort on subgraphs for which feedback is needed and useful, thus reducing annotation and cognitive costs.

Focusing on graph classification tasks, while `XIGL` supports any query strategy, our experiments demonstrate how `XIGL`'s explanation-based query strategies mostly improve on existing active learning baselines, which select instances based purely on prediction-based informativeness across several GNN architectures.

2 Preliminaries

Let $G = (\mathcal{V}, \mathcal{E}, X)$ be an (undirected) *graph*, where $\mathcal{V} = \{v_1, ..., v_N\}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, $X \in \mathbb{R}^{N \times d}$ is the node features, and

$y \in \{1, \ldots, l\}$ is the label assigned to G. We tackle *graph classification* problems where the goal is to learn a function $f : \mathcal{G} \to \mathcal{Y}$ mapping from the set of possible graphs $\mathcal{G}$ to the set of labels $\mathcal{Y}$ from training examples of annotated graphs.

We focus on scenarios in which the input graphs G encode both "causal" and "spurious" subgraphs, denoted C and S, respectively: whereas C determines the ground-truth label y, S only correlates with it. To build intuition, consider `CPatchMNIST` [4]: here, the graphs G represent handwritten digits in terms of superpixels (nodes) and their adjacency relation (edges). Node features encode the average color of the corresponding superpixel, and the label y is the digit itself. Crucially, the graphs are manipulated such that the color of the rightmost superpixels correlates with the digit, and as such act as shortcuts.

Neural nets are known to be biased to learn simpler solutions [26]. In real-world settings, the confounder S is often easier to learn than the causal one S. In our example, since the spurious superpixels are highly discriminative, GNNs tend to exploit them for prediction. This yields excellent in-distribution performance but near-random accuracy in unseen test instances where color is removed.

Throughout, we refer to the nodes in the causal subgraph C as *relevant* nodes and denote them $\mathcal{V}_r$, and to the nodes of its complement as *irrelevant* nodes, denoting them $\mathcal{V}_{ir}$. The latter covers the spurious subgraph S, as well as the rest of the graph. Ideally, we wish our GNNs to make predictions for the "right reason", that is, by relying on the causal subgraph rather than the spurious one.

3 Deconfounding through Active Explanation

We aim to deconfound the model by guiding it to not use shortcuts. We do so by penalizing the impact of irrelevant subgraphs of nodes V_{ir} on the model. As an indication of the impact of nodes, we use input gradients [17, 28], which we adopt among the many possible GNN explainers (see [14, 27] for a survey) for their simplicity and because they do not require training an ad-hoc explainer. We then minimize this value for irrelevant nodes in each graph, as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_e = -\sum_{i=1}^{l} \hat{y}_i \log(\hat{y}_i) + \lambda \frac{\sum_{j=1}^{N} (1 - e_j) s_j^2}{\sum_{j=1}^{N} s_j^2}, \tag{1}$$

where $\hat{y}_i$ is the predicted probability of class i, $s_j = \sum_{i=1}^{l} \nabla_j \log(\hat{y}_i)$ is sum of the input gradients, λ is a scaling term to balance between classification and explanation loss, and $\mathbf{e} \in \{0, 1\}^N$ is the ground-truth explanation of the graph G. In particular, for each node j, $\mathbf{e}$ indicates if j belongs to the relevant or irrelevant subgraph, i.e., e_j is 1 if v_j is relevant, i.e., $v_j \in \mathcal{V}_r$, and is 0 otherwise. In words, minimizing $\mathcal{L}_e$ corresponds to minimizing the ratio of the input gradients of irrelevant nodes and the total input gradients across all nodes. In contrast to [28], we incorporate a normalization term in the denominator to prevent the trivial solution in which the input gradients of all nodes, including the relevant ones, collapse to zero. The loss function above aims to balance learning the labels with learning the truly relevant subgraph. Computing the total loss gradient

during backpropagation yields second-order gradients with respect to the input. In practice, the resulting computational overhead was small.

