WOMBAT: WHITEBOX ORACLE FOR MOLECULAR BENCHMARKING AND ATTRIBUTION TESTING

Anonymous authors
Paper under double-blind review

ABSTRACT

When a graph neural network (GNN) explainer produces an unexpected attribution on a molecule, the attribution alone cannot reveal whether the explainer has failed or the model has learned a shortcut. We introduce WOMBAT, a benchmark of 14 whitebox GNNs, each with message-passing weights set by hand to detect a specific SMARTS motif. Each model's decision rule is known by construction, providing attribution ground truth against which explainer errors can be identified and studied. We validate the models on millions of PubChem molecules and evaluate post-hoc explainers including GNNExplainer, PGExplainer, and Integrated Gradients. Guided by our qualitative analysis, we construct a model that causes Integrated Gradients to spread attribution across the graph, even though the model reliably detects the intended motif. We release the dataset, models, and evaluation code to help researchers in the development of newer XAI tools for GNNs.

1 INTRODUCTION

Graph Neural Networks (GNNs) have emerged as the foundational architecture for modern cheminformatics, enabling progress in molecular property prediction, virtual screening, and the design of novel therapeutics (Zhang et al., 2025). As such, understanding GNN predictions is critical to ensure models rely on true chemical principles rather than spurious shortcuts (Proszewska et al., 2025; Wang et al., 2022). This is why post-hoc XAI methods (Ying et al., 2019; Sundararajan et al., 2017; Luo et al., 2020) are used to identify important graph features. However, verifying their reliability is currently limited by a lack of ground truth explanations for real-world molecular data.

Two common evaluation approaches use task-defined motifs as attribution ground truth (Ying et al., 2019; Proszewska et al., 2025) or perturbation-based metrics such as fidelity (Pope et al., 2019; Zheng et al., 2024). The former assumes that the model uses the intended motif (Faber et al., 2021); the latter may evaluate predictions on out-of-distribution inputs (Hooker et al., 2019).

We introduce WOMBAT (Whitebox Oracle for Molecular Benchmarking and Attribution Testing) as a different way of approaching this problem (Figure 1). WOMBAT consists of 14 manually engineered Whitebox GNNs, each designed to detect a specific, chemically meaningful motif from a subset of the 4860 SMARTS patterns identified in (Klekota & Roth, 2008). By manually setting the weights of the message, aggregation, and update functions, we create whitebox models that do not require training yet achieve 100% accuracy on many validation tasks. While we are not the first to propose whitebox GNNs, we are – to the best of our knowledge – the first to combine whitebox GNNs on real molecules grounded in chemistry (Table 1).

Table 1: Comparison of graph datasets and benchmarks. WOMBAT is the only one that combines designed-to-be-correct MPNNs (to eliminate confusion in explanation evaluations) with real molecular structures. Abbreviations of features and benchmarks are explained in the Related Work section.

Feature	MUTAG	Benzene	Flu.-Carb.	Alk.-Carb.	BA-2Mot.	ShapeG	B-XAIC	Tttm	WOMBAT
Real molecules	✔	✔	✔	✔	✗	✗	✔	✗	✔
Neg. Expl. eval.	✗	✗	✗	✗	✗	✗	✔	✗	✔
Whitebox Models	✗	✗	✗	✗	✗	✗	✗	✔	✔
No. of Tasks	1	1	1	1	1	1	7	3	14

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

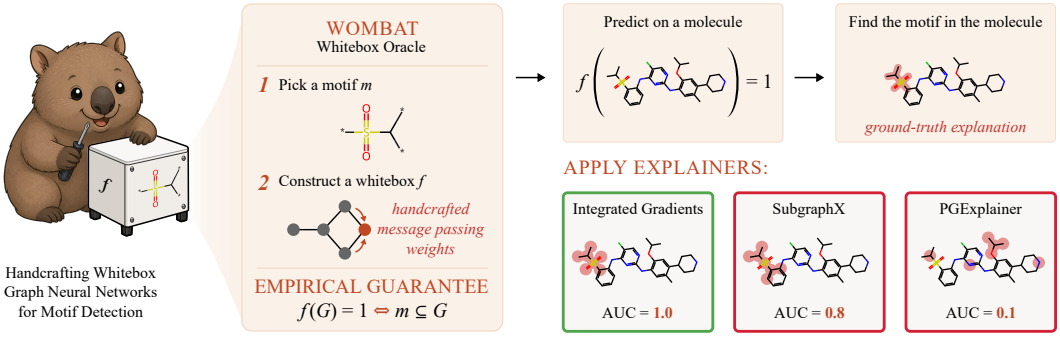


Figure 1: Overview of WOMBAT. For each motif m, we handcraft a whitebox GNN f that detects this motif in the input graph G with $f(G) = 1 \iff m \subseteq G$. This yields a ground truth explanation against which post-hoc explainers can be benchmarked.

This setup provides us with a ground truth for attribution. Since we define exactly which nodes and edges are used for a prediction, discrepancies in an explainer's output cannot be attributed to shortcomings of the underlying model, such as shortcut learning. In turn, WOMBAT can be used not only to benchmark explainability methods, but – much more importantly – as a **diagnostic tool** for improving GNN explainability methods, since any failure can be carefully studied. By applying standard explainers to these oracles, we perform a qualitative analysis which, among other findings, allows us to precisely characterize a failure mode of Integrated Gradients (IG) in WOMBAT and to construct a network that maintains its predictive reliability while causing IG's performance to drastically decline.

Because WOMBAT relies on a deliberately constructed MPNN architecture, we do not claim that the observed failure modes must occur with the same prevalence or must occur at all in other GNN architectures. Rather, WOMBAT provides controlled examples in which such behaviours can be isolated, reproduced, and studied.

In summary, our contributions are:

- We develop an MPNN architecture that supports manual weight-setting, allowing us to create 14 whitebox GNNs designed for the perfectly accurate detection of complex chemical motifs. These MPNNs are meant to be used as diagnostic tools for current and future XAI methods for GNNs.
- We provide a dataset of millions of molecules, sampled from PubChem with a focus on hard negatives using Tversky distance to challenge both models and explainers.
- We perform an evaluation of GNN explainers, identifying their specific failure modes and demonstrating sensitivity of Integrated Gradients and PGExplainer to specific weight configurations.

2 Related Work

GNN Explainability Methods. Post-hoc explainers attempt to identify important nodes, edges, or subgraphs responsible for a model's prediction. Optimization-based methods like GNNExplainer (Ying et al., 2019) and its parameterized successor PGExplainer (Luo et al., 2020) seek to find a compact subgraph that maximizes mutual information. Axiomatic approaches such as Integrated Gradients (IG) (Sundararajan et al., 2017) use path integrals. Inherently interpretable models, e.g. ProtGNN (Zhang et al., 2022), ProGReST (Rymarczyk et al., 2023), and PGIB (Seo et al., 2023), identify subgraphs in the form of prototypes to make an interpretable decision.

Evolution of Graph XAI Benchmarks. Historical benchmarks relied on small, domain-specific datasets with static ground truths. MUTAG (Debnath et al., 1991) and Benzene (McCloskey et al., 2019) require models to detect simple functional groups (e.g., NO_2 or aromatic rings). These were followed by Fluoride-Carbonyl (Flu.-Carb.) and Alkane-Carbonyl (Alk.-Carb.) (Sanchez-Lengeling

et al., 2020), which introduced multi-pattern tasks. However, these datasets are limited by their scale and the "shortcut problem": since the models are trained on statistical correlations, there is no guarantee they utilize the intended motifs for prediction (Agarwal et al., 2023). To address these limitations, synthetic generators such as BA-2Motifs (BA-2Mot.) and ShapeG (Agarwal et al., 2023) were introduced, providing graph generation with implanted motifs. Unfortunately, they lack the chemical fidelity of real molecules. The B-XAIC benchmark (Proszewska et al., 2025) improves this by providing 50,000 molecules across seven tasks with known ground-truth rationales including negative explanations evaluations (Neg. Expl. eval.). Yet, even B-XAIC relies on trained black-box models, which maintain the model-explainer ambiguity, where a failure in attribution could still stem from a model that learned a spurious correlation rather than the explainer's inaccuracy.

Axiomatic Evaluation. A paradigm shift occurred with the Tttm (True-to-the-model) benchmark (Monti et al., 2024; Nerini et al., 2025), which introduced the axiom that an explainer must be truthful to the model's actual logic rather than the data's patterns. Tttm utilizes white-box classifiers with hard-coded logic to audit explainers. However, Tttm focuses on abstract node classification in synthetic graphs. WOMBAT combines the rigor of Tttm with the chemical realism of B-XAIC. By providing 14 manually engineered, designed to be correct MPNNs that detect SMARTS-defined motifs (Klekota & Roth, 2008), WOMBAT establishes a benchmark dedicated to XAI for GNN applications in chemistry.

3 WOMBAT

3.1 Data Preparation

Selection of Motifs. WOMBAT evaluates GNN explainers using 14 hand-crafted whitebox GNNs, each designed to detect a specific molecular motif drawn from a comprehensive set of chemically meaningful SMARTS strings (Klekota & Roth, 2008). We present them in Figure 2. The motifs were hand-selected to cover different substructure sizes; we analyse properties of the patterns in Appendix H.1. Moreover, we restrict our selection to acyclic motifs, due to the well-documented inability of standard Message Passing Neural Networks (MPNNs) to detect specific cyclic substructures in general. Note that this excludes aromatic systems and heterocycles from the set of patterns. This restriction is not arbitrary, however: cycle-based evaluation would likely require either a cycle-aware architecture, graph features that explicitly encode cycle information, or a separate construction. Cycle-related weaknesses of MPNNs have been discussed in the literature before (Chen et al., 2020b; Huang et al., 2022). This restriction applies only to the *motif pool* and not to our dataset (our whiteboxes work properly even if cycles *are* present in the input molecules).

Motif adaptation. WOMBAT consists of substructures with wildcards. We adapt the original SMARTS definitions to better align with MPNN mechanics by relaxing the strict uniqueness requirement for wildcard neighbours (Section H.1). This relaxation ensures the detection task remains tractable, as strictly enforcing neighbour uniqueness is, in general, incompatible with standard message-passing operations. The disagreements between SMARTS and relaxed SMARTS are rare for the majority of patterns, with Patterns 2, 5 and 6 being the exceptions (Table S9). Finally, for Patterns 3 and 5, we relax `[NH]` atoms to `[N;H1,H2]`, allowing one or two attached hydrogens instead of strictly one; this changes neither class labels nor ground truth atom sets across their entire validation sets.

Ground Truths. We label nodes in the union of all motif occurrences under the matching rules above as 1, and all other nodes as 0. This includes neighbours required by wildcard terms, even when these requirements are expressed as degree constraints in the relaxed SMARTS (see Section H.1). Such neighbours contribute through their presence and therefore belong to the ground truth. Molecules without the motif have all-zero labels. Since our whiteboxes are constructed to implement the motif rules and empirically validated against them, our ground truths follow the *true-to-the-model* view of attribution (Chen et al., 2020a).

Whitebox Validation Dataset Creation. Before we test the explainers, we first empirically test the correctness of each of the whiteboxes. To perform this, we extracted all SMILES strings from the PubChem (Kim et al., 2025) database. Then, we tested if they are parsable by RDKit (Landrum et al.), and removed those that failed parsing. To ensure chemical validity, we filtered out molecules violating maximum valence constraints (carbon > 4, nitrogen > 4, oxygen > 2, and sulphur > 6).

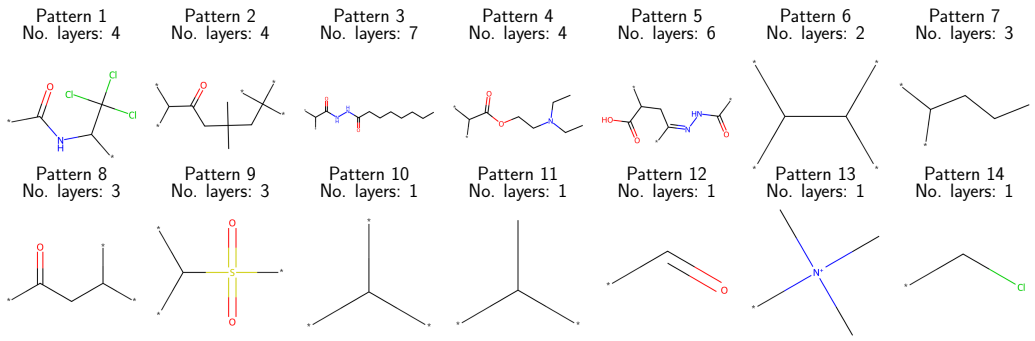


Figure 2: WOMBAT introduces 14 whitebox GNN models for detection of 14 distinct, chemically meaningful motifs. The biggest model, detecting Pattern 3, uses 7 MPNN layers before readout.

For each target motif, we then constructed a dataset by sampling up to 5 million molecules[1] per category:

- **Positive samples:** Randomly selected molecules containing the given motif.
- **Random negatives:** Randomly selected molecules lacking the motif.
- **Hard negatives (Tversky):** Molecules lacking the motif but sharing at least half of the pattern's active ECFP4 fingerprint bits (Tversky index (Tversky, 1977) ≥ 0.5 with $\alpha = 1$, $\beta = 0$).[2] This allows us to focus on molecules that are capable of "mimicking" parts of the motif's local structure.

Finally, we removed duplicated molecules from the resulting dataset. If a SMILES string was sampled as both a random and a hard (Tversky) negative, we prioritized the Tversky negative label. The distributions of distances and similarities across different patterns are detailed in Figure S11.

3.2 WHITEBOX MPNNS DESIGN

To evaluate XAI methods against a verifiable ground truth, we design a series of whitebox MPNNs. The core intuition (see Figure 3) behind this is to transform the standard message-passing network into a logical circuit. **We specify the weight matrices and biases manually**. First, we define every non-hydrogen atom as a graph *node*. Its input representation is a one-hot encoded vector of length 100 representing the atom's atomic number. Then, we append a one-hot encoded vector of length 10, representing the number of hydrogens attached to the atom. We define every bond in the molecule as a graph *edge*. Its input representation is a one-hot encoded vector of length 4, representing the bond's type (single, double, triple, aromatic). The message m_{ij} passed from node v_i to node v_j using MPNN layers with weight matrices W_{from}, W_{to}, and W_e is defined as:

$$m_{ij} = \text{ReLU}(x_i^T W_{\text{from}}) \odot \text{ReLU}(x_j^T W_{\text{to}}) \odot \text{ReLU}(e_{ij}^T W_e) \tag{1}$$

where x_i, x_j, and e_{ij} represent the features of nodes v_i, v_j, and their connecting edge, respectively. Note that $\odot$ denotes element-wise multiplication. To update the node representations, we aggregate the incoming messages via summation, add a bias vector b, and apply a ReLU non-linearity. The updated representation z_i for node v_i in its neighbourhood $\mathcal{N}(v_i)$ is thus computed as:

$$z_i = \text{ReLU}\left(b + \sum_{v_j \in \mathcal{N}(v_i)} m_{ji}\right). \tag{2}$$

[1] or the maximum available, if fewer

[2] Pattern fingerprints were computed by replacing wildcards with carbon atoms and treating the resulting SMARTS as molecules.

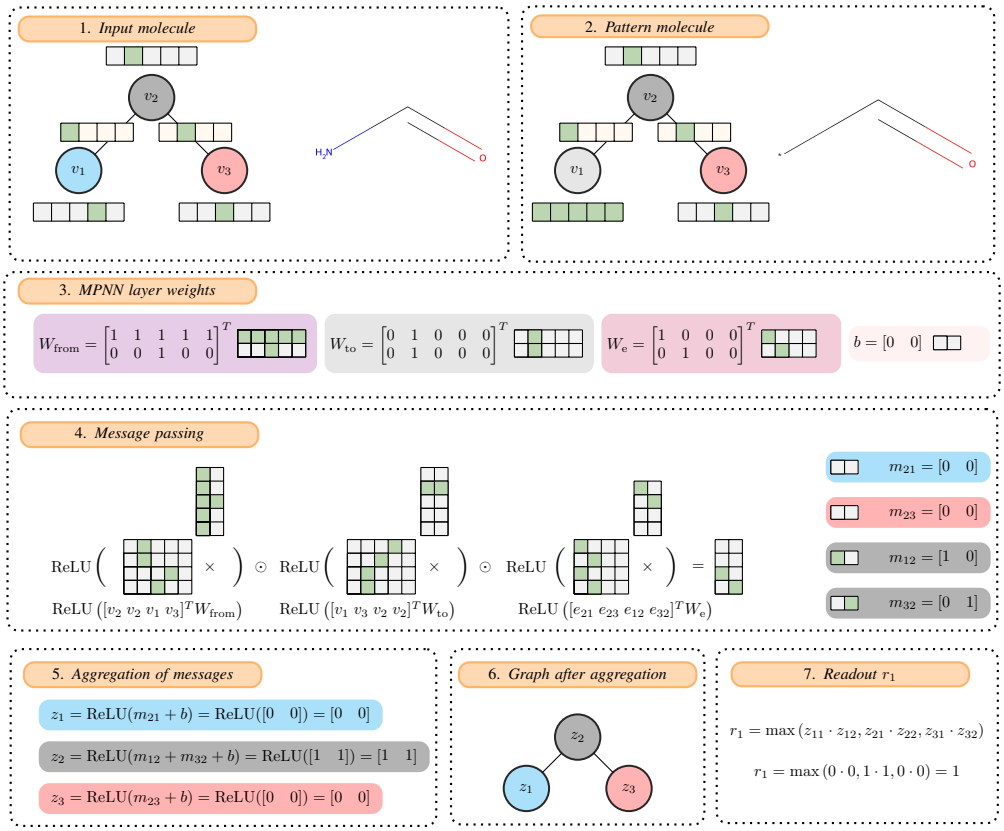


Figure 3: Schematic of the whitebox MPNN architecture and molecular computation flow. The process begins by defining the message-passing weights W_{from}, W_{to}, and W_e; notably, weights corresponding to wildcard tokens in the target motif are initialized to 1. Input graphs are represented via one-hot encoding. Message propagation from node v_i to v_j is computed according to Eq. 1, followed by the update of node representations z_i via Eq. 2 (here, the bias term $b = [0, 0]$). The final prediction, in this case, a value of 1, is obtained through the readout operation defined in Eq. 3. Note that, we only present one type of readout, and the colorcoding denotes the destination of incoming messages: cyan, gray, and red correspond to nodes v_1, v_2, and v_3, respectively.

