# DOES YOUR GRAPH LEARNING TASK REALLY REQUIRE MORE EXPRESSIVE GNNS ?

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Whether a Message Passing Neural Network (MPNN) is expressive enough for a given graph learning task is typically answered post hoc, by comparing model performance on a held-out set. We argue that this question should instead be answered a priori, as a property of the dataset. We formalize three dataset-level conditions: refinability, 1-WL separation within the dataset, and amenability. We show that each condition gives theoretical guarantees on the performance of MPNNs, but that 1-WL separation, the property that collision counts measure, depends on the whole dataset, can break with one new graph, and says nothing about nodes. We prove that amenability is the right notion: on an amenable family, a GIN with enough message-passing layers universally approximates any continuous permutation-invariant target, at both graph and node level, the guarantee holds on train and test together, and the fraction of non-amenable graphs bounds the loss of any MPNN. Amenability is a per-graph property, testable in quasi-linear time, and our implementation checks millions of graphs in minutes. Empirically, we find that more than $99\%$ of molecular graphs and more than $98\%$ of their $k$-hop neighborhoods are amenable, and that a 1-WL approximation of random-walk positional encodings gives the same gain as the encodings themselves, suggesting that MPNNs are expressive enough for typical molecular property prediction tasks.

## 1 INTRODUCTION

Graph neural networks (GNNs) are the standard tool for learning on graph-structured data, *e.g.*, molecules and social networks. The dominant family, message-passing neural networks (MPNNs), iteratively updates each node's representation from its neighbors; after $L$ rounds, every node has aggregated information from its $L$-hop neighborhood. This template covers most architectures of the field: GCN (Kipf & Welling, 2017), GIN (Xu et al., 2019), GAT (Veličković et al., 2018), etc.

A foundational line of work has tied the expressive power of MPNNs (which graphs they can tell apart) to a classical algorithm from combinatorics: the 1-Weisfeiler-Leman test (equivalently color refinement test). Xu et al. (2019) and Morris et al. (2019) showed independently that an MPNN can distinguish two graphs if and only if WL-1 distinguishes them, making WL-1 a tight upper bound on MPNN expressivity. Every claim that an architecture *is more expressive than an MPNN* is a claim that it goes beyond WL-1, and every architectural innovation aimed at expressivity (higher-order GNNs, subgraph GNNs, structural and positional encodings) is justified by exhibiting a WL-1 failure case it can resolve.

This existing expressivity literature is overwhelmingly model-centric: it asks how powerful a class of architectures is, and characterizes that power in terms of the WL hierarchy. A dataset enters only implicitly, through the assumption that worst-case graph instances may appear. Universal approximation theorems for GNNs (Keriven & Peyré, 2019; Chen et al., 2019; Azizian & Lelarge, 2021) confirm this picture: to approximate every continuous permutation-invariant target on every graph of size $n$, one needs $k$-WL power for $k$ growing with $n$.

The dataset-side perspective has been largely absent. Choosing among these architectures on a real dataset is, today, a purely empirical exercise: the question *is my dataset hard enough to need this expressivity?* has no definitive answer before training. An initial investigation by Zopf (2022) reports few color-refinement collisions across several benchmarks. In this work, we show that such



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.

collision-based diagnostics are insufficient, both as a characterization of GNNs behavior on real data and as a basis for theoretical guarantees in machine learning.

We argue that the right property to test is *amenability*. It is checked one graph at a time, it certifies that a plain MPNN has enough expressive power on a graph, and the fraction of graphs that fail it bounds the loss any MPNN can reach.

**Contributions** We introduce a framework for testing whether MPNNs have sufficient expressivity on a given dataset. Specifically:

1. **Amenability as a dataset-side test of MPNN expressivity.** We show that CR separation, the property that collision counts measure, is the wrong test: it depends on the whole dataset, one new graph can break it, and it says nothing about nodes. Amenability, a graph property introduced by Arvind et al. (2015) and Kiefer et al. (2015), fixes all three. It gives universal approximation for MPNNs at both graph and node level, it holds on train and test together, and the non-amenable fraction bounds the loss of any MPNN.
2. **A fast, practical test.** We give an algorithm that checks amenability in quasi-linear time per graph and handles millions of graphs in minutes.
3. **Empirical validation on molecular datasets.** More than $99\%$ of molecules are amenable, and so are more than $98\%$ of the $k$-hop neighborhoods around each atom. A 1-WL approximation of Random Walk Positional Encodings gives the same gain as full RWPE, so that gain comes from inductive bias, not expressivity.