3.1 Passive vs. Active Explanation Supervision

So far, we assumed access to the ground-truth explanation $\mathbf{e}$ for every graph in the training set. We call this scenario *passive explanation supervision*, since our model passively uses all instances to learn from. Obtaining ground-truth explanations for every instance is costly and time-intensive, as it requires expert annotation to determine the task-relevant nodes in each graph. Therefore, this assumption is impractical in real-world scenarios.

To address this limitation, we introduce an annotation budget B, which restricts the total number of graphs for which ground-truth explanations[5] can be collected. Instead of annotating the entire dataset, we select a small, informative subset of graphs so that training on this subset yields a highly accurate model that captures the underlying causal patterns.

We employ *active learning* (AL) [12, 19, 20, 29] to identify informative instances. While `XIGL` is agnostic to the choice of query strategy and can be combined with any active learning method, we introduce two novel explanation-based query strategies that leverage explanation uncertainty to identify instances whose explanation corrections are expected to provide the greatest benefit to the model. We include two standard query strategies in `XIGL` for comparison.

Let $\mathcal{D}_U = \{(G_i, y_i)\}_{i=1}^N$ be the dataset of graphs and their labels without their ground-truth explanation. The process begins by training the model on an initial small set of q graphs for which we have the explanations denoted as $\mathcal{D}_{\text{exp}} = \{(G_i, y_i, E_i)\}_{i=1}^q$. Then, among the remaining rest of $\mathcal{D}_U$, we select a batch of the most informative graphs and query an expert for their ground-truth explanations. The newly annotated batch is added to the previously annotated set $\mathcal{D}_{\text{exp}}$, and the model is subsequently fine-tuned on this set using Eq. 1. We repeat this procedure for $T \leq B$ iterations, ensuring that the total number of annotated instances does not exceed the budget. Algorithm 1 shows the steps of `XIGL`'s active learning. We investigate two classes of query strategies: prediction-based and explanation-based. The former are established active learning methods, whereas the latter are novel strategies proposed in this paper. The details of each strategy are provided below:

Prediction-based strategies

– **Maximum Classification Entropy** (MaCE) is one of the most popular uncertainty-based query strategies. MaCE queries graphs that the GNN is

[5] We use *ground-truth explanation* for simplicity. In practice, the explanation corrections provided by users do not need to match the true causal mechanism in the data, as they may not know the actual cause of the phenomenon being modeled. Yet, they may reliably indicate which nodes the model *should* rely on to avoid clear shortcuts.

Algorithm 1 Active Learning Steps in `XIGL`

Require: Labeled graph dataset $\mathcal{D}_U = \{(G_i, y_i)\}_{i=1}^N$, initial query budget q, query budget B, active learning iteration T, query strategy $\mathcal{S}$
Ensure: Updated GNN model f
1: Select q graphs and obtain their explanation correction:

$$\mathcal{D}_{\text{exp}} = \{(G_i, y_i, E_i)\}_{i=1}^q. \tag{2}$$

2: Train an initial GNN f_θ using: $\mathcal{L} = \mathcal{L}_{\text{ce}} + \lambda \mathcal{L}_{\text{e}}$.
3: **for** each iteration $t = 1, \ldots, T$ **do**
4: Generate predictions and explanations for graphs in $\mathcal{D}_U \setminus \mathcal{D}_{\text{exp}}$.
5: Select a query set $\mathcal{Q}$ of size B/T according to $\mathcal{S}$ (Section 3.1).
6: Obtain explanation correction for graphs in $\mathcal{Q}$.
7: Update the explanation-supervised set:

$$\mathcal{D}_{\text{exp}} \leftarrow \mathcal{D}_{\text{exp}} \cup \{(G_i, y_i, E_i) : G_i \in \mathcal{Q}\}. \tag{3}$$

8: Fine-tune the GNN using $\mathcal{L} = \mathcal{L}_{\text{ce}} + \lambda \mathcal{L}_{\text{e}}$.

highly uncertain to classify, using the Shannon entropy:

$$G^* = \underset{G \in \mathcal{D}_U}{\text{argmax}} - \sum_{i=1}^{l} p(y_i \mid G) \log p(y_i \mid G) \tag{4}$$

– **Random Sampling** randomly queries graphs according to a uniform distribution. Although random sampling does not make use of predictive information, we include it in this category as a baseline for comparison.