MPNN Readout. Let V be the set of nodes and let $x_i \in \mathbb{R}^d$ denote the activation vector of node v_i after the final MPNN layer, where d is the output dimensionality. We design the graph-level readout function, r, to act as a structural indicator that yields a positive value if and only if there exists at least one node where all d features are simultaneously "activated" (i.e., strictly positive). We propose two alternative readout formulations to achieve this:

$$r_1 = \max_{v_i \in V} \left(\prod_{j=1}^{d} \text{ReLU}(x_{i,j}) \right) \tag{3}$$

$$r_2 = \text{ReLU}\left(1 - \prod_{v_i \in V} \text{ReLU}\left(1 - \prod_{j=1}^{d} \text{ReLU}(x_{i,j}) \right) \right) \tag{4}$$

where $x_{i,j}$ represents the j-th feature of node v_i. The underlying intuition for both equations is identical: the inner product acts as a logical AND over the d node features, while the outer operation acts as a logical OR across all nodes in the graph. While the first equation is simpler, the $\max$ operator negatively impacts gradient routing (negatively influencing gradient-based explainers). Furthermore, it can unfairly confuse explainers, as its output values are not bounded by 1 and can be arbitrarily large. The second equation addresses this by utilizing De Morgan's laws applied to continuous values, simulating the global OR without relying on $\max$ pooling.

Experimental Validation. We evaluate the accuracy of our constructed whitebox MPNNs with readouts r_1 and r_2 using the validation dataset (detailed in Section 3.1). Across millions of tested molecules, the models achieved perfect classification performance[3]

Masking semantics. In WOMBAT, nodes can be effectively masked by setting their features to 0. That is because (as long as all biases are not positive[4]) a node with its features set to zero can, effectively, neither receive nor send any messages due to the message definition – meaning the calculation is equivalent to the calculation as if it was not present altogether. This property is useful for ensuring fair evaluation of attribution methods that rely on a predefined baseline, such as Integrated Gradients (Sundararajan et al., 2017), as the trivial zero-baseline relying on setting all features to vectors of zeroes is guaranteed to result in the model predicting 0.

3.3 Evaluation of Attribution Methods

Dataset & metrics. We construct the XAI evaluation dataset by randomly sampling up to 10,000 positive and 10,000 negative molecules from the whitebox validation set (Section 3.1), using all available molecules if a pool size is below this threshold. The negative subset is stratified: we first sample up to 5,000 hard negatives from the Tversky pool, and then draw from the random negative pool until the 10,000-molecule target is reached. Because our whitebox models are manually constructed and detect specific motifs, the exact ground-truth subgraphs can be directly extracted using their corresponding SMARTS strings. The faithfulness of node-level attributions is assessed using metrics dependent on the presence of the target motif:

- **Positive Samples:** We measure the explainer's ability to prioritize motif nodes over non-motif nodes per molecule using the AUROC and Average Precision (AP). We report the macro-average of these metrics across all positive graphs.[5] We discuss properties of AUROC in relation to Fidelity in Appendix C.
- **Negative Samples:** Following Proszewska et al. (2025), we evaluate attribution uniformity on graphs lacking the motif. We compute the interquartile range ($\mathrm{IQR} = Q_3 - Q_1$) of the node attribution scores. An explainer successfully processes a negative graph if no node receives an outlier attribution score—defined as falling outside the interval $[Q_1 - 1.5 \cdot \mathrm{IQR}, Q_3 + 1.5 \cdot \mathrm{IQR}]$. We report the aggregate success rate across all negative graphs.

Edge-to-Node Attribution Conversion. While our benchmark evaluates node-level attributions, certain methods (e.g., PGExplainer (Luo et al., 2020)) exclusively generate edge-level importance scores. Because any edge belonging to a motif strictly implies that its incident nodes also belong to that motif, we project edge scores onto the nodes. Specifically, if native node-level explanations are unavailable, we compute a node's attribution by summing the attribution scores of all its incident edges.

4 Experiments and Results

4.1 Quantitative results

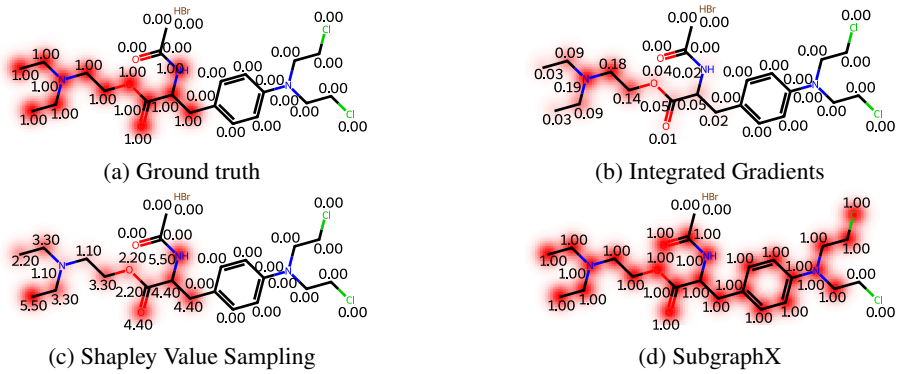
Because the WOMBAT benchmark guarantees that any deviation from the ground truth comes from the explainer's attribution error, we focus on whether an explainer achieved a perfect AUROC score for a given whitebox in Table 2. We also summarize the attribution performance of each explainer across all whitebox models by reporting the mean AUROC and mean Average Precision (AP) in Tables S7 and S8, respectively. We share details of the experimental setup in Appendix B.

We observe that gradient-based methods (such as IG, Input $\times$ Gradient, and Saliency) and SHAP Sampling significantly outperform the perturbation-based GNNExplainer and PGExplainer, which

[3], Zero false positives and zero false negatives at a prediction threshold of 0.5. For every motif evaluated, the models were tested against a set of strictly more than 5 million negative samples (see Table S10).

[4]None of our main whiteboxes have positive biases. We demonstrate one whitebox with this property in Appendix F.3, but it's not a part of our main testing suite.

[5]Molecules composed entirely of the target motif are excluded, as AUROC is undefined in the absence of negative (non-motif) nodes.



(a) Ground truth (b) Integrated Gradients

(c) Shapley Value Sampling (d) SubgraphX

Figure 4: Attributions generated by different explainers when evaluated on the whitebox task of recognizing Pattern 4. Other explainers (i.e., PGExplainer, GNNExplainer, Saliency, and Input $\times$ Gradient) produced null explanations. Integrated Gradients and Shapley Value Sampling attribute all motif nodes higher than non-motif nodes, whereas SubgraphX attributes importance to multiple non-motif nodes.

Table 2: Explainer performance across different motifs. We observe that IG is the most successful in this scenario. Moreover, we can see that a change of a readout function directly impacts the performance of IG. A checkmark indicates the explainer achieved a perfect score ($\mathrm{AUROC} = 1$) for all positive instances of the given pattern. The sub-columns r_1 and r_2 denote the readout functions utilized by the whitebox models, corresponding to Eq. 3 and Eq. 4, respectively.

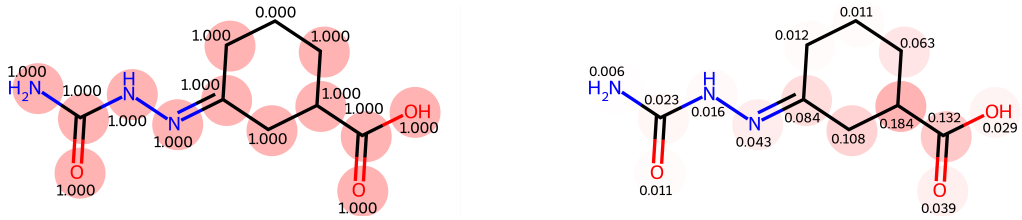
Pattern	GNN Expl.		Input $\times$ Grad.		IG		PG Expl.		SHAP Sampl.		Saliency		SubgraphX	
	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2
1	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
2	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
3	✗	✗	✔	✗	✔	✔	✗	✗	✗	✗	✔	✗	✗	✗
4	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
5	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
6	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
7	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
8	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
9	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
10	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
11	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
12	✗	✔	✗	✗	✗	✔	✗	✗	✔	✔	✗	✗	✗	✗
13	✗	✗	✗	✗	✗	✔	✗	✗	✗	✗	✗	✗	✗	✗
14	✗	✗	✗	✗	✗	✔	✗	✗	✔	✔	✗	✗	✗	✗

largely fail to exceed random baseline performance across most motifs. Second, despite the high average metrics of top-performing explainers, Table 2 highlights that achieving strictly perfect explanations remains challenging, with only IG under r_2 demonstrating semi-consistent perfection. Finally, all evaluated explainability methods, with the exception of PGExplainer, exhibited ideal performance on negative samples (i.e., molecules lacking the target motif). PGExplainer failed this evaluation in exactly one instance out of the 140,000 examples tested, a behavior observed in the case of both readout functions. This isolated failure occurred for Pattern 12, on which PGExplainer does not collapse to $\mathrm{AUROC} = 0.5$. The failure is caused by PGExplainer learning a heuristic to detect Pattern 12; this heuristic fails for a very specific class of edge cases, causing misattributions. We illustrate these edge cases in Figure S6 and discuss their causes in Appendix E.2.

4.2 Failure Modes of Explainers

In this part of the work we present failure modes for Gradient-based methods and PGExplainer, while the analysis for GNNExplainer and Shapley Value Sampling is provided in the Appendix Section E. Representative attributions across explainers are shown in Figure 4.

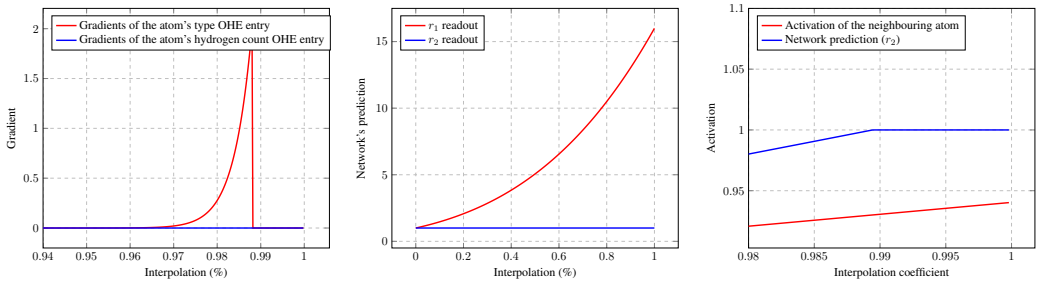
Gradient-based methods. Saliency and Input $\times$ Gradient perform well on the max-pooling readout (r_1). However, r_1 restricts gradient routing: when a molecule contains multiple motif instances,



(a) Ground truth attribution

(b) Integrated Gradients attribution

Figure 5: Comparison of attribution generated by Integrated Gradients and the ground truth. Integrated Gradients is unfaithful to the model by assigning higher attribution to the non-motif carbon in the cyclohexane ring than to the NH_2 functional group.



(a) Gradients of the non-motif atom during IG interpolation.

(b) Network's prediction during interpolation of the non-motif atom (other atoms and edges are not scaled ($\alpha = 1.0$))

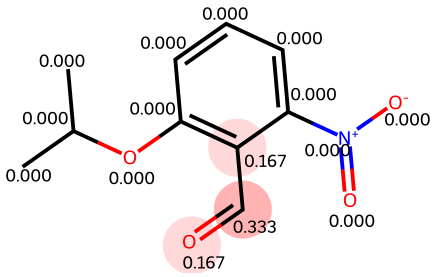
(c) Interpolation of the non-motif atom while other atoms and edges are interpolated at $\alpha = 0.98$.

Figure 6: Analysis of causes of spurious attribution of the non-motif atom in Pattern 5. Figure 6a shows gradients that are accumulated by this atom near the decision boundary. Figure 6b demonstrates that under readout r_2 the atom does not impact the network's final prediction, regardless of its interpolation state. However, it increased the network's output under readout r_1, as it increases internal activations. Figure 6c shows the effect of interpolating the non-motif atom on activation of a neighbouring node and final network's prediction under readout r_2, providing a justification for curve observed in the Figure 6a.

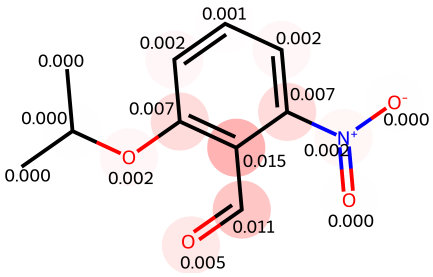
gradients are directed to only one substructure, leaving others without attribution. Under the r_2 readout, both methods collapse, a direct consequence of violating the sensitivity axiom (Sundararajan et al., 2017). As a result, IG demonstrates an improvement under r_2.

Integrated Gradients. Under the r_2 readout, there are two molecules for which IG is unfaithful to the model recognising Pattern 5. We demonstrate one of these cases in the Fig. 5. Here, a non-motif carbon receives a higher attribution than some atoms belonging to the pattern. However, it has no effect on the final prediction under r_2. Interpolating its features from the zero baseline leaves the r_2 prediction unchanged throughout the entire integration path (Fig. 6b). Nevertheless, the atom increases the network's internal activations, as evidenced by its effect on the intermediate r_1 readout. Consequently, there exists a region along the integration path, near the model's decision threshold, where further scaling this atom's representation increases the network's activation and yields positive local gradients (Fig. 6a, Fig. 6c). Since Integrated Gradients accumulates gradients along the integration path, these transient positive gradients result in a positive attribution for the non-motif atom. Similar issues happen for Pattern 4, where IG yields imperfect scores frequently (failure rate of approximately 15%). We present our full analysis in Appendix D.

PGExplainer. PGExplainer's failure can be partially attributed to the fact that it bases edge attributions based on final activations of the edge's terminals. In WOMBAT, many nodes (including pattern nodes) may not be activated in the final layer, significantly capping potential PGExplainer's performance in case of the larger substructures (such as Pattern 3 or Pattern 4) and causing PGExplainer to learn a heuristic in case of the Pattern 12. We provide more details in Appendix E.



(a) IG attribution for the Pattern 12 whitebox (regular version).



(b) IG attribution for the anti-IG Pattern 12 whitebox. Note the "leaky" attribution.

Figure 7: IG attributions for two Pattern 12 implementations with identical r_2 predictions on one-hot molecular inputs. Here, the regular implementation yields perfect AUROC, while redundant computations involving neighbours cause attribution spillover.

5 Discussion

WOMBAT isolates explainer failures on models whose decision rules are known. It establishes that a failure *can* occur and helps explain *why*; however, its *frequency* in trained GNNs remains an open question. Changing the architecture (or even weights) can impact the quality of explanations. We demonstrate this in Appendix F.2, where we create a more "idiomatic" MPNN (using GINEConv architecture (Hu et al., 2019)) and in Appendix F.3, where we show that different implementations of whiteboxes for Pattern 12 can yield wildly different explanations, including significantly worsening IG's performance by "blurring" its attribution. WOMBAT therefore complements benchmarks that use trained models and focus on quantitative evaluation, such as Proszewska et al. (2025); Sanchez-Lengeling et al. (2020).

The collapse of Saliency and Input × Gradient under r_2 illustrates the known limitation of relying on local gradients, which can vanish despite the prediction depending on the input (Sundararajan et al., 2017). Integrated Gradients (IG) generally performs well on WOMBAT, but still can attribute non-motif atoms as well; this is caused by transient gradients along its interpolation path. While the existence of path artifacts is known (Miglani et al., 2020; Sanyal & Ren, 2021; Enguehard, 2023; Kapishnikov et al., 2021), WOMBAT lets us trace their causes through interpretable weights. We hypothesise that they can be stronger in MPNNs whose message aggregation admits more non-motif contributions than WOMBAT's gated messages. We create an adversarial whitebox (Appendix F.3) which shows that additional propagation of redundant messages can amplify them (see Figure 7), but does not establish their prevalence in trained models.

PGExplainer attributes edges based on their endpoints' final representations. In WOMBAT, many motif and non-motif nodes have final representations equal to zero, making some relevant and irrelevant edges indistinguishable to the PGExplainer. We hypothesise that oversmoothing (Rusch et al., 2023) can worsen this problem when it removes distinctions needed for explanation. We furthermore demonstrate changes of weights for Pattern 12 that "help" PGExplainer learn to explain networks properly and changes that impede it from doing so in Appendix F.3.

6 Conclusions

We introduced WOMBAT, a benchmark of 14 hand-crafted whitebox GNNs for acyclic molecular motif detection, validated on millions of PubChem molecules. Their known decision rules allow attribution errors to be isolated and studied without uncertainty about the model's behaviour. Evaluating seven XAI methods, we traced failures to specific computations and used those findings to construct whiteboxes that amplify them. We release the data, models and code to support further diagnostic studies and the development of better graph explainers. WOMBAT makes even rare failures useful: understanding *why* an explanation fails provides a concrete target for improving it.