We believe these results will be valuable to the GNN community on two fronts. For practitioners, testing amenability is a fast diagnostic of whether a dataset needs more than an MPNN. For theory, it changes how we read the success of higher-order architectures: on amenable datasets, expressivity cannot explain their gains, and the question becomes which mechanism does.

## 2 PRELIMINARIES

### 2.1 GRAPH-THEORETIC PRELIMINARIES

**Attributed graph.** An *attributed graph* is a tuple $G = (V, E, X, X_E)$ where $V = \{1, \dots, n\}$ is the vertex set, $E \subseteq V \times V$ is the edge set, $X \in \mathbb{R}^{n \times d}$ is the node feature matrix, and $X_E \in \mathbb{R}^{|E| \times d'}$ is the edge feature matrix. We write $X(v)$ for the feature of node $v$, $X_E(u, v)$ for the feature of edge $\{u, v\}$, and $N(v)$ for the set of neighbours of $v$. We write $\mathcal{G}_n$ for the set of attributed graphs on at most $n$ vertices with node features in a compact $\mathcal{X} \subseteq \mathbb{R}^d$ and edge features in a compact $\mathcal{X}_E \subseteq \mathbb{R}^{d'}$. A *dataset* is a subset $\mathcal{D} \subseteq \mathcal{G}_n$.

**Graph isomorphism.** An *isomorphism* between graphs $G$ and $H$ is a bijection $\varphi : V(G) \to V(H)$ that preserves adjacency, $\{u, v\} \in E(G) \iff \{\varphi(u), \varphi(v)\} \in E(H)$, and all features, $X_G(v) = X_H(\varphi(v))$ and $X_{E,G}(u, v) = X_{E,H}(\varphi(u), \varphi(v))$. We write $G \cong H$ when such a $\varphi$ exists.

**Graph automorphism and orbits.** An *automorphism* of $G$ is an isomorphism $G \to G$. The *automorphism group* $\mathrm{Aut}(G)$ acts on $V$, partitioning it into *orbits* $\mathrm{Orb}(v) = \{\varphi(v) : \varphi \in \mathrm{Aut}(G)\}$. Two nodes in the same orbit are *structurally equivalent*.

**Color refinement.** *Color refinement* (CR), also known as the 1-dimensional Weisfeiler–Leman test (WL-1), is the iterative vertex coloring

$$c^{(0)}(v) = X(v),$$
$$c^{(t+1)}(v) = \mathrm{HASH}\Big(c^{(t)}(v),\ \{\!\{(c^{(t)}(u),\ X_E(v, u)) : u \in N(v)\}\!\}\Big),$$

with HASH injective. The coloring stabilizes in at most $n$ steps; we write $\mathrm{CR}(v)$ for the stable color of $v$, $t_{\mathrm{CR}}(G)$ for the convergence depth, and $\mathrm{CR}_k(v)$ for the color after $k$ rounds. The *fingerprint* of $G$ is the multiset $\mathrm{CR}(G) = \{\!\{\mathrm{CR}(v) : v \in V\}\!\}$, and we write $G \sim_{\mathrm{CR}} H$ iff $\mathrm{CR}(G) = \mathrm{CR}(H)$. Automorphisms preserve CR colors, so every orbit lies inside a single CR cell.

**Confuser.** A graph $H$ is a *confuser* of $G$ if $H \not\cong G$ but $H \sim_{\mathrm{CR}} G$.

**Refinable graph.** A graph $G$ is *refinable* if $\mathrm{CR}(u) = \mathrm{CR}(v) \iff \mathrm{Orb}(u) = \mathrm{Orb}(v)$ for all $u, v \in V(G)$, *i.e.* CR assigns distinct colors to structurally different nodes.