Explanation-based strategies

– **Maximum Explanation Entropy** (MaEE) queries the graphs that the GNN is most uncertain in its *explanation*. Specifically, it prioritizes graphs in which input gradients are most uniform across nodes, as measured by the entropy of softmax-normalized node-wise input gradients:

$$G^* = \underset{G \in \mathcal{D}_U}{\text{argmax}} - \frac{\sum_{i=1}^{|\mathcal{V}|} p_i \log p_i}{\log |G|}, \tag{5}$$

where $p_i = \frac{\exp(s_i)}{\sum_{j \in G} \exp(s_j)}$ is the softmax-normalized input gradient s_i. Since the number of nodes in each graph varies, we normalize the entropy to compare across graphs.
– **Minimum Explanation Entropy** (MiEE) queries the graphs that the GNN is most *certain* in its explanation. We adopt this strategy based on the simplicity bias of neural networks [26], which causes models to preferentially learn simpler yet spurious patterns. As a result, the model may quickly overfit

to confounders and produce overly confident explanations that incorrectly attribute the prediction to the confounding features.

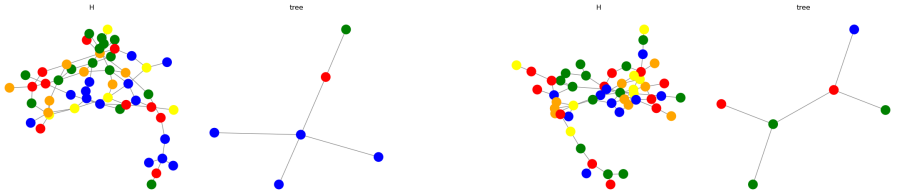
$$G^* = \underset{G \in \mathcal{D}_U}{\operatorname{argmin}} - \frac{\sum_{i=1}^{|\mathcal{V}|} p_i \log p_i}{\log |G|}. \tag{6}$$

4 Experiments

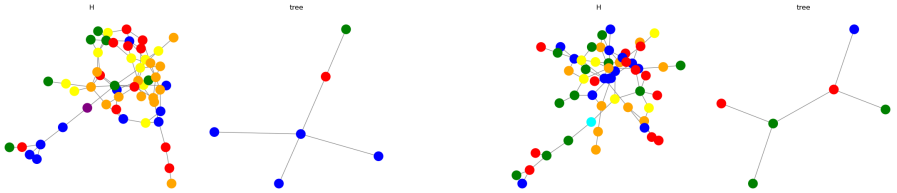
Datasets We use one synthetic and one real-world dataset for our experiments: ER-color is a synthetic dataset comprising 1000 Erdős-Rényi graphs with 50 nodes and edge probability of 0.05. We randomly assign the colors red, blue, green, yellow, and orange to the nodes using a uniform distribution. We split the dataset into 70%, 15%, and 15% for train, validation, and test sets, respectively. We then randomly attach two motifs to the base graphs, each representing a label. Hence, these motifs represent the *right reasons*. To create confounders that induce a distribution shift between the train and validation/test sets, we add j purple nodes for label 0 and j cyan nodes for label 1 as confounders only in the train set, while keeping the validation and test sets unchanged. Node features are represented as one-hot encoding of colors. For confounding colors, the active feature value is set to 100 rather than 1, thereby amplifying the confounding signal. We use $j = 1$ for passive learning experiments, and for active learning, we randomly select j where $j \sim \mathsf{Uniform}(1, \ldots, 25)$. The reason for the varying number of confounders in active learning experiments is to create different levels of informativeness among training instances. Figure 2 shows the cases with and without confounders, i.e., training or validation/test sets, respectively.