AI USE STATEMENT

In this work, we used generative AI tools to help rephrase or draft parts of the manuscript and to provide feedback on unfinished versions, including identifying potential errors in mathematical arguments and proofs. We also used generative AI tools to provide feedback on experimental methodology and the interpretation and presentation of results. Additionally, generative AI tools were used to assist with writing and editing source code, including TikZ code for illustrations. Furthermore, AI systems were used to assist with information and literature searches. We have not used generative AI tools for unsupervised or agentic code generation, and all whiteboxes in the main testing suite had all their weights set by a human[6]. We have not used LLMs for hypothesis generation. We have reviewed AI-assisted work: autocomplete suggestions were immediately verified by the author responsible for source code, while LLM-generated alerts about potential manuscript errors were reviewed and fixed by humans, if appropriate. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT.

We provide open-source code (`https://anonymous.4open.science/r/wombat/`) for the entire evaluation pipeline, including all whitebox weights. The full WOMBAT dataset cannot be linked during double-blind review because it is too large to include in the anonymized repository, while its existing hosting would reveal author identity; a sample of the Pattern 1 validation dataset is included in the repository. The dataset will be released under the CC0 1.0 license and the code under the MIT license.

REFERENCES

Chirag Agarwal, Owen Queen, Himabindu Lakkaraju, and Marinka Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

Hugh Chen, Joseph D Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data? *arXiv preprint arXiv:2006.16234*, 2020a.

Zhengdao Chen, Lei Chen, Soledad Villar, and Joan Bruna. Can graph neural networks count substructures? *Advances in neural information processing systems*, 33:10383–10395, 2020b.

Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry*, 34(2):786–797, 1991.

Joseph Enguehard. Sequential integrated gradients: a simple but effective method for explaining language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7555–7565, 2023.

Lukas Faber, Amin K. Moghaddam, and Roger Wattenhofer. When comparing to ground truth is wrong: On evaluating gnn explanation methods. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, pp. 332–341, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3467283. URL `https://doi.org/10.1145/3447548.3467283`.

Matthias Fey, Jinu Sunil, Akihiro Nitta, Rishi Puri, Manan Shah, Blaz Stojanovic, Ramona Bendias, Barghi Alexandria, Vid Kocijan, Zecheng Zhang, Xinwei He, Jan E. Lenssen, and Jure Leskovec. PyG 2.0: Scalable learning on real world graphs. In *Temporal Graph Learning Workshop @ KDD*, 2025.

Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32, 2019.

[6]The only whitebox with LLM generated weights was created as a test to check if LLMs are capable of creating WOMBAT whiteboxes. It was not used to test XAI methods, but is available and clearly marked in the source code.

Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. *arXiv preprint arXiv:1905.12265*, 2019.

Yinan Huang, Xingang Peng, Jianzhu Ma, and Muhan Zhang. Boosting the cycle counting power of graph neural networks with I^2-gnns. *arXiv preprint arXiv:2210.13978*, 2022.

Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5050–5058, 2021.

Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem 2025 update. *Nucleic acids research*, 53(D1):D1516–D1525, 2025.

Justin Klekota and Frederick P Roth. Chemical substructures that enrich for biological activity. *Bioinformatics*, 24(21):2518–2525, 2008.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

Greg Landrum, Julie Penzotti, et al. Rdkit. `https://github.com/rdkit/rdkit/`.

Meng Liu, Youzhi Luo, Limei Wang, Yaochen Xie, Hao Yuan, Shurui Gui, Haiyang Yu, Zhao Xu, Jingtun Zhang, Yi Liu, Keqiang Yan, Haoran Liu, Cong Fu, Bora M Oztekin, Xuan Zhang, and Shuiwang Ji. DIG: A turnkey library for diving into graph deep learning research. *Journal of Machine Learning Research*, 22(240):1–9, 2021. URL `http://jmlr.org/papers/v22/21-0343.html`.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in neural information processing systems*, 33:19620–19631, 2020.

Kevin McCloskey, Ankur Taly, Federico Monti, Michael P Brenner, and Lucy J Colwell. Using attribution to decode binding mechanism in neural network models for chemistry. *Proceedings of the National Academy of Sciences*, 116(24):11624–11629, 2019.

Vivek Miglani, Narine Kokhlikyan, Bilal Alsallakh, Miguel Martin, and Orion Reblitz-Richardson. Investigating saturation effects in integrated gradients. *arXiv preprint arXiv:2010.12697*, 2020.

Corrado Monti, Paolo Bajardi, Francesco Bonchi, André Panisson, and Alan Perotti. A true-to-the-model axiomatic benchmark for graph-based explainers. *Transactions on Machine Learning Research*, 2024.

Francesco Paolo Nerini, Francesco Bonchi, and André Panisson. A true-to-the-model benchmark for edge-level attributions of gnn explainers. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 78–93. Springer, 2025.

Phillip E Pope, Soheil Kolouri, Mohammad Rostami, Charles E Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10772–10781, 2019.

Magdalena Proszewska, Tomasz Danel, and Dawid Rymarczyk. B-xaic dataset: Benchmarking explainable ai for graph neural networks using chemical data. *arXiv preprint arXiv:2505.22252*, 2025.

T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

Dawid Rymarczyk, Daniel Dobrowolski, and Tomasz Danel. Progrest: Prototypical graph regression soft trees for molecular property prediction. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 379–387. SIAM, 2023.

Benjamin Sanchez-Lengeling, Jennifer Wei, Brian Lee, Emily Reif, Peter Wang, Wesley Qian, Kevin McCloskey, Lucy Colwell, and Alexander Wiltschko. Evaluating attribution for graph neural networks. *Advances in neural information processing systems*, 33:5898–5910, 2020.

Soumya Sanyal and Xiang Ren. Discretized integrated gradients for explaining language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10285–10299, 2021.

Sangwoo Seo, Sungwon Kim, and Chanyoung Park. Interpretable prototype-based graph information bottleneck. *Advances in Neural Information Processing Systems*, 36:76737–76748, 2023.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

Amos Tversky. Features of similarity. *Psychological review*, 84(4):327, 1977.

Qianwen Wang, Kexin Huang, Payal Chandak, Marinka Zitnik, and Nils Gehlenborg. Extending the nested model for user-centric xai: A design study on gnn-based drug repurposing. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):1266–1276, 2022.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32, 2019.

Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(5): 5782–5799, 2022.

Odin Zhang, Haitao Lin, Xujun Zhang, Xiaorui Wang, Zhenxing Wu, Qing Ye, Weibo Zhao, Jike Wang, Kejun Ying, Yu Kang, et al. Graph neural networks in modern ai-aided drug discovery. *Chemical Reviews*, 125(20):10001–10103, 2025.

Zaixi Zhang, Qi Liu, Hao Wang, Chengqiang Lu, and Cheekong Lee. Protgnn: Towards self-explaining graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 9127–9135, 2022.

Xu Zheng, Farhad Shirani, Tianchun Wang, Wei Cheng, Zhuomin Chen, Haifeng Chen, Hua Wei, and Dongsheng Luo. Towards robust fidelity for evaluating explainability of graph neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 12250–12275, 2024.

APPENDIX TABLE OF CONTENTS

A Limitations **13**

B Experimental setup **14**

C AUROC-Fidelity relationship **14**

C.1 Preliminaries 14

C.2 AUROC-Fidelity relationship 15

C.3 Limitations of AUROC-based evaluation 16

D Causes of misattribution in Integrated Gradients **16**

D.1 Notation 17

D.2 Pattern 4 17

D.3 Pattern 5 19

D.4 Preliminaries 20

D.5 Integrated Gradients in Pattern 4 21

D.6 Integrated Gradients in Pattern 5 24

E Additional Results on Failure Modes of Shapley Value Sampling, PGExplainer, and GNNExplainer **25**

E.1 Shapley Value Sampling 25

E.2 PGExplainer 25

E.3 GNNExplainer 29

F Impact of Whitebox Architecture **30**

F.1 Decision boundary sharpness 30

F.2 GINEConv whitebox experiment 30

F.3 Differing implementations 33

G Additional experimental results **39**

H Additional dataset information **42**

H.1 Patterns 42

H.2 Validation molecules 42

A LIMITATIONS

While our whitebox models have been validated empirically across millions of molecules without a single false positive or false negative, we have not formally verified them using mathematical proof assistants (e.g., Lean). However, we consider our large-scale empirical validation sufficient for the purposes of this benchmark. Furthermore, we **strongly caution against using WOMBAT primar-**

ily as a leaderboard or to declare that "Explainer X is better than Explainer Y". Because WOMBAT guarantees that any attribution discrepancies cannot be explained by unknown model behavior such as shortcut learning, **the benchmark is best utilized for case-by-case diagnostic studies**. For instance, even if a method such as Integrated Gradients achieves near-perfect attributions for 99.99% of compounds under a specific readout, WOMBAT isolates the remaining 0.01% failure cases, allowing researchers to study the exact structural or mathematical conditions that cause the explainer to break down. Furthermore, WOMBAT does not and cannot make any claim about *prevalence* of any identified failure modes in more idiomatic GNNs and rather shows that a weakness is present. That weakness then needs to be judged on case-by-case basis as to its probable severity in more idiomatic GNNs. Finally, our evaluation has focused on whether XAI methods properly rank nodes: while this is critical for an XAI method to work well, other considerations (such as ease of setting appropriate threshold) exist in downstream settings; we discuss these limitations further in Appendix C.3.

B Experimental setup

PGExplainer and GNNExplainer. We use PGExplainer's and GNNExplainer's implementations from PyTorch Geometric library (Fey et al., 2025), adopting default hyperparameters for qualitative analysis. PGExplainer is trained on all molecules, randomly shuffled each epoch, using the model's predictions as targets. Because PGExplainer yields only edge-level attributions, we map these to node-level importance following the procedure in Section 3.3. For both explainers, we set `mode="regression"` and `return_type="raw"` to accommodate the unbounded outputs of the r_1 readout.

Gradient-based explainers & Shapley Value Sampling. Using Captum (Kokhlikyan et al., 2020), we compute node and edge attributions via Integrated Gradients, Input $\times$ Gradient, Saliency and Shapley Value Sampling. We evaluate only node-level results, aggregated by summing the absolute values of each node's 110-dimensional feature vector. We use default settings, with the exception of SVS (raising n_{samples} to 100). Due to high computational complexity, evaluation is restricted to a maximum of 100 molecules per class (positive and negative).

SubgraphX. Using the DiG library (Liu et al., 2021), we apply SubgraphX with default settings. Since SubgraphX identifies coalitions rather than individual node scores, we assign binary attributions: 1 if a node is in the coalition and 0 otherwise. We omit results for negative explanations, as SubgraphX provides coalitions by design. Due to high computational complexity, evaluation is restricted to a maximum of 100 molecules.

C AUROC-Fidelity relationship

C.1 Preliminaries

Assumptions. In this section, we will focus on readout r_2, which has better theoretical properties and does not unfairly confuse explainers like the r_1 readout. Unless otherwise noted, in this section we assume all models are using r_2 readout. Furthermore, it can be observed that r_2 readout effectively binarizes network's output. This was also checked during the empirical validation, where there was no molecule for which a model using r_2 readout would return logit not being equal to 1 or 0[7].

Whiteboxes. Whitebox models in WOMBAT were designed to be correct, which was later confirmed by extensive empirical evaluation on known molecules. In other words, we have empirically validated that for a graph G, satisfying our predefined valence rules, and motif m, whitebox model f_m (under the readout r_2) satisfies the following property:

$$f_m(G) = 1 \iff m \subseteq G \tag{5}$$

[7] In practice, such situation happens only if the input features are scaled (i.e. the input features are not one-hot encoded, but are scaled one-hot encoding). In this section, however, we are working with Fidelity metric, which does not *scale* input features, but completely masks them.

Ground truth. As stated in Section 3.1, WOMBAT's ground truth is the union of the node sets of all motif occurrences. This includes nodes matched by wildcard terms. Where wildcard terms are replaced by degree constraints in the relaxed SMARTS, we include all neighbours of the matched atoms subject to those constraints, since their presence is required to satisfy them.

Fidelity. Let $G_s \subseteq G$ be a subgraph of G that constitutes an explanation of f_m. Let $G \setminus G_s$ be complement of G_s. Then, we define *fidelity* of G_s as follows (Yuan et al., 2022):

$$\text{Fidelity}^{+\text{prob}}(G, G_s) = f_m(G) - f_m(G \setminus G_s) \tag{6}$$

which intuitively measures whether removal of G_s from the input changes model's prediction; the greater the Fidelity^{+}, the better G_s serves as model's explanation. Alternatively, fidelity can be defined as follows:

$$\text{Fidelity}^{-\text{prob}}(G, G_s) = f_m(G) - f_m(G_s) \tag{7}$$

which intuitively measures whether removal of all nodes *not* belonging to G_s changes model's prediction. In that case, the lower the Fidelity^{-}, the better G_s serves as model's explanation.

Sparsity. Let $G_s \subseteq G$ be a subgraph of G that constitutes an explanation of f_m. Let $G \setminus G_s$ be complement of G_s. Then, we define *sparsity* of G_s as follows (Yuan et al., 2022):

$$\text{Sparsity}(G, G_s) = 1 - \frac{|V_{G_s}|}{|V_G|} \tag{8}$$

C.2 AUROC-Fidelity relationship

Theorem 1. Let f_m be a WOMBAT whitebox model with readout r_2. Let $G = (V, E)$ be a graph, such that $m \subsetneq G$. Let $G_{\text{GT}} \subsetneq G$ be ground truth explanation of f_m on G. Then, G_{GT} maximises $\text{Fidelity}^{+\text{prob}}$ and minimises $\text{Fidelity}^{-\text{prob}}$.

Proof. Recall that:

$$f_m(G) = 1 \iff m \subseteq G \tag{9}$$

Since G_{GT} is union of all occurrences of motif m in G:

$$\text{Fidelity}^{+\text{prob}}(G, G_{\text{GT}}) = f_m(G) - f_m(G \setminus G_{\text{GT}}) = 1 - 0 = 1 \tag{10}$$

and

$$\text{Fidelity}^{-\text{prob}}(G, G_{\text{GT}}) = f_m(G) - f_m(G_{\text{GT}}) = 1 - 1 = 0 \tag{11}$$

Furthermore, note that since we are using readout r_2, it follows that for any G, $f_m(G) \in \{0, 1\}$ (as long as the input features are not scaled, which we assume to be the case). This trivially implies that the maximum of $\text{Fidelity}^{+\text{prob}}$ is 1 and minimum $\text{Fidelity}^{-\text{prob}}$ is 0. In other words, ground truth explanation graph maximises/minimises relevant fidelities. □

Theorem 2. Let f_m be a WOMBAT whitebox model with readout r_2. Let $G = (V, E)$ be a graph, such that $m \subsetneq G$. Let $G_{\text{GT}} \subsetneq G$ be ground truth explanation of f_m on G and let $a_1, a_2, \ldots, a_n$ be ground truth attributions of nodes $v_1, v_2, \ldots, v_n$, respectively, where $a_i = 1 \iff v_i \in G_{\text{GT}}$ and $a_i = 0 \iff v_i \notin G_{\text{GT}}$. Then let $\hat{a_1}, \hat{a_2}, \ldots, \hat{a_n}$ be attributions generated by some attribution method. Then, if:

$$\text{AUROC}([\hat{a_1}, \ldots, \hat{a_n}]\,;\, [a_1, \ldots, a_n]) = 1$$

there exists $t \in \mathbb{R}$ such that there exists $V_s = \{v_i : \hat{a_i} \geq t\}$ for which exists $G_s = G[V_s]$, such that G_s maximises $\text{Fid}^{+\text{prob}}$ and minimises $\text{Fid}^{-\text{prob}}$.

Proof. By definition of AUROC, if $\mathrm{AUROC}([\hat{a_1}, \ldots, \hat{a_n}]\ ;\ [a_1, \ldots, a_n]) = 1$, then:

$$\max_{i:\ a_i=0} \hat{a_i} < \min_{i:\ a_i=1} \hat{a_i}$$

Hence, there exists $t \in \mathbb{R}$ such that:

$$t \in \left(\max_{i:\ a_i=0} \hat{a_i}\ ;\ \min_{i:\ a_i=1} \hat{a_i} \right]$$

Then we can define V_s:

$$V_s = \{v_i : \hat{a_i} \geq t\}$$

and observe that $v_i \in V_s \iff \hat{a_i} \geq t \iff a_i = 1 \iff v_i \in G_{\mathrm{GT}}$. Hence, $G[V_s] = G_{\mathrm{GT}}$, which maximises $\mathrm{Fid}^{+\mathrm{prob}}$ and minimises $\mathrm{Fid}^{-\mathrm{prob}}$ (Theorem 1). □

Theorem 3. Let f_m be a WOMBAT whitebox model with readout r_2. Let $G_{\mathrm{GT}} \subsetneq G$ be ground truth explanation of f_m on G. Let there only be one occurrence of motif m in G. Then, G_{GT} maximises sparsity among all possible explanations that minimise $\mathrm{Fidelity}^{-\mathrm{prob}}$.