**Amenable graph.** A graph $G$ with $n$ nodes is *amenable* if for all $H \in \mathcal{G}_n$, $\mathrm{CR}(G) = \mathrm{CR}(H) \iff G \cong H$. Equivalently, $G$ has no confuser. Amenability implies refinability (Arvind et al., 2015).

*Remark* 1 (Refinable $\not\Rightarrow$ amenable). The converse implication fails.

**Random-walk positional encoding.** The *random-walk positional encoding* (RWPE) of length $K$ assigns to each node $v$ the vector of return probabilities $\mathrm{RWPE}_K(v) = \big((D^{-1}A)^k_{vv}\big)_{k=1}^{K} \in \mathbb{R}^K$, where $D$ is the degree matrix (Dwivedi et al., 2022b).

### 2.2 Message-passing neural networks

**MPNN.** A *message-passing neural network* computes node embeddings via $K$ rounds of neighborhood aggregation,

$$h_v^{(t+1)} \;=\; \mathrm{UPD}^{(t)}\Big(h_v^{(t)},\; \mathrm{AGG}^{(t)}\big(\{\!\{(h_u^{(t)}, X_E(v,u)) : u \in N(v)\}\!\}\big)\Big), \quad h_v^{(0)} = X(v),$$

followed by a permutation-invariant readout $\Phi(G) = \mathrm{READOUT}(\{\!\{h_v^{(K)} : v \in V\}\!\})$.

**Theorem 1** (MPNN–CR equivalence)**.** *For any MPNN $\Phi$ with $K$ layers, $\mathrm{CR}_K(G) = \mathrm{CR}_K(H) \implies \Phi(G) = \Phi(H)$. GIN with injective aggregation and sum readout saturates this bound: its discriminative power equals that of $K$-step color refinement (Xu et al., 2019).*

## 3 Amenability as a dataset-side test of MPNN expressivity

An MPNN sees a dataset only through color refinement (Theorem 1), so whether it has enough expressive power is a property of the dataset and not of the architecture alone. If a dataset is amenable, a plain MPNN of known, measurable depth is not the bottleneck; and where amenability fails, the non-amenable fraction is exactly what caps the achievable loss.

### 3.1 Why amenability is the right condition

We compare three conditions on a dataset $\mathcal{D} \subseteq \mathcal{G}_n$. $P_1$ and $P_2$ measure different things, one inside each graph and one across graphs, and neither implies the other. Amenability gives both.

**Refinability ($P_1$): the node-level view.** $\mathcal{D}$ satisfies $P_1$ if every $G \in \mathcal{D}$ is refinable (Section 2.1): CR gives two nodes the same color only when an automorphism maps one onto the other. After $t_{\mathrm{CR}}(\mathcal{D}) = \sup_{G\in\mathcal{D}} t_{\mathrm{CR}}(G)$ iterations, an MPNN then separates every pair of structurally different nodes, which is what a node-level prediction head needs (Theorem 4). But $P_1$ looks at one graph at a time and says nothing about how graphs compare: $C_6$ and $2K_3$ are both refinable, yet they share a CR fingerprint (Remark 1), so no MPNN can tell them apart.

**CR separation ($P_2$): the dataset view, and its limits.** $\mathcal{D}$ satisfies $P_2$ if CR separates every pair of non-isomorphic graphs of $\mathcal{D}$: $G \not\cong H \implies \mathrm{CR}(G) \neq \mathrm{CR}(H)$. Write $k_2(\mathcal{D})$ for the smallest number of iterations at which this holds ($\infty$ if it never does). $P_2$ is what graph-level approximation needs: on a dataset that satisfies it, a GIN with $k_2(\mathcal{D})$ layers approximates any continuous permutation-invariant target, by the equivalence between isomorphism testing and function approximation of Chen et al. (2019), as stated in Theorem 6. However, it has three drawbacks, and each one disqualifies it as a diagnostic.

- *It is relative to $\mathcal{D}$.* $P_2$ is a property of a set of graphs, not of any graph in it, and one new graph can break it (Proposition 1). A check on the training set says nothing about the test set.
- *It is pairwise.* Checking it means comparing fingerprints across the whole dataset and resolving every collision with an isomorphism test.

- *It is blind to nodes.* $P_2$ does not imply $P_1$ (Appendix A.2), and $k_2(\mathcal{D})$ falls short of the depth that node-level separation needs.