CPatchMNIST [4] is an extension of `MNISTsp` [9], itself a conversion of the MNIST dataset to graphs via a superpixelation algorithm. In `CPatchMNIST`, on the other hand, the top left and bottom right superpixels are colored according to the graph label in the training set to represent confounders. However, in validation and test sets, these colors are randomized.

Experimental Setup We examined four GNN architectures—`GCN` [8], `GIN` [25], `GraphSAGE` [7], and `GAT` [24]—to assess both their vulnerability to shortcut learning and their ability to recover from it by avoiding spurious dependencies through explanation guidance. Further implementation details are available in the Appendix. For the active learning experiments, we query 5 instances per round over 20 rounds for `ER-color`, and 50 instances per round over 10 rounds for `CPatchMNIST`. We observed that when the annotation budget is large, the gap between different query strategies largely vanishes. Intuitively, this occurs because the informative instances selected by active learning methods constitute a subset of the data that is increasingly likely to be included in larger randomly sampled subsets. As a result, the benefits of querying informative instances become less noticeable.



(a) No confounder, $y = 0$ (b) No confounder, $y = 1$



(c) Confounder with a new color, $y = 0$ (d) Confounder with a new color, $y = 1$

Fig. 2: `ER-color` dataset examples with and without confounders. The trees shown are the ground-truth explanations for each class. As confounders, for $y = 0$, a purple node is added to the graph, and for $y = 1$, a cyan node is added.

Table 1: **Effect of passive supervision in `XIGL` across datasets.** Accuracy (%) on the unconfounded test split. Results are averaged with standard deviation across 5 seeds.

	`ER-color`		`CPatchMNIST`	
	No Sup.	**Passive**	**No Sup.**	**Passive**
`GCN`	60.40±11.36	**71.20**±20.73	26.28±12.58	**30.57**±2.90
`GIN`	52.93±4.23	**74.20**±21.18	0.20±0.12	**33.38**±1.13
`GraphSAGE`	56.80±14.24	**70.53**±17.76	**31.5**±6.85	30.68±1.88
`GAT`	50.93±4.21	**74.27**±18.74	22.61±3.93	**25.25**±9.38

4.1 Results

Passive Explanation Supervision Table 1 shows the test accuracy for `ER-color` and `CPatchMNIST` for different setups and GNN architectures, respectively. As the results show, passive supervision generally improves the accuracy compared to no-explanation supervision (No Sup.) and their ability to generalize to unconfounded data. We noticed that passive supervision's performance improves further if we first train the models for a few epochs only on $\mathcal{L}_e$, followed by training on $\mathcal{L}_{ce} + \lambda\mathcal{L}_e$, refer to Table 3 in Appendix A for more details. The results shown in Table 1 under Passive correspond to this setup. As the results show, on `ER-color`, most models perform near the chance level without explanation supervision but improve substantially with explanation supervision.

Table 2: **Effect of AL strategies of XIGL across datasets.** Accuracy (%) on the unconfounded test split. Results are averaged with standard deviation across 5 seeds.

	ER-color				CPatchMNIST			
	Random	**MaCE**	**MaEE**	**MiEE**	**Random**	**MaCE**	**MaEE**	**MiEE**
GCN	58.13±21.07	54.67±9.98	59.87±20.43	**64.00**±13.93	26.52±3.48	22.56±6.21	28.21±2.10	**28.75**±0.73
GIN	**68.80**±22.93	63.87±12.40	66.67±27.13	66.93±19.50	**32.74**±2.80	25.50±3.88	30.87±2.61	27.77±2.77
GraphSAGE	59.87±16.67	**73.20**±23.74	55.20±7.49	56.27±11.90	25.86±2.30	22.57±8.46	26.38±5.25	**28.28**±3.94
GAT	52.67±14.12	**60.40**±22.10	53.87±9.81	50.13±1.79	26.90±2.34	26.59±1.22	**29.20**±11.64	28.57±5.86

This indicates that the models primarily rely on shortcut information and largely ignore the causal patterns, whereas explanation supervision guides them toward the causally relevant features.