Proof. Note that under assumption that there is only one motif m present in G, it follows that $G_{\mathrm{GT}} = m$. Assume that there exists some G' such that:

$$\mathrm{Fid}^{-\mathrm{prob}}(G, G') = 0$$

and

$$\mathrm{Sparsity}(G, G') > \mathrm{Sparsity}(G, G_{\mathrm{GT}})$$

Since $\mathrm{Sparsity}(G, G') > \mathrm{Sparsity}(G, G_{\mathrm{GT}})$, it follows that $|V_{G'}| < |V_{G_{\mathrm{GT}}}|$. Hence, there exists a node $v \in G$ such that $v \in G_{\mathrm{GT}} = m$ but $v \notin G'$. However:

$$\exists_{v \in m}\ v \notin G' \implies m \not\subseteq G' \implies f_m(G') = 0$$

This implies that $\mathrm{Fid}^{-\mathrm{prob}}(G, G') = f_m(G) - f_m(G') = 1 - 0 = 1$, leading to a contradiction. □

Note that Theorem 3 does *not* necessarily hold in the multi-motif case, as attributing only one of the motifs could be more sparse than attribution created by the ground truth. Attributing only one motif, would, however, not maximise $\mathrm{Fid}^{+\mathrm{prob}}$: optimal attribution (sparsity-wise) that would maximise/minimise relevant fidelities would likely need to contain one "full" motif and attribute singular nodes belonging to other occurrences of m in the graph. We believe, however, such attribution to be unintuitive.

C.3 Limitations of AUROC-based evaluation

Our evaluation has focused on whether the explainer ranks nodes correctly, as we believe incorrect ranking to be of fundamental significance. That said, achieving $\mathrm{AUROC} = 1.0$ by itself is not enough for an explainer to be regarded as *perfect*. While $\mathrm{AUROC} = 1$ guarantees that there exists a threshold that perfectly separates ground-truth from non-ground-truth nodes, it does not quantify calibration of attribution scores or margins between them. In downstream applications the ground truth is unknown, making it harder to identify an "importance" threshold. Therefore, the results in this section should be interpreted as statements about the existence of fidelity-optimal thresholded masks, rather than about the quality of attribution magnitudes.

D Causes of misattribution in Integrated Gradients

In this section, we provide implementation details of whiteboxes recognizing Pattern 4 and Pattern 5. Then, we study exact failure modes of Integrated Gradients in these networks.

D.1 NOTATION

In our architecture, each node is represented by a latent vector $z_i^{(\ell)} \in \mathbb{R}^{d_i}$ after passing through the ℓ-th layer. We adopt a notation where each dimension is represented by a letter (A through G). A node is considered **active** for a given feature at layer ℓ if its activation at the corresponding vector index exceeds a threshold of 0.5. For example, if there is a node that after 3rd MPNN layer has 5th element of its latent vector activated, we will say that this node is E3-activated. We adopt this notation, as presenting whitebox weights without any intuition would be highly impractical.

For brevity, we define the relational operator $X \sim Y$ to mean: *An X-active node connected via a single bond to at least one Y-active node.*, as this is a pattern by which majority of our features are defined.

D.2 PATTERN 4

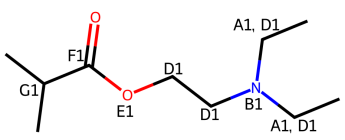
D.2.1 IMPLEMENTATION DETAIL: NEIGHBOUR COUNTING

Features are generally defined by a pattern: *"A node with activation X connected via edge type Z to at least N nodes with activation Y."* While ensuring the correct node and edge types is straightforward, counting neighbours beyond the first layer requires mathematical care. In our architecture, counting is implemented by applying a negative bias *after* sum aggregation. Because aggregated messages can exceed binary values (e.g., a node could receive a summed message of 3), naive counting without proper thresholding can yield false positives. Pattern 4 requires nitrogen atom to be connected to 2 identical non-trivial substructures ([CH2][CH3] chains). In this case, this requires 2 separate logical circuits.

D.2.2 LAYER DEFINITIONS

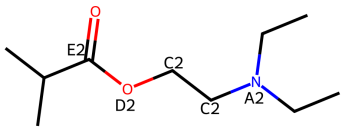
Layer 1: The first layer establishes the foundational chemical features:

- **A1:** Carbon with exactly 2 hydrogens, that is connected via single bond to at least one carbon atom with exactly 3 hydrogens.
- **B1:** Nitrogen that is connected via single bond to at least 3 carbon atoms.
- **C1:** Nitrogen that is connected via single bond to at least 4 carbon atoms.
- **D1:** Carbon with exactly 2 hydrogens, that is connected via single bond to at least one carbon atom.
- **E1:** Oxygen with no hydrogens, that is connected via single bond to at least one carbon atom.
- **F1:** Carbon with no hydrogens, that is connected via a double bond to an oxygen atom.
- **G1:** Carbon with exactly 1 hydrogen that is connected via a single bond to at least 3 other nodes.



Layer 2. The second layer aggregates low-level chemical features into higher-level ones and splits "reasoning" of the model into 2 separate logical circuits (A2, B2).

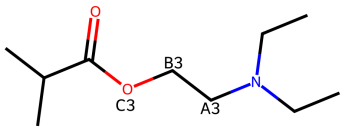
- **A2:** B1 that is **not** C1 and is connected via a single bond to at least 2 A1-activated nodes.
- **B2:** C1 that is connected via a single bond to at least 2 A1-activated nodes.
- **C2:** D1 $\sim$ D1
- **D2:** E1 $\sim$ D1
- **E2:** F1 $\sim$ G1



Note that a node cannot be A2 and B2-activated at the same time. The need to split these computations into 2 separate activations is a direct result of the fact that B1-active nodes can either have B1 $=$ 1 (if the nitrogen has 3 neighbouring carbons) or B1 $=$ 2 (if it has 4 neighbouring carbons). Because our message passing relies on multiplication, a naive implementation of A2 activation which does not take that into account would be sensitive to false positives. As such, the split into B1 and C1 activations in the first layer and A2 and B2 in the second guarantees that a correct calculation is performed regardless of nitrogen's number of neighbours. A2 feature ("B1 that is not C1") is implemented by setting large negative weight for C1 activation, preventing A2-activations on nodes that were C1-activated.

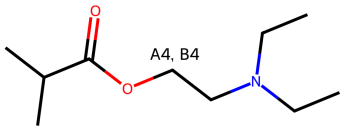
Layer 3. This layer continues aggregation of lower-level features into higher-level ones.

- **A3:** C2 with A2 or B2.
- **B3:** C2 $\sim$ D2
- **C3:** D2 $\sim$ E2



Layer 4. For network to classify a molecule as containing Pattern 4, there should exist a node that contains all of the following activations:

- **A4:** B3 $\sim$ A3
- **B4:** B3 $\sim$ C3



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A4-active and B4-active.

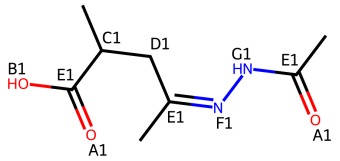
D.3 PATTERN 5

In contrast to the GNN for Pattern 4, GNN for Pattern 5 is conceptually simpler, without relying on 2-circuit architecture.

LAYER DEFINITIONS

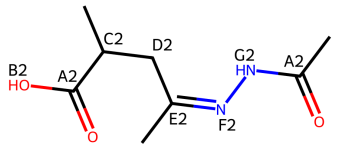
Layer 1: The first layer establishes the foundational chemical features:

- **A1:** Oxygen with exactly 0 hydrogens, that is connected via a double bond to at least one carbon atom.
- **B1:** Oxygen with exactly 1 hydrogen, that is connected via a single bond to at least one carbon atom.
- **C1:** Carbon with exactly 1 hydrogen, that is connected via single bonds to at least 3 other atoms.
- **D1:** Carbon with exactly two hydrogens, that is connected via single bonds to at least 2 other carbon atoms.
- **E1:** Carbon with exactly 0 hydrogens, that is connected to at least 3 other atoms.
- **F1:** Nitrogen with exactly 0 or 1 hydrogens, that is connected via a double bond to at least 1 carbon.
- **G1:** Nitrogen with exactly 1 or 2 hydrogens, that is connected via a single bond to at least 1 other nitrogen.



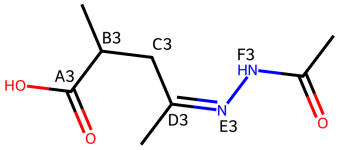
Layer 2: The second layer aggregates low-level chemical features into higher-level ones.

- **A2:** An E1-active node connected via a double bond to at least one A1-active node.
- **B2:** B1 $\sim$ E1
- **C2:** C1 $\sim$ D1
- **D2:** D1 $\sim$ C1
- **E2:** An E1-active node connected via a double bond to at least one F1-active node.
- **F2:** F1 $\sim$ G1
- **G2:** G1 $\sim$ F1



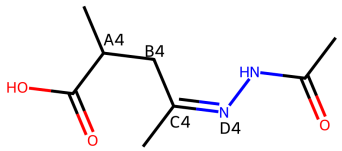
Layer 3: The third layer continues to aggregate lower-level chemical features into higher-level ones.

- **A3:** A2 $\sim$ B2
- **B3:** C2 $\sim$ A2
- **C3:** D2 $\sim$ C2
- **D3:** E2 $\sim$ D2
- **E3:** A F2-active node connected via a double bond to at least one E2-active node.
- **F3:** G2 $\sim$ A2



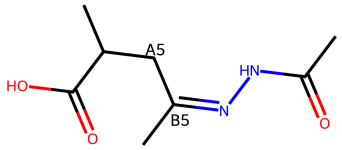
Layer 4: The fourth layer continues to aggregate lower-level chemical features into higher-level ones.

- **A4:** B3 $\sim$ A3
- **B4:** C3 $\sim$ B3
- **C4:** D3 $\sim$ C3
- **D4:** E3 $\sim$ F3



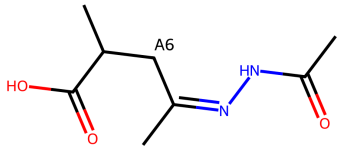
Layer 5: The fifth layer continues to aggregate lower-level chemical features into higher-level ones.

- **A5:** B4 $\sim$ A4
- **B5:** A C4-active node connected via a double bond to at least one D4-active node.



Layer 6: The sixth layer "looks" for A5-active node connected via a single bond to B5-active node:

- **A6:** A5 $\sim$ B5



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A6-active.

D.4 PRELIMINARIES

Introduced by Sundararajan et al. (2017), Integrated Gradients is an attribution method formally defined as:

$$\mathrm{IG}_i(x) = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} \mathrm{d}\alpha \tag{12}$$

where:

1. f is a model being explained. In WOMBAT's case, this means one of our 14 whiteboxes.
2. x is an input to the network. In WOMBAT's case, this means tensors encoding features of nodes and edges (which we will denote, respectively, as x_V and x_E) of a given graph (as specified in Section 3.2).
3. $\mathrm{IG}_i(x)$ is attribution of a specific element of the input, which can be identified by the number i.
4. x'_i is a *baseline*. This is a parameter of the method; it should represent a "neutral" input to the network, i.e. an input that yields a neutral prediction. In WOMBAT's case, this means tensors x'_V and x'_E that match shapes of x_V and x_E, but fully consist of zeroes. All of our networks predict that there is no pattern present when receiving zero-baseline as an input, making the baseline selection justified.

As such, in WOMBAT's case Equation 12 can be seen as 2 equations:

$$\mathrm{IG}_{V_i}(x_V, x_E) = x_{V_i} \int_{\alpha=0}^{1} \frac{\partial f(\alpha x_V, \alpha x_E)}{\partial x_{V_i}} \mathrm{d}\alpha$$
$$\mathrm{IG}_{E_i}(x_V, x_E) = x_{E_i} \int_{\alpha=0}^{1} \frac{\partial f(\alpha x_V, \alpha x_E)}{\partial x_{E_i}} \mathrm{d}\alpha$$

In practice, however, we don't use edge attributions for our evaluations. As such, we can simplify these equations back to 1 equation:

$$\mathrm{IG}_i(x_V, x_E) = x_{V_i} \int_{\alpha=0}^{1} \frac{\partial f(\alpha x_V, \alpha x_E)}{\partial x_{V_i}} \mathrm{d}\alpha \tag{13}$$

For brevity, throughout the remainder of this paper we use the term gradient at interpolation step α, meaning:

$$x_{V_i} \frac{\partial f(\alpha x_V, \alpha x_E)}{\partial x_{V_i}}$$

rather than the raw partial derivative.

D.5 Integrated Gradients in Pattern 4

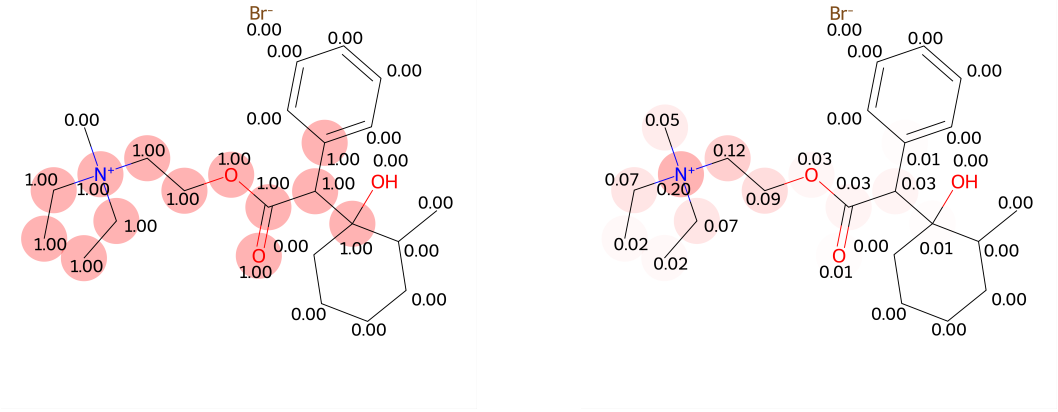
As shown in Figure S1, Integrated Gradients can fail by misattributing non-motif atoms. Figure S3a demonstrates a sudden spike in gradients accumulated by the non-motif atom during the IG's interpolation between baseline and actual input to the network which causes the misattribution. In this section, we investigate *why* it happens in a greater detail. First, we need to index atoms belonging to the molecule. We provide the exact same molecule from Figure S1 in Figure S2a, along with node indices and their respective attributions (denoted as [Node idx]: [Attribution]). In this case, Atom 3 receives bigger attribution than Atom 0 or Atom 5, even though it does not belong to the motif – while these atoms do.

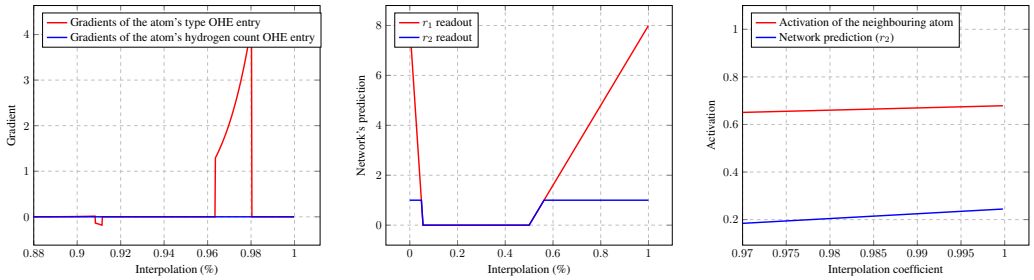
As demonstrated in Figure S3a, the gradient spike for Atom 3 occurs when input features are approximately scaled to $\alpha = 0.96$, peaking around $\alpha = 0.98$. We identify one of the paths leading to this issue and demonstrate it intuitively in Figure S2b.

Assume that all features are interpolated by a given α. Consider an atom a_i, with atomic number n_i and number of adjacent hydrogens h_i. Then it is represented by a vector v_i such that:

$$v_i = [0\ 0\ \ldots\ \overbrace{\alpha}^{v[n_i]}\ \ldots\ \overbrace{\alpha}^{v[100+h_i]}\ \ldots]$$

That is because in WOMBAT's case, representation of every atom is a concatenation of one hot encoding of its atomic number and number of its adjacent hydrogens.


Br⁻
0.00
0.00
0.00



(a) Gradients of the non-motif atom during IG interpolation.

(b) Network's prediction during interpolation of the non-motif atom.

(c) Interpolation of the non-motif atom while other atoms and edges are interpolated at $\alpha = 0.97$.

Figure S3: Analysis of causes of spurious attribution of the non-motif atom in Pattern 4. Figure S3a shows gradients that are accumulated by this atom near the decision boundary. Figure S3b demonstrates that, regardless of atom's presence, network makes a correct classification. However, interpolating atom between states causes model's prediction to drop to 0. Figure S3c showcases impact interpolation of the non-motif atom on activation of neighbouring node and final network's prediction under readout r_2, providing a justification for curve observed in the Figure S3a.

Similarly, all edge representations can be expressed as vector of length 4, where α is the sole non-zero value; its placement is dependent on bond arity.

In case of the atom 3, $n_3 = 6$ (because it is a carbon) and $h_3 = 3$ (because it is a CH_3 carbon). We define now v_i':

$$v_i' = [0\,0\ \ldots\ \overbrace{\alpha + \varepsilon}^{v[n_i]}\ \ldots\ \overbrace{\alpha}^{v[100+h_i]}\ \ldots]$$

where ε is infinitesimally small. In this section, we occasionally refer to v_i' as ε-perturbed representation of a_i. Note that we will only consider perturbing atomic numbers, as we observe gradient spikes for them.