**Amenability ($P_3$): no confuser anywhere.** $\mathcal{D}$ satisfies $P_3$ if every $G \in \mathcal{D}$ is amenable (Section 2.1): no graph of $\mathcal{G}_n$ is a confuser of $G$. The notion comes from the graph isomorphism literature (Arvind et al., 2015; Kiefer et al., 2015). We bring it to machine learning because it is the absolute version of $P_2$. $P_2$ asks that no graph of $\mathcal{D}$ collide with another graph of $\mathcal{D}$. $P_3$ asks that no graph of $\mathcal{D}$ collide with any graph at all. This one change removes the three drawbacks. Amenability is a property of a single graph, so it is checked one graph at a time, in quasi-linear time (Section 4.3). It survives unions with other amenable sets. And it implies refinability, so the node-level view of $P_1$ comes with it: at depth $t_{\mathrm{CR}}(\mathcal{D})$, no two structurally different nodes share an embedding. Under $P_2$ alone they can, and no prediction head can recover what the merge lost.

**Proposition 1** (Amenability gives $P_1$ and $P_2$ on train and test together)**.**

1. *If $\mathcal{D}$ satisfies $P_3$, then it satisfies $P_1$ and $P_2$, with $k_2(\mathcal{D}) \le t_{\mathrm{CR}}(\mathcal{D})$.*
2. *If $\mu$ is a distribution on $\mathcal{G}_n$ supported on amenable graphs, then any $\mathcal{D}_{\mathrm{train}}$ and $\mathcal{D}_{\mathrm{test}}$ sampled from $\mu$ satisfy $P_3$ together, and therefore $P_1$ and $P_2$ on $\mathcal{D}_{\mathrm{train}} \cup \mathcal{D}_{\mathrm{test}}$. This fails for $P_2$: there is a $\mu$ under which $\mathcal{D}_{\mathrm{train}}$ satisfies $P_2$ but $\mathcal{D}_{\mathrm{train}} \cup \mathcal{D}_{\mathrm{test}}$ violates it with positive probability.*

In practice the support of $\mu$ is unknown and amenability is measured on a sample. Because it is checked graph by graph, the amenable fraction of $\mathcal{D}_{\mathrm{train}}$ is an unbiased estimate of $\Pr_\mu(G \text{ is amenable})$, the quantity $p$ that bounds the loss in Corollary 1. $P_2$ is a property of the whole set and has no such per-graph estimate.

### 3.2 What non-amenable graphs cost

Let $\mu$ be a probability distribution on $\mathcal{D} \subseteq \mathcal{G}_n$, let $y : \mathcal{G}_n \to \mathbb{R}$ be a continuous permutation-invariant target[1], and take the squared loss $\ell(a,b) = (a-b)^2$ with $\mathcal{L}(f) = \mathbb{E}_{G\sim\mu}\,\ell(f(G), y(G))$. Write $[G]_{\mathrm{CR}} = \{H \in \mathcal{D} : H \sim_{\mathrm{CR}} G\}$ for the CR class of $G$, and $\mathcal{F}_{\mathrm{MPNN}}$ for the continuous permutation-invariant functions on $\mathcal{D}$ that are constant on each CR class. Every MPNN lies in $\mathcal{F}_{\mathrm{MPNN}}$ by Theorem 1, so the infimum risk over $\mathcal{F}_{\mathrm{MPNN}}$ lower-bounds what any MPNN attains, and Theorem 6 makes that bound tight.

**Theorem 2** (CR-class risk decomposition)**.** *The infimum population risk over all MPNNs is*

$$\inf_{f\in\mathcal{F}_{\mathrm{MPNN}}} \mathcal{L}(f) \;=\; \mathbb{E}_{G\sim\mu}\Big[\,\mathrm{Var}\big(y(H)\,\big|\,[H]_{\mathrm{CR}} = [G]_{\mathrm{CR}}\big)\,\Big],$$

*attained almost everywhere only by $f^\star(G) = \mathbb{E}[\,y(H) \mid [H]_{\mathrm{CR}} = [G]_{\mathrm{CR}}\,]$.*