On CPatchMNIST, most models perform slightly above chance level even without supervision, indicating that the causal signal can be learned to a limited degree. However, explanation supervision clearly improves performance for all architectures except for GraphSAGE. This effect is particularly evident for GIN, which quickly overfits to the shortcuts and suffers a sharp drop in test accuracy, while explanation supervision in XIGL helps it recover and generalize better. While early stopping can reduce overfitting, it also limits the extent to which reliance on shortcuts becomes apparent. To expose this behavior and evaluate the robustness of explanation supervision, we do not use early stopping.

Active Explanation Supervision Table 2 shows the results of the different query strategies across both datasets and all GNNs. As the results indicate, the effectiveness of the query strategies varies across datasets and models. Nevertheless, active query selection generally outperforms random sampling in most settings. On CPatchMNIST, the explanation-based strategies consistently outperform the prediction-based strategies across all GNNs. Interestingly, random sampling achieves the best performance for GIN in both datasets. We leave a systematic investigation of this effect for future work. Overall, MiEE emerges as the most consistently high-performing query strategy across models and datasets. These results highlight the potential of explanation-based active learning and motivate further investigation into explanation-driven query strategies, particularly those targeting highly confident yet potentially incorrect explanations.

As expected, the active learning component in XIGL generally achieves lower performance than passive learning due to its substantially smaller annotation budget. Nevertheless, the gap is relatively small, despite active learning using less than 20% of the training data. These results suggest that active learning can achieve performance comparable to passive learning while requiring significantly fewer annotated explanations.

5 Discussion, Related Work, & Conclusion

Our work suggests that, by exploiting expert corrections to model explanations, `XIGL` can help remove shortcut dependencies from models, and that it can reduce annotation costs compared to passive baselines. In future work, we plan to extend the experiments to more datasets and GNN architectures, and specifically to self-explainable GNNs [15, 21, 22]. Moreover, we plan to study the impact of explanation faithfulness [17] on debiasing success: if an explanation does not capture the model's actual reasoning, then corrections to it may not help improve the model's behavior [3]. This is relevant for both post-hoc explainers [13] and self-explainable architectures [4]. Additionally, evaluating different kinds of shortcuts and real-world datasets is planned for future work.

Acknowledgments. Taraneh Younesian was funded by Huawei DREAMS Lab. All content represents the opinion of the authors, which is not necessarily shared nor endorsed by their respective employers and/or sponsors in Huawei DREAMS Lab. Antonio Longa was supported by the Research Council of Norway through its Centre of Excellence Integreat - The Norwegian Centre for knowledge-driven machine learning, project number 33264. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Health and Digital Executive Agency (HaDEA). Neither the European Union nor the granting authority can be held responsible for them. Grant Agreement no. 101120763 - TANGO

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

Bibliography

[1] Arjovsky, M., Bottou, L., Gulrajani, I., Lopez-Paz, D.: Invariant risk minimization. arXiv preprint arXiv:1907.02893 (2019)

[2] Azzolin, S., Longa, A., Barbiero, P., Lio, P., Passerini, A.: Global explainability of GNNs via logic combination of learned concepts. In: The Eleventh International Conference on Learning Representations (2023)

[3] Azzolin, S., Longa, A., Teso, S., Passerini, A.: Reconsidering faithfulness in regular, self-explainable and domain invariant gnns. In: International Conference on Learning Representations. vol. 2025 (2025)