We now demonstrate a computation path that causes gradients during interpolation for v_3 to be non-zero. Consider activation C1, which was previously defined (see Section D.2.2) as a *nitrogen that is connected via single bond to at least 4 carbon atoms*. In practice, this is implemented by following weight set[8]:

$$W_{\text{from}} = [\overbrace{\text{OHE}(100, 6)}^{\text{From carbon}} \ | \ \overbrace{0\ \ldots\ 0}^{\text{regardless of its hydrogen count}}]$$

$$W_{\text{to}} = [\overbrace{\text{OHE}(100, 7)}^{\text{To nitrogen}} \ | \ \overbrace{0\ \ldots\ 0}^{\text{regardless of its hydrogen count}}]$$

$$W_e = [\overbrace{1}^{\text{single bond}}\ 0\ 0\ 0]$$

$$b = -3$$

Consider messages received by the atom 2 (nitrogen ion):

$$m_{i2} = v_i W_{\text{from}} \odot v_2 W_{\text{to}} \odot e_{\text{i2}} W_{\text{e}}$$

Since atom 2 is a nitrogen, it follows that:

[8]Note that here, for simplicity, we define a weight sets for C1 *only*, which can be denoted as vectors. In practice, W_{from}, W_{to} and W_e are *matrices* (see Figure 3).

$$v_2 W_{\text{to}} = \alpha$$

Similarly, atom 2 is connected *only* to carbon atoms via single bonds, which trivially yields:

$$\forall_{i \in \mathcal{N}(v_2)}\, v_i W_{\text{from}} = \alpha$$
$$\forall_{i \in \mathcal{N}(v_2)}\, e_{i2} W_{\text{e}} = \alpha$$

As such, $\forall_{i \in \mathcal{N}(v_2)}\, m_{i2} = \alpha^3$, and, subsequently, sum s of messages received by atom 2 is equal to $s = \sum_{i \in \mathcal{N}(v_2)} m_{i2} = 4\alpha^3$. This means that C1-activation in atom 2 becomes:

$$C_1 = \text{ReLU}(s + b) = \text{ReLU}(4\alpha^3 - 3)$$

Consider now message from an ε-perturbed node a_i. Since the edge encoding and recipient node encoding did not change, we only need to consider the "from" component of the message multiplication:

$$v_i' W_{\text{from}} = \alpha + \varepsilon$$

hence:

$$m_{i2}' = \alpha^2(\alpha + \varepsilon) = \alpha^3 + \varepsilon\alpha^2$$

meaning that sum of messages received by the atom 2 would in turn become $s' = 4\alpha^3 + \varepsilon\alpha^2 = s + \varepsilon\alpha^2$, and:

$$C_1' = \text{ReLU}(4\alpha^3 + \varepsilon\alpha^2 - 3)$$

As such, for sufficiently large α, $C_1' = C_1 + \varepsilon_2$, where ε_2 is some positive constant. Isomorphic reasoning can be performed to show the following:

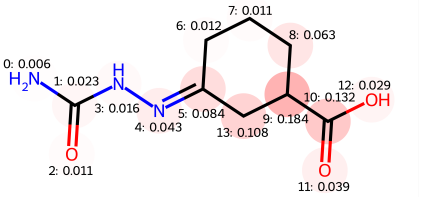
1. ε_2-perturbed C1 activation in atom 2 causes some ε_3-perturbation of activation B2 in atom 2 for sufficiently large α.
2. ε_3-perturbed B2 activation in atom 2 causes some ε_4-perturbation of activation A3 in atom 6 for sufficiently large α.
3. ε_4-perturbed A3 activation in atom 6 causes ε_5-perturbation of activation A4 in atom 7 for sufficiently large α.
4. ε_5-perturbation of activation A4 in atom 7 causes ε_6-perturbation of model's output for sufficiently large α.

This provides an explicit explanation for the gradient spike.

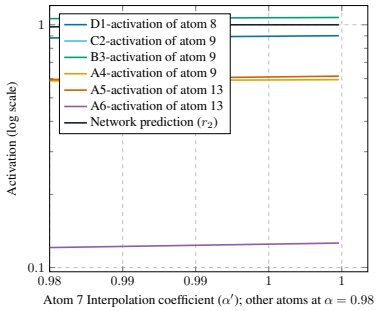
D.6 Integrated Gradients in Pattern 5

Ultimately, the cause of failure of Integrated Gradients in some of Pattern 5's examples is very similar to the one presented for Pattern 4 (Section D.5). As such, we will only describe a chain of perturbation propagation contributing to IG's unfaithfulness. We present indices of atoms belonging to the molecule in Figure S4a.

1. ε_0-perturbation of Atom 7 (not belonging to motif) causes ε_1-perturbation of D1 activation in the Atom 8 for sufficiently large α.
2. ε_1-perturbation of D1 activation in Atom 8 causes ε_2-perturbation of C2 activation in Atom 9 for sufficiently large α.



(a) Molecule from Figure 5, with atom indices and IG attributions annotated.



(b) Impact of interpolation of atom 7 while other atoms and edges are interpolated at $\alpha = 0.98$.

Figure S4: Case study of IG's failure in Pattern 5. Figure S4a represents node indices and their respective attributions, while Figure S4b shows some of activations impacted by interpolation of the Atom 7 when all elements are interpolated at $\alpha = 0.98$.

3. ε_2-perturbation of C2 activation in Atom 9 causes ε_3-perturbation of B3 activation in Atom 9 for sufficiently large α.
4. ε_3-perturbation of B3 activation in Atom 9 causes ε_4-perturbation of A4 activation in Atom 9 for sufficiently large α.
5. ε_4-perturbation of A4 activation in Atom 9 causes ε_5-perturbation of A5 activation in Atom 13 for sufficiently large α.
6. ε_5-perturbation of A5 activation in Atom 13 causes ε_6-perturbation of A6 activation in Atom 13 for sufficiently large α.
7. ε_6-perturbation of A6 activation in Atom 13 causes ε_7-perturbation of r_2 readout for sufficiently large α, yielding a positive gradient.

We plot values of these activations in Figure S4b.

E Additional Results on Failure Modes of Shapley Value Sampling, PGExplainer, and GNNExplainer

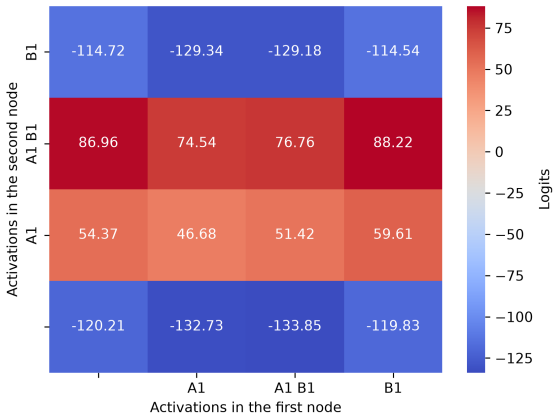
Both PGExplainer and GNNExplainer perform poorly on WOMBAT, even when evaluating simple patterns. At the same time, Shapley Value Sampling performs reasonably, although its performance often yields imperfect attributions. We focus our analysis on models utilizing the readout in Eq. 4, as its bounded output ($[0, 1]$) should theoretically simplify the attribution task.

E.1 Shapley Value Sampling

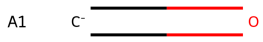
Although Shapley Value Sampling (SVS) maintains relatively high AUROC across r_1 and r_2 readouts, it frequently fails to achieve perfect node ranking, a trend that intensifies with motif complexity. This is most visible for Pattern 3 under the r_2 readout, where SVS achieves perfect attribution (AUROC $= 1.0$) for only 3 out of 22 positive molecules (Table S1). We empirically verify that increasing SVS's sampling budget improves its performance, especially across bigger motifs (see Figure S7).

E.2 PGExplainer

Edge classification. PGExplainer relies on a Multilayer Perceptron (MLP) to predict attributions of edges. Let $z_1, z_2, z_3, \ldots, z_n$ be model's latent representations of nodes $v_1, v_2, v_3, \ldots, v_n$ after the last MPNN layer (before the readout). To predict attribution for edge e_{ij}, it first requires calculation of the edge logit:

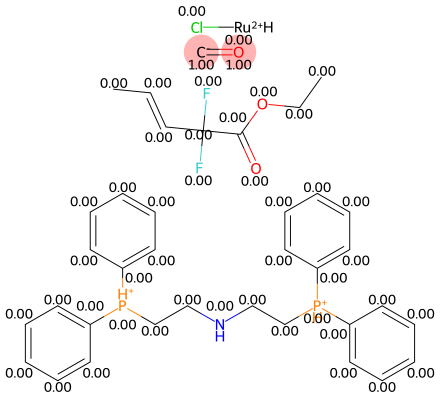


(a) Logits returned by MLP trained by PGExplainer to explain Pattern 12 depending on activations of the last layer in Pattern 12 whitebox.

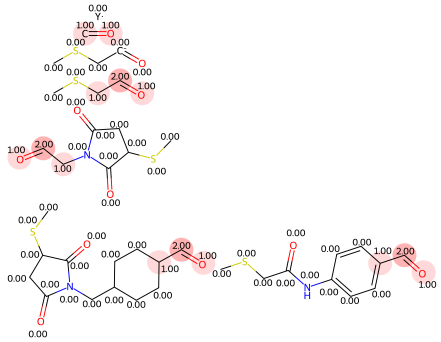


(b) Activations of Pattern 12 whitebox for a formyl anion.

Figure S5: Figure S5a demonstrates that while presence of B1 activation in one of edges terminals increases logit values, A1 activation is sufficient and necessary for PGExplainer to attribute the edge connecting 2 nodes. As such, carbons belonging to formyl anions, despite only being A1-activated (Figure S5b) still get attributed by PGExplainer.

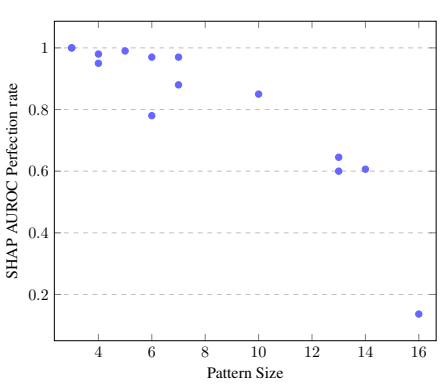


(a) Sole failing negative sample.

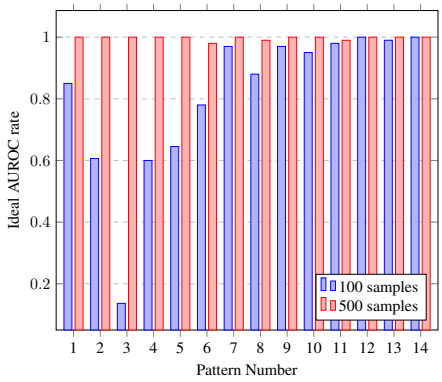


(b) One of 2 failing positive samples.

Figure S6: In case of both negative (Figure S6a) and positive samples (Figure S6b), PGExplainer fails by misattributing formyl anions. In the former, this causes IQR criterion failure (as null explanation was expected); in the latter, formyl anions are attributed alongside legitimate occurrences of Pattern 12 in molecule, causing AUROC to drop.



(a) Performance of Shapley Value Sampling degrades relative to the pattern size.

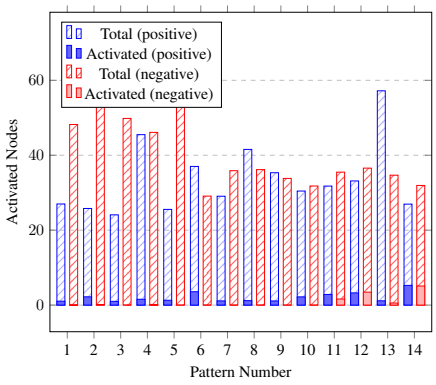


(b) Increasing number of samples helps to recover perfectly majority of motifs.

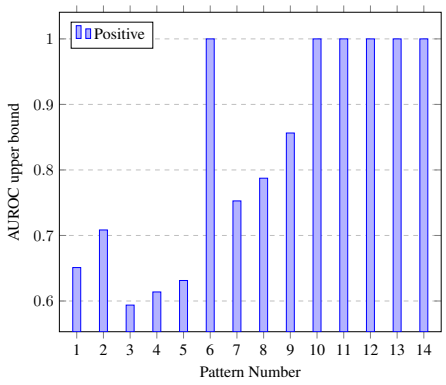
Figure S7: Shapley Value Sampling's ideal attribution rate is dependent on the size of the motif (Figure S7a). Increasing of the sample size helps it achieve better performance (Figure S7b). Here, we define the "Pattern Size" as number of heavy atoms in a molecule after substituting all wildcards as carbons.

Table S1: Comparison of amount of perfect (i.e. equal to 1.0) AUROCs for Shapley Value Sampling and Integrated Gradients.

Pattern No.	Shapley Value Sampling			Integrated Gradients		
	Perfect	Total	Perfection Rate	Perfect	Total	Perfection Rate
1	85	100	0.85	7621	7660	0.99
2	57	94	0.61	236	236	1.00
3	3	22	0.14	22	22	1.00
4	60	100	0.60	964	1137	0.85
5	20	31	0.65	29	31	0.94
6	78	100	0.78	9984	9984	1.00
7	97	100	0.97	9990	9990	1.00
8	88	100	0.88	9996	9996	1.00
9	97	100	0.97	9989	9989	1.00
10	95	100	0.95	9989	9989	1.00
11	98	100	0.98	9999	9999	1.00
12	100	100	1.00	9999	9999	1.00
13	99	100	0.99	9997	9997	1.00
14	100	100	1.00	10000	10000	1.00



(a) Mean numbers of activated nodes after passing through the last MPNN layer for each pattern in negative and positive samples compared to mean numbers of *all* nodes in negative and positive samples.



(b) Loose upper bound on AUROC of any explainer that classifies edges based on last-layer activations of edge terminal nodes (assuming oracle-like performance if any of the edge's terminals is non-zero, and predicting that an edge is unimportant otherwise).

Figure S8: PGExplainer's design choice of attributing edges based on their terminal nodes' activations greatly limits its performance in WOMBAT. Figure S8a shows that, on average, total number of nodes with non-zero activations is very limited, limiting PGExplainer's ability to discriminate between important and unimportant edges. To formalise this intuition, we present a loose upper bound to AUROC it could achieve in WOMBAT for every Pattern (Figure S8b).

$$\omega_{ij} = \mathrm{MLP}([z_i; z_j]) \tag{14}$$

for more details, please refer to Luo et al. (2020). In WOMBAT, the final-layer representations z_i are frequently zero vectors, including nodes belonging to the target pattern. On average, in WOMBAT's whiteboxes a vast majority of nodes is zero-activated after the last layer (Figure S8a). Consequently, many edges receive exactly the same input $[0; 0]$ in Equation 14 and must therefore receive the same score. In particular, if a pattern edge and an irrelevant edge both connect pairs of zero-activation nodes, PGExplainer cannot distinguish between them, irrespective of how successfully its explanation network is optimized. Edges incident to the active node may remain distinguishable from edges whose two endpoints have zero representations. This gives PGExplainer some capacity to recover the portion of a pattern lying in the immediate neighbourhood of that node. Perfect separation of motif and non-motif edges is impossible however whenever the ground-truth pattern contains zero–zero edges while the surrounding graph also contains irrelevant zero–zero edges. This ambiguity is consistent with our empirical results: under the main evaluation protocol, PGExplainer achieved reasonable, although imperfect, performance only on Pattern 12, one of the smallest evaluated patterns, consisting of three nodes.

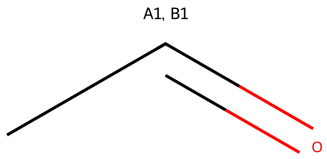
We establish an approximate upper bound on PGExplainer's performance in Figure S8b. We construct it as follows:

1. Assign a score of 1 to every ground truth motif node whose final-layer representation is non-zero, and to every ground-truth motif node adjacent to such a node.
2. Assign a score of 0 to all remaining nodes.

Note that we assume that PGExplainer assigns zero attribution to an edge when both endpoints have zero final layer representations. This does not necessarily have to be true, but given that a vast majority of nodes tend to be zero-activated (see Figure S8a), we believe it to be a sensible assumption. As such, we believe that this approximation should be treated as a practical (and loose) upper bound. It can be observed that this bound does not impact smaller Patterns (such as Pattern 12, for which PGExplainer achieves a decent performance), but caps PGExplainer's performance severely in case of Patterns such as Pattern 2 or 3.

PGExplainer in Pattern 12. While PGExplainer manages to explain majority of molecules containing Pattern 12, there are a few cases where it fails to score ideal AUROC score. Furthermore, it also fails the IQR test for 1 of the molecules in the negative sample test. All of these failures are caused by the presence of formyl anions in the graphs (see Figure S6). To understand this, one needs to consider weights of the Pattern 12 whitebox. It contains only 1 MPNN layer with the following activation definitions[9]:

1. **A1:** Carbon with exactly 1 hydrogen, that is connected via double bond to at least one oxygen atom.
2. **B1:** Carbon with exactly 1 hydrogen, that is connected via a single bond to at least one other atom.



The readout effectively computes $\mathrm{A1} \wedge \mathrm{B1}$. However, for a carbon atom with conventional valence, A1 generally implies B1. A1 specifies a carbon atom with a double bond to oxygen and a bond to exactly one hydrogen. Such a carbon usually has an additional single bond to another atom and therefore also activates B1. Accordingly, $\mathrm{A1} \implies \mathrm{B1}$ holds for most examples in the training dataset, although counterexamples are present. In turn, PGExplainer's MLP instead of learning to check for $\mathrm{A1} \wedge \mathrm{B1}$ in one of edge's terminals checks for A1; in other words, presence of A1-activation in one of edge's terminals is necessary and sufficient for edge attribution by PGExplainer (see Figure S5a). This shortcut leads to PGExplainer failing by attributing formyl anions, which do not contain Pattern 12, but they do exhibit A1-activation (see Figure S5b).