**Corollary 1** (Amenability bound)**.** *Let $p = \Pr_{G\sim\mu}(G \text{ is amenable})$. On an amenable class the conditional variance vanishes, since the class is a single isomorphism type and $y$ is permutation-invariant. Hence*

$$\inf_{f\in\mathcal{F}_{\mathrm{MPNN}}} \mathcal{L}(f) = (1-p)\,\mathbb{E}\big[\,\mathrm{Var}(y \mid [G]_{\mathrm{CR}})\,\big|\,G \text{ not amenable}\,\big] \le (1-p) \sup_{[G]\ \text{non-amenable}} \mathrm{Var}(y \mid [G]).$$

**Corollary 2** (Pointwise prediction at confusers)**.** *For any $G \in \mathcal{G}_n$ and any $f \in \mathcal{F}_{\mathrm{MPNN}}$,*

$$f(G) = \frac{1}{\#\{H \in \mathcal{G}_n : H \sim_{CR} G\}} \sum_{\substack{H\in\mathcal{G}_n \\ H\sim_{CR} G}} f(H),$$

*and the same identity holds for $\nabla_\theta f_\theta$ when $f_\theta$ is a parametrized MPNN.*

The loss floor shrinks with the non-amenable mass $1-p$, which is the quantity we measure on molecular benchmarks. Confusers also hurt when they never appear in training: by Corollary 2 an MPNN averages over the whole CR class, so it cannot tie a structural pattern carried by one member of a confuser pair to the target, and the same averaging applies to its gradients.

On amenable graphs the question is closed. CR is injective on isomorphism classes there (Theorem 7), so no positional or structural encoding can carry information that CR has not already extracted (Corollary 3). Our RWPE experiments test this prediction directly.

[1] Think of $y$ as the best function we could hope to learn, for instance one computed by a higher-order GNN.

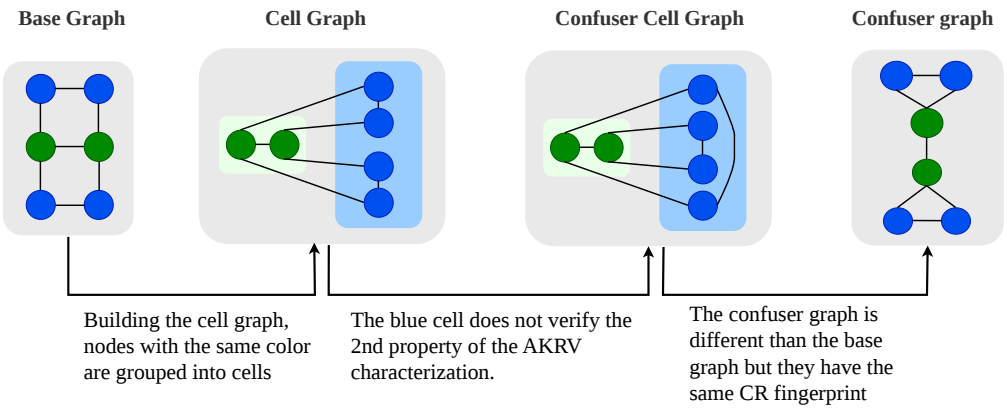



Figure 1: Example of a non-amenable graph. We illustrate the construction of the cell graph from the CR coloring and show how violating the amenability characterization of Arvind et al. (2015) yields a confuser graph.

**Why not just count collisions.** Zopf (2022) counts CR fingerprint collisions, which tests $P_2$ on the dataset at hand. The count does not carry over to new data: one new graph can break $P_2$ (Proposition 1), whereas the amenable fraction estimates $p$ and bounds the loss on unseen graphs (Corollary 1). The depth it certifies, $k_2$, is too shallow for node-level separation, which needs $t_{\mathrm{CR}}$ (Theorem 4). And every collision, including every pair of isomorphic duplicates, must be resolved by an isomorphism test, whereas amenability is one $O((n+m)\log n)$ check per graph.

## 4 An algorithm for efficient verification of amenability of a dataset

In this section we consider simple graphs without node features or edge features. Our implementation is based on the cell-graph language of Arvind et al. (2015) and uses results from Kiefer et al. (2015) to justify running amenability tests on attributed graphs. On a dataset of $N$ graphs with at most $n$ nodes and $m$ edges each, property $P_3$ is verified in $O(N(n+m)\log n)$ time, the same asymptotic cost as running color refinement itself.