[4] Azzolin, S., Teso, S., Lepri, B., Passerini, A., Malhotra, S.: GNN explanations that do not explain and how to find them. In: The Fourteenth International Conference on Learning Representations (2026)

[5] De Luca, V.M., Longa, A., Lio, P., Passerini, A.: xai-drop: Don't use what you cannot explain. In: Learning on Graphs Conference. pp. 16–1. PMLR (2025)

[6] Geirhos, R., Jacobsen, J.H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., Wichmann, F.A.: Shortcut learning in deep neural networks. Nature Machine Intelligence **2**(11), 665–673 (2020)

[7] Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)

[8] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)

[9] Knyazev, B., Taylor, G.W., Amer, M.: Understanding attention and generalization in graph neural networks. Advances in neural information processing systems **32** (2019)

[10] Krueger, D., Caballero, E., Jacobsen, J.H., Zhang, A., Binas, J., PRIOL, R.L., Zhang, D., Courville, A.: Out-of-distribution generalization via risk extrapolation ({re}x) (2021)

[11] Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking clever hans predictors and assessing what machines really learn. Nature communications **10**(1), 1–8 (2019)

[12] Li, D., Wang, Z., Chen, Y., Jiang, R., Ding, W., Okumura, M.: A survey on deep active learning: Recent advances and new frontiers. IEEE Transactions on Neural Networks and Learning Systems **36**(4), 5879–5899 (2024)

[13] Li, X., Wang, J., Yan, Z.: Can graph neural networks be adequately explained? a survey. ACM Comput. Surv. (2025). https://doi.org/10.1145/3711122

[14] Longa, A., Azzolin, S., Santin, G., Cencetti, G., Lio, P., Lepri, B., Passerini, A.: Explaining the explainers in graph neural networks: a comparative study. ACM Computing Surveys **57**(5), 1–37 (2025)

[15] Miao, S., Liu, M., Li, P.: Interpretable and generalizable graph learning via stochastic attention mechanism. In: International conference on machine learning. pp. 15524–15543. PMLR (2022)
[16] Pluska, A., Welke, P., Gärtner, T., MALHOTRA, S.: Logical distillation of graph neural networks. In: ICML 2024 Workshop on Mechanistic Interpretability (2024)
[17] Pope, P.E., Kolouri, S., Rostami, M., Martin, C.E., Hoffmann, H.: Explainability methods for graph convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
[18] Ross, A.S., Hughes, M.C., Doshi-Velez, F.: Right for the right reasons: training differentiable models by constraining their explanations. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 2662–2670 (2017)
[19] Settles, B.: Active learning literature survey (2009)
[20] Song, Z., Zhang, Y., King, I.: No change, no gain: Empowering graph neural networks with expected model change maximization for active learning. Advances in neural information processing systems **36**, 47511–47526 (2023)
[21] Tai, W., Zhong, T., Trajcevski, G., Zhou, F.: Redundancy undermines the trustworthiness of self-interpretable GNNs. In: Forty-second International Conference on Machine Learning (2025), https://openreview.net/forum?id=hFvp9NYfY9
[22] Tai, W., Zhong, T., Trajcevski, G., Zhou, F.: Self-consistency improves the trustworthiness of self-interpretable GNNs. In: The Fourteenth International Conference on Learning Representations (2026), https://openreview.net/forum?id=hxGdAUn3sB
[23] Teso, S., Alkan, Ö., Stammer, W., Daly, E.: Leveraging explanations in interactive machine learning: An overview. Frontiers in Artificial Intelligence (2023)
[24] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018)
[25] Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019)
[26] Yang, Y., Gan, E., Dziugaite, G.K., Mirzasoleiman, B.: Identifying spurious biases early in training through the lens of simplicity bias. In: International conference on artificial intelligence and statistics. pp. 2953–2961. PMLR (2024)
[27] Yuan, H., Yu, H., Gui, S., Ji, S.: Explainability in graph neural networks: A taxonomic survey. IEEE Trans. Pattern Anal. Mach. Intell. p. 5782–5799 (2023). https://doi.org/10.1109/TPAMI.2022.3204236
[28] Zhang, D., Betala, S., Agarwal, C.: Quantifying explanation quality in graph neural networks using out-of-distribution generalization. arXiv preprint arXiv:2602.07708 (2026)
[29] Zhang, J., Katz-Samuels, J., Nowak, R.: Galaxy: Graph-based active learning at the extreme. In: International Conference on Machine Learning. pp. 26223–26238. PMLR (2022)