Optimisation. As PGExplainer requires a separate training phase, it is highly sensitive to hyperparameters such as learning rate and epoch count. We conducted a grid search on Pattern 14, one of the simplest patterns the explainer failed to identify, to assess this sensitivity (Figure S9). The results indicate a tendency for the model to fail to converge. To test stability, we executed seven independent runs using the best-performing hyperparameters from the grid search ($\mathrm{lr} = 0.1$, 20 epochs); however, the model successfully converged in only one instance.

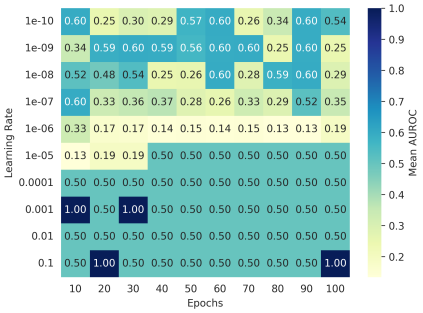


Figure S9: Results of performing grid search for training parameters of PGExplainer for Pattern 14 with readout r_2. Each value comes from a fully separate training run.

E.3 GNNExplainer

Optimisation. Similarly to PGExplainer, GNNExplainer is sensitive to hyperparameter selection. We performed an exhaustive random search across 50,000 parameter combinations to determine if it could correctly attribute a single positive molecule for Pattern 14. Surprisingly, not a single combination was a successful attribution.

[9]For activation notation in WOMBAT, please refer to Section D.1.

F Impact of Whitebox Architecture

F.1 Decision boundary sharpness

We further investigated whether the "sharp" decision boundaries of our whitebox architecture impact optimization of PGExplainer and GNNExplainer. To address this, we implemented a modified MPNN formulation that replaces product operators with a minimum function:

$$m'_{ij} = \min\left(\text{ReLU}(x_i^T W_{\text{from}}),\ \text{ReLU}(x_j^T W_{\text{to}}),\ \text{ReLU}(e_{ij}^T W_e)\right) \tag{15}$$

$$r'_2 = \text{ReLU}\left(1 - \prod_{v_i \in V} \text{ReLU}\left(1 - \min_j(\text{ReLU}(x_{i,j}))\right)\right) \tag{16}$$

While this architecture maintains perfect accuracy on Patterns 1 and 2, it did not improve explainability. Ten independent PGExplainer runs (varying epoch counts) and a random search of over 11,000 GNNExplainer configurations all failed to explain Pattern 14.

F.2 GINEConv whitebox experiment

WOMBAT's custom architecture facilitates whitebox construction, but may raise concerns about whether our findings transfer to more conventional MPNN architectures. To explore this question, we construct an additional whitebox designed to detect the simultaneous presence of fluorine and a carbonyl group, a task inspired by Sanchez-Lengeling et al. (2020). The model uses the GINEConv layer Hu et al. (2019) implemented in PyTorch Geometric Fey et al. (2025), and is released alongside WOMBAT to facilitate further research. Note that we do not claim to have created a whitebox network for Sanchez-Lengeling et al. (2020), as we have not performed a validation on the dataset described there and our task definitions and valence assumptions are likely to differ.

Task definition. We check for simultaneous existence of two SMARTS patterns in a structure:

1. `[!#1]F` – Fluorine atom connected to some other nonhydrogen atom.
2. `[#6]=O` – Oxygen atom connected via a double bond to a carbon atom.

Input encoding. Input molecules are encoded as all other molecules in WOMBAT, albeit our network discards the hydrogen OHE as it is not needed to make a prediction. Furthermore, we assume that all molecules satisfy WOMBAT's constraints on valence.

Architecture overview. Our network consists of the following:

1. An embedding layer;
2. a GINEConv layer;
3. an MLP-based readout layer.

where a GINEConv's layer is defined as follows:

$$x'_i = \text{MLP}\left((1+\varepsilon)x_i + \sum_{j \in \mathcal{N}(i)} \text{ReLU}(x_j + e_{ji})\right) \tag{17}$$

where x_i means a representation of the i-th node before the layer and x'_i representation of the i-th node after the layer.

Hyperparameters and network calibration. In the Equation 17, we set $\varepsilon = 0$. Furthermore, we treat prediction above the threshold of 0.5 as positive, and below that threshold as negative.

Notation. For brevity, we define a vector $a_i = (1+\varepsilon)x_i + \sum_{j \in \mathcal{N}(i)} \mathrm{ReLU}(x_j + e_{ji})$, meaning aggregated messages received by node v_i before MLP.

Embeddings. We define the embeddings as follows:

- Carbon atoms get embedded as $\mathrm{embedding}_{\mathrm{C}} = [0 \quad 0 \quad -10 \quad 1 \quad 0]$
- Oxygen atoms get embedded as $\mathrm{embedding}_{\mathrm{O}} = [0 \quad 0 \quad 0 \quad 0 \quad 0]$
- Fluorine atoms get embedded as $\mathrm{embedding}_{\mathrm{F}} = [1 \quad -1 \quad -10 \quad 0 \quad 0]$
- Double bonds get embedded as $\mathrm{embedding}_{\mathrm{double}} = [0 \quad 0 \quad 0 \quad 0 \quad 1]$
- All other atoms and bonds get embedded as $\mathrm{embedding}_{\mathrm{other}} = [0 \quad 0 \quad 0 \quad -10 \quad 0]$

MLP layer in GINEConv. The MLP performs a linear projection from $\mathbb{R}^5$ to $\mathbb{R}^2$ and then applies $\tanh$ activation function.

The weights for the perceptrons are as follows:

1. $[0 \quad 0 \quad 1 \quad 1 \quad 1]$ and bias of -1.
2. $[1 \quad 1 \quad 0 \quad 0 \quad 0]$ and bias of 0.

The first perceptron detects $\mathrm{C} = \mathrm{O}$ pattern, and the second one detects $\mathrm{F} - *$ pattern. We will now provide the rationale for weights of the first perceptron.

Note that in the case of the first perceptron, it won't activate in carbon atoms as $\mathrm{embedding}_{\mathrm{C}}[2] = -10$[10]. It can however be activated in oxygen atoms ($\mathrm{embedding}_{\mathrm{O}}[2] = 0$).

Observe that:

$$\mathrm{ReLU}(\mathrm{embedding}_{\mathrm{C}} + \mathrm{embedding}_{\mathrm{double}}) = [0 \quad 0 \quad 0 \quad 1 \quad 1]$$

but:

$$\mathrm{ReLU}(\mathrm{embedding}_{\mathrm{C}} + \mathrm{embedding}_{\mathrm{single}}) = [0 \quad 0 \quad 0 \quad 0 \quad 0]$$

as $\mathrm{embedding}_{\mathrm{single}} = \mathrm{embedding}_{\mathrm{other}}$, which effectively "vetoes" the carbon signal. As such, for an oxygen atom to achieve the following activation:

$$a_i = [0 \quad 0 \quad 0 \quad 1 \quad 1]$$

it needs to be an oxygen atom, and it needs to be connected to a carbon atom with a double bond. Note that, if the oxygen atom is connected with a double bond to the carbon atom, $\mathrm{MLP}(a_{\mathrm{oxygen}})[0] = \tanh(2-1) = \tanh(1)$ and otherwise $\mathrm{MLP}(a_{\mathrm{oxygen}})[0] \leq \tanh(0)$.

Now, we will discuss the second perceptron. Note that:

$$\mathrm{ReLU}([1 \quad -1 \quad -10 \quad 0 \quad 0]) = [1 \quad 0 \quad 0 \quad 0 \quad 0]$$

Observe that all atoms neighbouring with a fluorine atom will receive a non-zero message from it and subsequently activate the fluorine perceptron.

Note that technically this whitebox is not perfect: in some cases the veto signal could technically be "overwhelmed" by many carbon atoms forming double bonds with non-oxygen atoms. While this could be partially rectified by setting the veto weight to a smaller value (e.g. -100), such cases are unlikely to happen and – furthermore – setting this weight to -10 is enough to pass our simplified validation.

At the same time, note that fluorine atom won't activate this perceptron on its own, as $1 \cdot 1 + 1 \cdot (-1) + 0 = 0$. This prevents fluoride ions from activating this perceptron when it fails our predefined SMARTS (i.e. the fluorine atom must be connected to something else).

[10] We assume 0-based indexing.

Table S2: Explainer performance on the GINEConv whitebox detecting the simultaneous presence of fluorine and a carbonyl group. Mean AUROC and perfect explanations (AUROC = 1) are evaluated on positive molecules; IQR success rates are evaluated on negative molecules. Only Saliency produces any perfect explanations, in 3 of 9,999 cases. PGExplainer's perfect IQR success rate reflects collapsed, uniformly zero attributions. Conversely, non-uniform attributions on negative molecules may faithfully reflect the contribution of one of the two required substructures.

Explainer	Mean AUROC↑	Perfect explanations	Perfect rate	Mean Success Rate (IQR criterion)
PGExplainer	0.5000	0 / 9,999	0.00%	1.0000
Integrated Gradients	0.7077	0 / 9,999	0.00%	0.1478
Saliency	0.7884	3 / 9,999	0.03%	0.0590
SHAP Sampling	0.6494	0 / 100	0.00%	0.3600
Input × Gradient	0.7077	0 / 9999	0.00%	0.2274
GNN Explainer	0.7075	0 / 9999	0.00%	0.1516
SubgraphX	0.6560	0 / 100	0.00%	N/A

Readout. Max readout is performed to obtain the $x \in \mathbb{R}^2$ vector representing the graph. We then make a final prediction by using one linear MLP layer (with one perceptron) with the following weights:

$$[1 \quad 1]$$

and a bias term of -1.51, meant to calibrate the network. Sigmoid activation is used to obtain the final prediction.

Validation. We experimentally validated the network on 153173 positives and 846826 negatives randomly sampled from the Pattern 12 validation set (SMILES-wise; labels were generated to match the pattern from this task) and found no misclassifications. Note that this is a "simplified" validation pipeline, as the "regular" one would require filtering entire PubChem and finding Tversky negatives. We omitted it due to time constraints.

Quantitative results. None of the explainers produce a perfect explanation, with the exception of Saliency (3 out of 9,999 cases). Their mean AUROC is also substantially lower than in case of WOMBAT's other whiteboxes. We present the full results in Table S2.

Qualitative results. For the evaluation, we've used the same pipeline as for WOMBAT's benchmarks. For qualitative analysis, we randomly sampled 20 SMILES from positive and 20 SMILES from negative set and observed the following:

- IG, SVS, Saliency, Input x Gradient consistently fail to properly attribute the neighbouring atom in F-* substructure. This atom is being relied on topologically (as its existence is crucial for activation), but other than participating in message passing it does not "add" anything to the final score itself. This may be partially attributed to the fact that we explain only node/edge features. Formulation based on masking graph connectivity itself may help with this issue, as masking the edge would lower the network's prediction. This was not a problem in other whiteboxes, as masking an edge feature vector automatically scaled down all messages sent through that edge, due to our MPNN formulation in Equation 1.
- Similarly, IG, SVS and Input × Gradient fail to attribute the oxygen atom in C=O substructure. Again, this atom is being relied on topologically (as its existence is crucial for activation), but other than participating in message passing it does not "add" anything to the final score itself (note that embedding of oxygen atom is a vector of zeroes). In case of IG and SVS, this may also be partially caused due to the baseline being a zero-vector, as in case of this network there is no difference between oxygen's embedding when it's masked and when it's not. This could potentially be addressed by performing IG directly on an atom's embedding space, not on the space of inputs to the GNN, and then using a different baseline for the oxygen atom. This interpolation would likely not fix everything, as failure to attribute neighbouring atom of fluorine does not share this failure mode (carbon atoms, which often neighbour fluorine atoms, do not have zero-embeddings, yet are not properly credited).

- Saliency is the only explainer to attribute the oxygen atom in the C=O substructure, as other entries in one hot encoding (that are zeros) accumulate gradients. The oxygen entry itself accumulates zero gradients, as oxygen embedding is a vector of zeroes (this also shows why Input $\times$ Gradient does not attribute that oxygen). As such, this attribution seems to be accidental, especially given the fact that saliency fails to attribute neighbours of fluorine atoms.
- PGExplainer collapsed during training and produces null attributions. However, during our test run on a much smaller number of molecules to explain (and in its training dataset) it managed to converge. Qualitative analysis of that checkpoint shows that it properly attributes C=O motif, while not attributing F-* motif. That said, it sometimes also attributes motifs that are not C=O, like S=O or CON.
- GNNExplainer, similarly to Input x Gradient and other explainers, only attributes fluorine atoms belonging to F-* substructures and carbon atoms belonging to the C=O substructures.
- SubgraphX is severely disadvantaged, as the explanation is not a connected subgraph (C=O and F-* motif can be on different sides of the molecule). As such, it sometimes produces giant attributions that span through a majority of the molecule to connect the two motifs, while sometimes attributing only one of them. It also fails to attribute oxygen atoms in C=O substructures.
- On negative samples, substantially more explainers fail the IQR criterion. PGExplainer is the exception because its null attributions are necessarily uniform and therefore always satisfy the criterion. We attribute the lower success rates to negative molecules containing one of the two required substructures. Although such molecules remain below the classification threshold, the present substructure genuinely increases the network's output, and faithful explainers may assign it non-uniform attribution. Consequently, failure of the IQR criterion does not necessarily indicate either shortcut learning or explainer failure.

F.3 Differing implementations

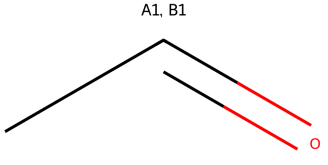
WOMBAT whiteboxes for given patterns are *not unique*. In other words, there may exist multiple (or – in fact – infinitely many) whiteboxes that are able to detect a given pattern. Appending layers that propagate signal without affecting the readout leaves the r_2 decision function identical (within space of one-hot-encoded inputs), and there are many non-trivial reformulations besides. Note that implementation changes can also change a model's logits (if using the r_1 readout, which is not bounded) without the model losing its correctness (i.e. making correct predictions for a predetermined threshold of 0.5).

To illustrate this point, we will present 4 different whiteboxes for Pattern 12 detection. The first of them is the original whitebox for Pattern 12, while the remaining three are new. For the weight notation, please refer to Section D.1. Unless stated otherwise, all results will be stated for the readout function r_2. All new whiteboxes were validated on a subset of randomly sampled 1 million molecules from Pattern 12's validation set.

F.3.1 The original implementation

Layer 1. The first layer aggregates basic chemical features.

1. **A1:** Carbon with exactly 1 hydrogen, that is connected via double bond to at least one oxygen atom.
2. **B1:** Carbon with exactly 1 hydrogen, that is connected via a single bond to at least one other atom.



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A1-active and B1-active at the same time.

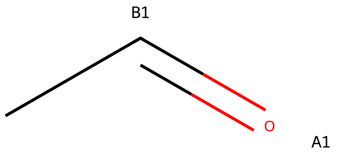
Observations. This whitebox exposes PGExplainer's weakness, i.e. being capable of performing shortcut learning to make explanations that are not always faithful. This is discussed in more detail in the Section E.2.

F.3.2 THE SECOND WHITEBOX (PRO-PGEXPLAINER)

This whitebox consists of 2 layers.

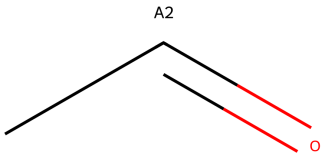
Layer 1. The first layer aggregates basic chemical features.

1. **A1:** Oxygen that is connected via a double bond to at least one carbon.
2. **B1:** Carbon with exactly 1 hydrogen, that is connected via a single bond to at least one other atom.



Layer 2. The second layer aggregates low-level chemical features into higher-level ones.

1. **A2:** B1 with a double bond to A1.



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A2-active.

Observations. This whitebox changes the implementation so that PGExplainer cannot overfit the way it did for the first implementation. Here, the only rule for PGExplainer to "learn" is that if an edge has a terminal in A2-active node, it needs to be attributed, which is generally trivial for an MLP to learn[11]. Indeed, PGExplainer always scored AUROC=1.0 for that whitebox.

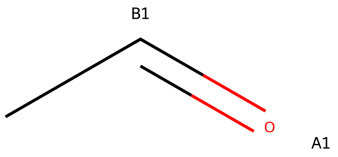
F.3.3 THE THIRD WHITEBOX (ANTI-PGEXPLAINER)

This whitebox consists of 3 layers.

Layer 1. The first layer aggregates basic chemical features and is identical to the first layer from the pro-PGExplainer whitebox.

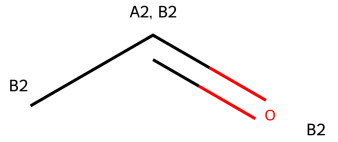
[11] This is in opposition to the $\mathrm{A1} \wedge \mathrm{B1}$ rule PGExplainer should have learned for the original implementation, but failed because of the heuristic $\mathrm{A1} \implies \mathrm{B1}$ which was true for *most* but not all molecules

1. **A1:** Oxygen that is connected via a double bond to at least one carbon.
2. **B1:** Carbon with exactly 1 hydrogen, that is connected via a single bond to at least one other atom.



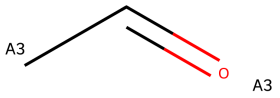
Layer 2. The second layer aggregates low-level chemical features into higher-level ones.

1. **A2:** B1 with a double bond to A1.
2. **B2:** Activation in *any* node (this is done by setting the bias term to 1 and was not done in any other whitebox).



Layer 3. The third layer "moves" the A2 activation to the nodes bordering the carbon atom.