### 4.1 The cell graph

The color refinement procedure ends with a stable partition $P_G = \{X_1, \ldots, X_r\}$ of $V(G)$ into *cells*: two vertices sit in the same cell iff CR cannot tell them apart. Two vertices $u, v$ in the same cell $X$ must have the same multiset of neighbor colors:

1. Every vertex of $X$ has the same number of neighbors inside $X$ itself; in other words, $G[X]$ is a $d_X$-regular graph for some integer $d_X \geq 0$.
2. For every other cell $Y$, every vertex of $X$ has the same number of neighbors in $Y$, and every vertex of $Y$ has the same number of neighbors in $X$; in other words, $G[X, Y]$ is biregular with some degree pair $(d_{XY}, d_{YX})$.

The only information CR retains about $G$ is the cell sizes $|X_i|$, the intra-cell shapes $G[X_i]$ (each one regular), and the inter-cell shapes $G[X_i, X_j]$ (each one biregular). The cell graph repackages this information as a labeled graph on the cells.

**Definition 1** (Cell graph)**.** The *cell graph* $\mathcal{C}(G)$ is the complete graph with vertex set $P_G$, where each vertex (cell) and each edge (pair of cells) carries a structural label:

- a cell $X \in P_G$ is *homogeneous* if $G[X]$ is empty ($d_X = 0$) or complete ($d_X = |X| - 1$), and *heterogeneous* otherwise;

- an edge $\{X, Y\}$ is *isotropic* if $G[X, Y]$ is empty ($d_{XY} = 0$) or complete bipartite ($d_{XY} = |Y|$), and *anisotropic* otherwise.

**Definition 2** (Anisotropic component)**.** An *anisotropic component* of $\mathcal{C}(G)$ is a maximal connected subgraph using only anisotropic edges.

**Why this is the right object.** A cell is homogeneous when its internal structure is trivial, nothing inside it or everything inside it, and an edge is isotropic when the connection between two cells is similarly trivial, no edges between them, or all possible edges. In both cases the cell sizes alone force the shape, leaving no room for a non-isomorphic alternative.

Anisotropic edges and heterogeneous cells are exactly the places where two graphs can differ in structure while sharing the same equitable partition. Testing amenability therefore amounts to checking that these "free" parts of $\mathcal{C}(G)$ are constrained enough to admit only one graph up to isomorphism.

### 4.2 The characterization theorem

The following is the central technical result of Arvind et al. (2015).

**Theorem 3** (AKRV characterization)**.** *A graph $G$ is amenable if and only if its cell graph $\mathcal{C}(G)$ satisfies the following four conditions:*

1. *For every cell $X \in P_G$, the induced subgraph $G[X]$ is empty, complete, a matching $mK_2$, the complement of a matching $\overline{mK_2}$, or the 5-cycle $C_5$.*
2. *For every pair of distinct cells $X, Y \in P_G$, the induced bipartite graph $G[X, Y]$ is empty, complete bipartite, a disjoint union of stars $sK_{1,t}$, or the bipartite complement of $sK_{1,t}$.*
3. *Every anisotropic component $\mathcal{A}$ of $\mathcal{C}(G)$ is a tree. Rooted at a cell of minimum cardinality in $\mathcal{A}$, every directed edge $(X, Y)$ satisfies $|X| \leq |Y|$.*
4. *Every anisotropic component contains at most one heterogeneous cell, and if it does, that cell has minimum cardinality in the component.*