Table 3: Hyperparameter settings for different GNN architectures for the passive supervision setup on `ER-color` and `CPatchMNIST`. "H" indicated the hidden dimension, "Aggr." indicated the aggregation function of the GNN, "Start E." indicates the starting epoch for training with both losses, and "E". indicated the total training epochs.

	`ER-color`								`CPatchMNIST`							
	# Layers	H	Dropout	Aggr.	LR	λ	Start E.	# E	# Layers	H	Dropout	Aggr.	LR	λ	Start E.	# E
GCN	3	100	0.3	add	$9e-7$	10	100	400	3	128	0.3	add	$3e-6$	19.22	30	500
GIN	3	64	0	mean	$5e-4$	10	50	300	5	256	0	mean	$8e-7$	38.77	30	300
SAGE	3	128	0	add	$6e-6$	1	30	300	5	128	0	add	$3e-6$	71.77	70	500
GAT	3	128	0	add	$3.e-5$	1	0	300	3	64	0	mean	$5e-5$	22.56	0	200

Table 4: Training hyperparameters for different GNN architectures on `ER-color` and `CPatchMNIST`.

	`ER-color`								`CPatchMNIST`							
	Random		MaCE		MaEE		MiEE		Random		MaCE		MaEE		MiEE	
	LR	λ	LR	λ	LR	λ	LR	λ	LR	λ	LR	λ	LR	λ	LR	λ
GCN	$1e-5$	1.09	$3e-5$	1.19	$7e-6$	26.40	$5e-5$	25.21	$6e-7$	251.75	$6e-6$	614.48	$6e-7$	317.17	$8e-7$	446.70
GIN	$2e-4$	29.91	$7e-4$	1.06	$6e-4$	81.08	$2e-4$	2.51	$6e-7$	162.74	$2e-6$	672.56	$3e-6$	606.85	$6e-7$	212.88
SAGE	$5e-5$	23.91	$1e-4$	20.10	$5e-5$	22.23	$8e-5$	57.29	$9e-6$	325.70	$3e-5$	380.74	$2e-5$	850.60	$3e-6$	102.23
GAT	$9e-5$	751.34	$6e-5$	565.93	$5e-5$	565.52	$9e-6$	247.76	$4e-5$	929.88	$2e-5$	130.81	$7e-5$	482.22	$1e-4$	961.77

A GNN Details

In this section, we present the hyperparameters for different GNN architectures across both datasets in the passive supervision scenario. Table 3 shows these values. We set the batch size to 16 and 256 for `ER-color` and `CPatchMNIST`, respectively, across all models and models. We evaluate all methods in a wide range of learning rates: $(1e-7, 1e-3)$ and λ between 1 and 1000. For a fair comparison, we fix the number of epochs for the passive methods and set a fixed value across the active query strategies. As mentioned before, because early stopping can halt training before the effect of shortcuts becomes apparent, we deliberately do not use it. We plan to release the full codebase to reproduce our experiments upon acceptance.

B AL Details

For active learning experiments, we set q to 10 for both datasets. For `ER-color`, we query 5 instances per round for 20 rounds and 30 epochs per round. For `CPatchMNIST`, we query 50 instances per round for 10 rounds and 30 epochs per round. These settings hold for all models. The learning rates and λ (varying in the same range as the passive experiments) for each GNN, query strategy, and dataset combination are shown in Table 4.