1. **A3:** B2 connected with A2 via any bond.



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A3-active.

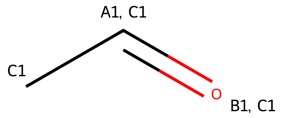
Observations. This whitebox is meant to further show PGExplainer's implementation reliance, as it effectively "spreads around" the activation that essentially tells that the motif is present to adjacent nodes, making PGE's reliance on last-layer activations troublesome. That is because – this time – merely attributing all edges which have at least one A3-activated terminal can yield false positives (as the carbon's neighbour will be A3-activated, meaning edges connecting it to its irrelevant neighbours would get attributions). Indeed, under that implementation scheme, PGE's failures are not isolated cases (although it does not return null attributions; it achieves a mean AUROC of 0.983, but ideal attributions only for 5% of molecules). Note that – in this case – the final representations are too ambiguous from the perspective of PGExplainer (i.e. its imperfect performance is *not* an artifact of optimisation process causing it to learn some heuristic).

F.3.4 THE FOURTH WHITEBOX (ANTI-IG)

This whitebox consists of 6 layers.

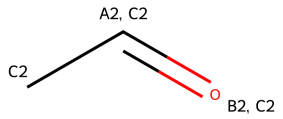
Layer 1. The first layer aggregates basic chemical features. Note the presence of C1 activation, which carries almost no information but will be computationally used to "smear" attributions.

1. **A1:** Carbon with exactly 1 hydrogen, that is connected via a single bond to at least one other atom.
2. **B1:** Oxygen, that is connected via any bond to at least one other atom.
3. **C1:** Any atom connected to any atom via any bond (note that this is possible without setting positive bias)



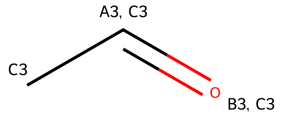
Layer 2. The second layer does nothing, except preserving the A1, B1 and C1 activations further. To preserve A1 and B1 activations, we use predicate checking for connection to C1 (every non-isolated atom is C1-activated).

1. **A2:** A1-activated node connected via any bond to at least one C1-activated atom.
2. **B2:** B1-activated node connected via any bond to at least one C1-activated atom.
3. **C2:** C1-activated node connected via any bond to at least one C1-activated atom.



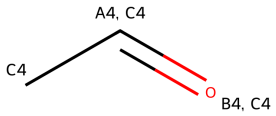
Layer 3. The third layer also does nothing, except further activation preservation.

1. **A3:** A2-activated node connected via any bond to at least one C2-activated atom.
2. **B3:** B2-activated node connected via any bond to at least one C2-activated atom.
3. **C3:** C2-activated node connected via any bond to at least one C2-activated atom.



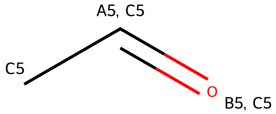
Layer 4. The fourth layer likewise only preserves activations.

1. **A4:** A3-activated node connected via any bond to at least one C3-activated atom.
2. **B4:** B3-activated node connected via any bond to at least one C3-activated atom.
3. **C4:** C3-activated node connected via any bond to at least one C3-activated atom.



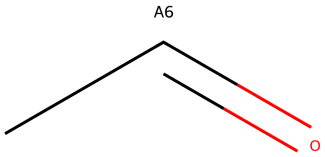
Layer 5. The fifth layer likewise only preserves activations.

1. **A5:** A4-activated node connected via any bond to at least one C4-activated atom.
2. **B5:** B4-activated node connected via any bond to at least one C4-activated atom.
3. **C5:** C4-activated node connected via any bond to at least one C4-activated atom.



Layer 6. The sixth layer combines chemical features into a higher-level one.

1. **A6:** A5-activated node connected via a double bond to at least one B5-activated atom.



Readout. The final graph-level readout function simply checks for the existence of at least one node that is A6-active.

Observations. The additional layers preserve the motif recognition rule on one-hot-encoded molecular inputs: every non-isolated node is C1-active, so the propagation steps preserve which nodes satisfy the A1 and B1 conditions. However, they introduce additional computational dependencies on neighbouring features through the C_i activations, allowing non-motif atoms to contribute gradients along IG's interpolation path. This effectively "smears" the resulting attributions: despite a mean AUROC of 0.97, IG achieves perfect attributions on only 7.60% of the evaluated molecules. We illustrate the network's layer-by-layer activations on a larger molecule in Figure S10 and the corresponding IG attribution in Figure 7. Note that attributed non-motif nodes are *not* logically needed, as the substructure alone is both necessary and sufficient for the network to predict its existence.

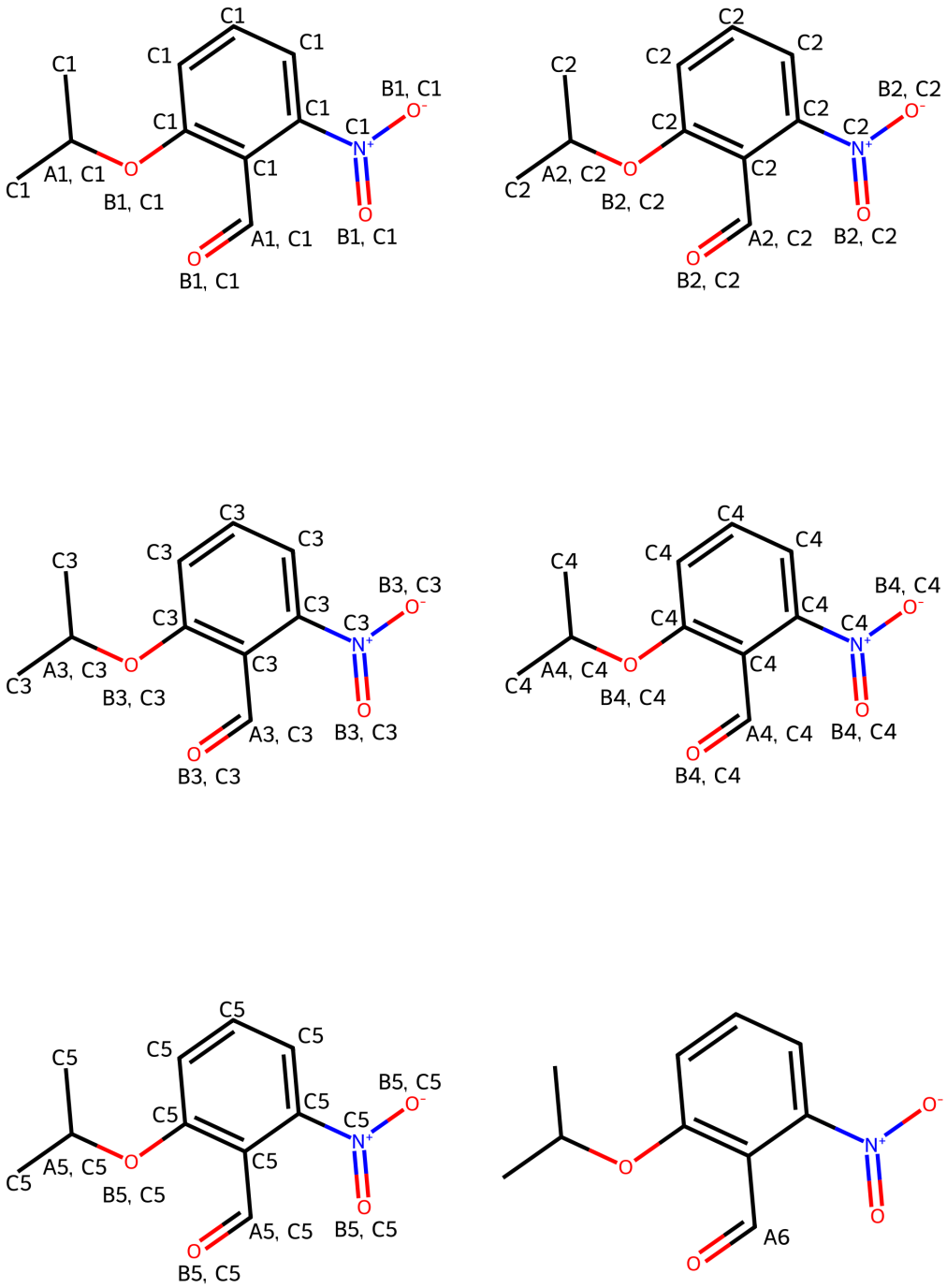


Figure S10: Layer-by-layer activations of the anti-IG Pattern 12 on an example molecule.

G ADDITIONAL EXPERIMENTAL RESULTS

In this section we present results presenting AUROC for readout r_1 in Table S3 and for readout r_2 in Table S4, and combined results colorcoded in Table S7. Then we present results for mean Average Precision for readout r_1 in Table S5, for readout r_2 in Table S6, and combined in Table S8.

Table S3: Mean AUROCs of all explainers across different whiteboxes (using readout function from Equation 3) when evaluated on positive examples.

	mean AUROC ↑						
No.	**GNN Expl.**	**Input x Grad.**	**IG**	**PG Expl.**	**SHAP Sampl.**	**Saliency**	**SubgraphX**
Pattern 1	0.500 ± 0.000	0.994 ± 0.000	0.994 ± 0.000	0.500 ± 0.000	0.989 ± 0.002	0.994 ± 0.000	0.935 ± 0.011
Pattern 2	0.500 ± 0.000	0.997 ± 0.002	0.997 ± 0.002	0.500 ± 0.000	0.975 ± 0.003	0.997 ± 0.002	0.960 ± 0.012
Pattern 3	0.500 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.966 ± 0.006	1.000 ± 0.000	0.916 ± 0.026
Pattern 4	0.500 ± 0.000	0.982 ± 0.002	0.982 ± 0.002	0.500 ± 0.000	0.980 ± 0.003	0.986 ± 0.002	0.798 ± 0.019
Pattern 5	0.500 ± 0.000	0.987 ± 0.006	0.989 ± 0.005	0.500 ± 0.000	0.939 ± 0.030	0.991 ± 0.004	0.857 ± 0.032
Pattern 6	0.500 ± 0.000	0.942 ± 0.001	0.942 ± 0.001	0.500 ± 0.000	0.993 ± 0.003	0.942 ± 0.001	0.914 ± 0.016
Pattern 7	0.500 ± 0.000	0.982 ± 0.001	0.982 ± 0.001	0.500 ± 0.000	0.995 ± 0.003	0.982 ± 0.001	0.939 ± 0.014
Pattern 8	0.500 ± 0.000	0.973 ± 0.001	0.973 ± 0.001	0.500 ± 0.000	0.990 ± 0.004	0.973 ± 0.001	0.888 ± 0.019
Pattern 9	0.500 ± 0.000	0.985 ± 0.001	0.985 ± 0.001	0.500 ± 0.000	0.997 ± 0.001	0.985 ± 0.001	0.941 ± 0.013
Pattern 10	0.500 ± 0.000	0.870 ± 0.001	0.870 ± 0.001	0.500 ± 0.000	0.996 ± 0.002	0.870 ± 0.001	0.874 ± 0.016
Pattern 11	0.500 ± 0.000	0.939 ± 0.001	0.939 ± 0.001	0.500 ± 0.000	0.997 ± 0.003	0.939 ± 0.001	0.888 ± 0.018
Pattern 12	0.980 ± 0.001	0.980 ± 0.001	0.980 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	0.980 ± 0.001	0.824 ± 0.024
Pattern 13	0.500 ± 0.000	0.985 ± 0.001	0.985 ± 0.001	0.500 ± 0.000	1.000 ± 0.001	0.985 ± 0.001	0.905 ± 0.017
Pattern 14	0.500 ± 0.000	0.980 ± 0.001	0.980 ± 0.001	0.500 ± 0.000	1.000 ± 0.000	0.980 ± 0.001	0.943 ± 0.014
Mean	0.534	0.971	0.971	0.536	0.987	0.972	0.899

Table S4: Mean AUROCs of all explainers across different whiteboxes (using readout function from Equation 4) when evaluated on positive examples.

	mean AUROC ↑						
No.	**GNN Expl.**	**Input x Grad.**	**IG**	**PG Expl.**	**SHAP Sampl.**	**Saliency**	**SubgraphX**
1	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.992 ± 0.002	0.500 ± 0.000	0.929 ± 0.011
2	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.980 ± 0.003	0.500 ± 0.000	0.957 ± 0.012
3	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.964 ± 0.004	0.500 ± 0.000	0.895 ± 0.027
4	0.500 ± 0.000	0.500 ± 0.000	0.992 ± 0.001	0.500 ± 0.000	0.975 ± 0.004	0.500 ± 0.000	0.793 ± 0.019
5	0.500 ± 0.000	0.500 ± 0.000	0.995 ± 0.003	0.500 ± 0.000	0.976 ± 0.007	0.500 ± 0.000	0.854 ± 0.031
6	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.982 ± 0.005	0.500 ± 0.000	0.898 ± 0.016
7	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.997 ± 0.002	0.500 ± 0.000	0.938 ± 0.014
8	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.988 ± 0.004	0.500 ± 0.000	0.889 ± 0.019
9	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.999 ± 0.001	0.500 ± 0.000	0.935 ± 0.013
10	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.997 ± 0.002	0.500 ± 0.000	0.874 ± 0.016
11	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.997 ± 0.002	0.500 ± 0.000	0.889 ± 0.018
12	1.000 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.824 ± 0.024
13	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.903 ± 0.018
14	0.500 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	1.000 ± 0.000	0.500 ± 0.000	0.942 ± 0.015
Mean	0.536	0.500	0.999	0.536	0.989	0.500	0.894

Table S5: Mean APs of all explainers across different whiteboxes (using readout function from Equation 3) when evaluated on positive examples.

	mean AP ↑							
No.	**Baseline**	**GNN Expl.**	**Input x Grad.**	**IG**	**PG Expl.**	**SHAP Sampl.**	**Saliency**	**SubgraphX**
1	0.406 ± 0.001	0.406 ± 0.001	0.995 ± 0.000	0.995 ± 0.000	0.406 ± 0.001	0.987 ± 0.003	0.995 ± 0.000	0.886 ± 0.018
2	0.581 ± 0.012	0.581 ± 0.012	0.998 ± 0.001	0.998 ± 0.001	0.581 ± 0.012	0.979 ± 0.003	0.998 ± 0.001	0.935 ± 0.018
3	0.678 ± 0.020	0.678 ± 0.020	1.000 ± 0.000	1.000 ± 0.000	0.678 ± 0.020	0.980 ± 0.003	1.000 ± 0.000	0.929 ± 0.022
4	0.462 ± 0.005	0.462 ± 0.005	0.980 ± 0.002	0.980 ± 0.002	0.462 ± 0.005	0.978 ± 0.004	0.984 ± 0.002	0.687 ± 0.031
5	0.587 ± 0.041	0.587 ± 0.041	0.995 ± 0.003	0.995 ± 0.002	0.587 ± 0.041	0.972 ± 0.008	0.995 ± 0.002	0.786 ± 0.048
6	0.294 ± 0.001	0.294 ± 0.001	0.928 ± 0.001	0.928 ± 0.001	0.294 ± 0.001	0.991 ± 0.004	0.928 ± 0.001	0.857 ± 0.026
7	0.297 ± 0.001	0.297 ± 0.001	0.975 ± 0.001	0.975 ± 0.001	0.297 ± 0.001	0.993 ± 0.004	0.975 ± 0.001	0.873 ± 0.028
8	0.270 ± 0.001	0.270 ± 0.001	0.959 ± 0.001	0.959 ± 0.001	0.270 ± 0.001	0.983 ± 0.006	0.959 ± 0.001	0.765 ± 0.035
9	0.296 ± 0.001	0.296 ± 0.001	0.977 ± 0.001	0.977 ± 0.001	0.296 ± 0.001	0.997 ± 0.001	0.977 ± 0.001	0.872 ± 0.024
10	0.266 ± 0.001	0.266 ± 0.001	0.837 ± 0.002	0.837 ± 0.002	0.266 ± 0.001	0.994 ± 0.003	0.837 ± 0.002	0.814 ± 0.025
11	0.210 ± 0.001	0.210 ± 0.001	0.915 ± 0.002	0.915 ± 0.002	0.210 ± 0.001	0.996 ± 0.003	0.915 ± 0.002	0.780 ± 0.032
12	0.145 ± 0.001	0.967 ± 0.001	0.967 ± 0.001	0.967 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	0.967 ± 0.001	0.657 ± 0.042
13	0.136 ± 0.001	0.136 ± 0.001	0.979 ± 0.001	0.979 ± 0.001	0.136 ± 0.001	0.999 ± 0.001	0.979 ± 0.001	0.720 ± 0.034
14	0.173 ± 0.001	0.173 ± 0.001	0.969 ± 0.001	0.969 ± 0.001	0.173 ± 0.001	1.000 ± 0.000	0.969 ± 0.001	0.868 ± 0.030
Mean	0.343	0.402	0.962	0.962	0.404	0.989	0.963	0.816

Table S6: Mean APs of all explainers across different whiteboxes (using readout function from Equation 4).

	mean AP ↑							
No.	**Baseline**	**GNN Expl.**	**Input x Grad.**	**IG**	**PG Expl.**	**SHAP Sampl.**	**Saliency**	**SubgraphX**
1	0.406 ± 0.001	0.406 ± 0.001	0.406 ± 0.001	1.000 ± 0.000	0.406 ± 0.001	0.991 ± 0.002	0.406 ± 0.001	0.872 ± 0.018
2	0.581 ± 0.012	0.581 ± 0.012	0.581 ± 0.012	1.000 ± 0.000	0.581 ± 0.012	0.983 ± 0.003	0.581 ± 0.012	0.933 ± 0.018
3	0.678 ± 0.020	0.678 ± 0.020	0.678 ± 0.020	1.000 ± 0.000	0.678 ± 0.020	0.978 ± 0.003	0.678 ± 0.020	0.911 ± 0.022
4	0.462 ± 0.005	0.462 ± 0.005	0.462 ± 0.005	0.991 ± 0.001	0.462 ± 0.005	0.967 ± 0.006	0.462 ± 0.005	0.682 ± 0.031
5	0.587 ± 0.041	0.587 ± 0.041	0.587 ± 0.041	1.000 ± 0.000	0.587 ± 0.041	0.984 ± 0.005	0.587 ± 0.041	0.783 ± 0.047
6	0.294 ± 0.001	0.294 ± 0.001	0.294 ± 0.001	1.000 ± 0.000	0.294 ± 0.001	0.976 ± 0.006	0.294 ± 0.001	0.846 ± 0.025
7	0.297 ± 0.001	0.297 ± 0.001	0.297 ± 0.001	1.000 ± 0.000	0.297 ± 0.001	0.996 ± 0.003	0.297 ± 0.001	0.870 ± 0.028
8	0.270 ± 0.001	0.270 ± 0.001	0.270 ± 0.001	1.000 ± 0.000	0.270 ± 0.001	0.981 ± 0.006	0.270 ± 0.001	0.764 ± 0.035
9	0.296 ± 0.001	0.296 ± 0.001	0.296 ± 0.001	1.000 ± 0.000	0.296 ± 0.001	0.999 ± 0.001	0.296 ± 0.001	0.869 ± 0.025
10	0.266 ± 0.001	0.266 ± 0.001	0.266 ± 0.001	1.000 ± 0.000	0.266 ± 0.001	0.996 ± 0.002	0.266 ± 0.001	0.814 ± 0.025
11	0.210 ± 0.001	0.210 ± 0.001	0.210 ± 0.001	1.000 ± 0.000	0.210 ± 0.001	0.996 ± 0.003	0.210 ± 0.001	0.784 ± 0.032
12	0.145 ± 0.001	1.000 ± 0.000	0.145 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.145 ± 0.001	0.658 ± 0.042
13	0.136 ± 0.001	0.136 ± 0.001	0.136 ± 0.001	1.000 ± 0.000	0.136 ± 0.001	1.000 ± 0.000	0.136 ± 0.001	0.717 ± 0.035
14	0.173 ± 0.001	0.173 ± 0.001	0.173 ± 0.001	1.000 ± 0.000	0.173 ± 0.001	1.000 ± 0.000	0.173 ± 0.001	0.867 ± 0.030
Mean	0.343	0.404	0.343	0.999	0.404	0.989	0.343	0.812

Table S7: Mean AUROCs of explainers across different whiteboxes evaluated on positive examples. Sub-columns r_1 and r_2 denote the readout functions utilized by the whitebox models (Equation 3 and Equation 4, respectively).

No.	GNN Expl.		Input × Grad.		IG		PG Expl.		SHAP Sampl.		Saliency		SubgraphX	
	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2
1	0.500	0.500	0.994	0.500	0.994	1.000	0.500	0.500	0.989	0.992	0.994	0.500	0.935	0.929
2	0.500	0.500	0.997	0.500	0.997	1.000	0.500	0.500	0.975	0.980	0.997	0.500	0.960	0.957
3	0.500	0.500	1.000	0.500	1.000	1.000	0.500	0.500	0.966	0.964	1.000	0.500	0.916	0.895
4	0.500	0.500	0.982	0.500	0.982	0.992	0.500	0.500	0.980	0.975	0.986	0.500	0.798	0.793
5	0.500	0.500	0.987	0.500	0.989	0.995	0.500	0.500	0.939	0.976	0.991	0.500	0.857	0.854
6	0.500	0.500	0.942	0.500	0.942	1.000	0.500	0.500	0.993	0.982	0.942	0.500	0.914	0.898
7	0.500	0.500	0.982	0.500	0.982	1.000	0.500	0.500	0.995	0.997	0.982	0.500	0.939	0.938
8	0.500	0.500	0.973	0.500	0.973	1.000	0.500	0.500	0.990	0.988	0.973	0.500	0.888	0.889
9	0.500	0.500	0.985	0.500	0.985	1.000	0.500	0.500	0.997	0.999	0.985	0.500	0.941	0.935
10	0.500	0.500	0.870	0.500	0.870	1.000	0.500	0.500	0.996	0.997	0.870	0.500	0.874	0.874
11	0.500	0.500	0.939	0.500	0.939	1.000	0.500	0.500	0.997	0.997	0.939	0.500	0.888	0.889
12	0.980	1.000	0.980	0.500	0.980	1.000	1.000	1.000	1.000	1.000	0.980	0.500	0.824	0.824
13	0.500	0.500	0.985	0.500	0.985	1.000	0.500	0.500	1.000	1.000	0.985	0.500	0.905	0.903
14	0.500	0.500	0.980	0.500	0.980	1.000	0.500	0.500	1.000	1.000	0.980	0.500	0.943	0.942
Mean	0.534	0.536	0.971	0.500	0.971	0.999	0.536	0.536	0.987	0.989	0.972	0.500	0.899	0.894

Table S8: Mean Average Precision (AP) of explainers across different whiteboxes evaluated on positive examples. Sub-columns r_1 and r_2 denote the readout functions utilized by the whitebox models (Equation 3 and Equation 4, respectively). Standard errors are omitted for brevity. Cell colors represent a piecewise continuous gradient based on landmark colors: **Red** (≤ 0.50), **Pale Red** (≈ 0.65), **Light Green** (≈ 0.85), and **Green** (1.00).

No.	Baseline	GNN Expl.		Input × Grad.		IG		PG Expl.		SHAP Sampl.		Saliency		SubgraphX	
		r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2	r_1	r_2
1	0.406	0.406	0.406	0.995	0.406	0.995	1.000	0.406	0.406	0.987	0.991	0.995	0.406	0.886	0.872
2	0.581	0.581	0.581	0.998	0.581	0.998	1.000	0.581	0.581	0.979	0.983	0.998	0.581	0.935	0.933
3	0.678	0.678	0.678	1.000	0.678	1.000	1.000	0.678	0.678	0.980	0.978	1.000	0.678	0.929	0.911
4	0.462	0.462	0.462	0.980	0.462	0.980	0.991	0.462	0.462	0.978	0.967	0.984	0.462	0.687	0.682
5	0.587	0.587	0.587	0.995	0.587	0.995	1.000	0.587	0.587	0.972	0.984	0.995	0.587	0.786	0.783
6	0.294	0.294	0.294	0.928	0.294	0.928	1.000	0.294	0.294	0.991	0.976	0.928	0.294	0.857	0.846
7	0.297	0.297	0.297	0.975	0.297	0.975	1.000	0.297	0.297	0.993	0.996	0.975	0.297	0.873	0.870
8	0.270	0.270	0.270	0.959	0.270	0.959	1.000	0.270	0.270	0.983	0.981	0.959	0.270	0.765	0.764
9	0.296	0.296	0.296	0.977	0.296	0.977	1.000	0.296	0.296	0.997	0.999	0.977	0.296	0.872	0.869
10	0.266	0.266	0.266	0.837	0.266	0.837	1.000	0.266	0.266	0.994	0.996	0.837	0.266	0.814	0.814
11	0.210	0.210	0.210	0.915	0.210	0.915	1.000	0.210	0.210	0.996	0.996	0.915	0.210	0.780	0.784
12	0.145	0.967	1.000	0.967	0.145	0.967	1.000	1.000	1.000	1.000	1.000	0.967	0.145	0.657	0.658
13	0.136	0.136	0.136	0.979	0.136	0.979	1.000	0.136	0.136	0.999	1.000	0.979	0.136	0.720	0.717
14	0.173	0.173	0.173	0.969	0.173	0.969	1.000	0.173	0.173	1.000	1.000	0.969	0.173	0.868	0.867
Mean	0.343	0.402	0.404	0.962	0.343	0.962	0.999	0.404	0.404	0.989	0.989	0.963	0.343	0.816	0.812

H ADDITIONAL DATASET INFORMATION

H.1 PATTERNS

Pattern statistics. The KRFP fingerprint of Klekota & Roth (2008) contains 4,860 patterns, drawn from substructures enriched in biologically active compounds. 1,655 of them (34.1%) don't contain rings, and therefore are eligible for inclusion. Each whitebox requires manual weight construction and validation against more than five million negatives. This construction and validation effort limited our selection to 14 motifs. We did not sample the admissible set for diversity; we took patterns that qualified and varied their size deliberately. This coverage is not complete: no motif contains an alcohol, a primary amine, a nitrile or a urea. Ring-containing motifs are excluded by construction.

Pattern SMARTS. Here, we present KRFP SMARTS and their relaxed versions. Note that some cases do not require relaxation, as in simple cases node identity requirements can be fulfilled by neighbour counting. This relaxation allows our whiteboxes to be fully reliable, while recognizing complex patterns. See Table S9 for analysis of disagreements between the original and relaxed SMARTS.

1. `[!#1][CH]([NH]C(=O)[!#1])C(Cl)(Cl)Cl` → `[CHD3]([NH][CD3](=O))C(Cl)(Cl)Cl`
2. `[!#1][CH]([!#1])C(=O)[CH2]C([CH3])([CH3])[CH2]C([!#1])([!#1])[!#1]` → `[CHD3]C(=O)[CH2]C([CH3])([CH3])[CH2][CD4]`
3. `[!#1][CH]([!#1])C(=O)[NH][NH]C(=O)[CH2][CH2][CH2][CH2][CH2][CH2][CH3]` → `[CHD3]C(=O)[N;H1,H2][N;H1,H2]C(=O)[CH2][CH2][CH2][CH2][CH2][CH2][CH3]`
4. `[!#1][CH]([!#1])C(=O)O[CH2][CH2]N([CH2][CH3])[CH2][CH3]` → `[CHD3]C(=O)O[CH2][CH2]N([CH2][CH3])[CH2][CH3]`
5. `[!#1][CH]([CH2]C(=N[NH]C(=O)[!#1])[!#1])C(=O)[OH]` → `[CHD3]([CH2][CD3](=N[N;H1,H2][CD3](=O)))C(=O)[OH]`
6. `[!#1][CH]([!#1])[CH]([!#1])[!#1]` → `[CHD3][CHD3]`
7. `[!#1][CH]([!#1])[CH2][CH2][CH3]` → `[CHD3][CH2][CH2][CH3]`
8. `[!#1][CH]([!#1])[CH2]C(=O)[!#1]` → `[CHD3][CH2][CD3](=O)`
9. `[!#1][CH]([!#1])S(=O)(=O)[!#1]` → `[CHD3][SD4](=O)(=O)`
10. `[!#1][CH]([!#1])[!#1]` → `[!#1][CH]([!#1])[!#1]`
11. `[!#1][CH]([!#1])[CH3]` → `[!#1][CH]([!#1])[CH3]`
12. `[!#1][CH]=O` → `[!#1][CH]=O`
13. `[!#1][N+]([CH3])([CH3])[CH3]` → `[!#1][N+]([CH3])([CH3])[CH3]`
14. `[!#1][CH2]Cl` → `[!#1][CH2]Cl`

H.2 VALIDATION MOLECULES

In this section, we present additional statistics and details regarding our validation datasets. Figure S11 illustrates the structural similarity distributions relative to ECFP4 motif fingerprints, while Table S10 details the validation dataset sizes for each motif.

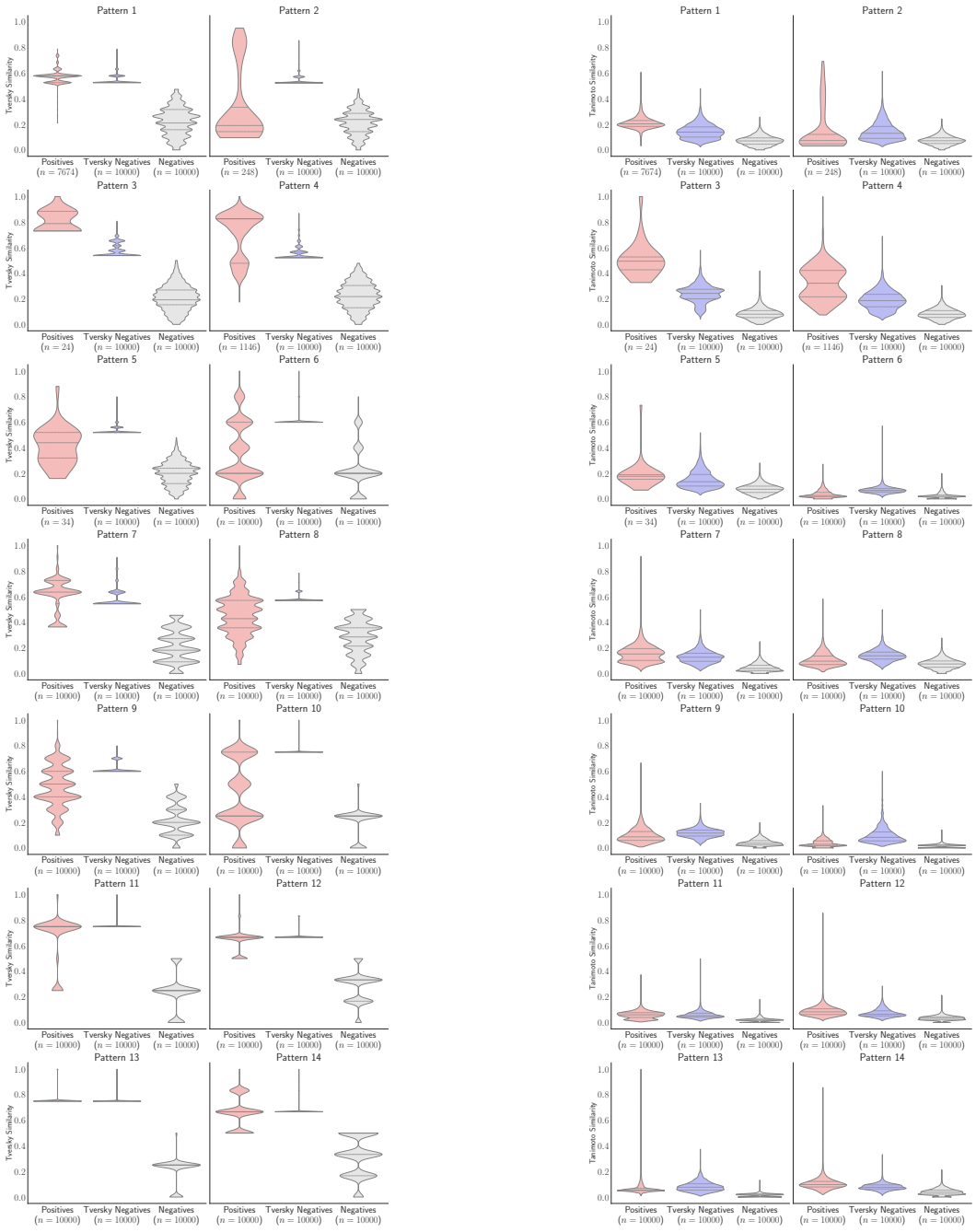


Figure S11: Structural similarity distributions relative to ECFP4 motif fingerprints. Similarity scores (Tversky and Tanimoto) for molecules used to all of our patterns.

Table S9: Fractions of molecules for which the original KRFP SMARTS and our relaxed SMARTS disagree either in motif detection or in the identified atom set.

Pattern No.	Disagreements	Total	%
1	1	7,674	0.01
2	184	248	74.19
3	0	24	0.00
4	2	1,146	0.17
5	10	34	29.41
6	591	10,000	5.91
7	0	10,000	0.00
8	106	10,000	1.06
9	5	10,000	0.05
10	0	10,000	0.00
11	0	10,000	0.00
12	0	10,000	0.00
13	0	10,000	0.00
14	0	10,000	0.00
Total	899	99,126	0.91

Table S10: In this table, we showcase the sizes of the validation sets for each of our patterns. Molecule counts under the "All molecules" field show the factual size of our validation dataset that was prepared by us by filtering PubChem. Unfortunately, a small subset of SMILES strings was corrupted during preprocessing; as such, the "Valid molecules" field shows on how many molecules each of the whiteboxes was actually tested. Note that there is very little difference between the two subsets; however, we've chosen to compare these numbers for transparency.

Pattern No.	**All molecules**			**Valid molecules**		
	Positives	**Tversky**	**Negatives**	**Positives**	**Tversky**	**Negatives**
1	7,674	491,447	4,976,465	7,674	491,447	4,976,452
2	248	118,672	4,994,354	248	118,672	4,994,352
3	24	1,568,016	4,925,632	24	1,568,016	4,925,626
4	1,146	569,455	4,972,881	1,146	569,455	4,972,877
5	34	70,609	4,996,581	34	70,609	4,996,571
6	5,000,000	5,000,000	4,729,159	4,999,997	4,999,997	4,729,155
7	886,073	4,346,189	4,792,735	886,073	4,346,189	4,792,729
8	3,376,760	2,866,888	4,859,902	3,376,759	2,866,887	4,859,894
9	659,042	1,685,436	4,919,363	659,042	1,685,436	4,919,355
10	5,000,000	13,491	4,998,512	4,999,998	13,491	4,998,501
11	5,000,000	241,794	4,985,482	4,999,995	241,794	4,985,478
12	1,970,537	717,424	4,965,399	1,970,527	717,424	4,965,389
13	286,098	127,085	4,993,947	286,098	127,085	4,993,942
14	894,517	3,950,564	4,812,155	894,517	3,950,564	4,812,149