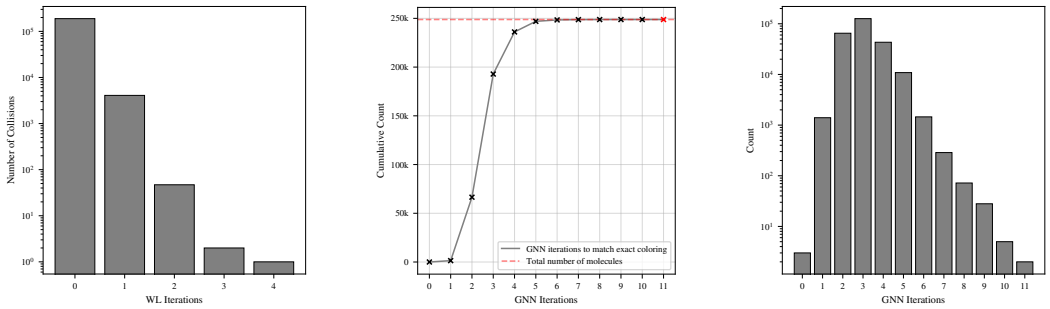
### 4.3 Algorithm and complexity

The check takes $O((n+m)\log n)$ time (Arvind et al., 2015); Appendix B.1 gives the full procedure and its complexity analysis.

Most datasets of interest are attributed, whereas the results of Arvind et al. (2015) apply only to unattributed graphs. However, given an attributed graph $(V, E, X, X_E)$, running the same algorithm on $(V, E)$ but with color refinement performed on the full attributed graph still yields a sound test: if the algorithm declares $G$ amenable, then $G$ is amenable.

Table 1: 1-WL is essentially sufficient on common molecular benchmarks. CR distinguishes every graph, a random MPNN already separates the vast majority, and most graphs are amenable. $k_2(\mathcal{D})$ is the number of CR iterations needed to separate every pair of graphs ($P_2$), and $t_{\mathrm{CR}}(\mathcal{D})$ the number needed for CR to stabilize on every graph, the depth at which an MPNN separates all non-equivalent nodes of an amenable graph. Both are maxima over all graphs of the dataset. Random GNN separation is a graph-level measure: the fraction of molecules whose graph embedding, computed by a randomly initialized MPNN, differs from the embedding of every other molecule of the dataset.

| | #Graphs | CR Separation | Random GNN Separation | Amenability | $k_2(\mathcal{D})$ | $t_{\mathrm{CR}}(\mathcal{D})$ |
|---|---|---|---|---|---|---|
| **ZINC-250k** | 249,456 | 100.00% | 99.44% | 99.38% | 5 | 11 |
| **AQSOL** | 9,823 | 100.00% | 92.36% | 91.39% | 7 | 12 |
| **PCQM4Mv2** | 3,452,118 | 100.00% | 87.12% | 99.47% | 6 | 12 |
| **Aggregate** | 3,711,397 | 100.00% | 87.96% | 99.44% | 7 | 12 |




(a) Number of 1-WL fingerprint collisions across the dataset as a function of the number of color-refinement iterations.

(b) Cumulative number of molecules whose nodes are all distinguished by a random GNN with at most $k$ layers.

(c) Distribution of the minimum GNN depth required to distinguish every node within a molecule.

Figure 2: Empirical depth required to satisfy properties $P_1$ (node-level distinguishability) and $P_2$ (graph-level distinguishability) on ZINC-250k. Similar figures are available in Appendix D for the other datasets.

## 5 From theory to experiment: a case study on molecular graphs

Molecular property prediction is one of the main applications of GNNs (Gilmer et al., 2017), and benchmarks such as ZINC, AQSOL and PCQM4Mv2 (Dwivedi et al., 2022a; Hu et al., 2020) are where new architectures are compared. Many of the leading methods on them go beyond MPNNs and 1-WL, through more expressive positional or structural encodings (Dwivedi et al., 2022b; Bouritsas et al., 2023; Lim et al., 2023) or higher-order architectures (Morris et al., 2019; Hussain et al., 2024; Bodnar et al., 2021). We use amenability to ask whether their gains come from this extra expressivity, on ZINC-250k, AQSOL and PCQM4Mv2 (3.7M molecules). On molecules, the answer is largely no. Per-split results are in Appendix D.

**Summary of findings.**

1. $99.4\%$ of molecules are amenable: a plain MPNN is not the bottleneck on these benchmarks.
2. CR separates every pair of molecules, yet about $21{,}000$ of them are not amenable: testing separation on the dataset, as Zopf (2022) does, misses real confusers.
3. Node-level separation needs $t_{\mathrm{CR}} = 11$ to 12 layers, about twice the depth that separates graphs, and a random MPNN reaches it.
4. Local neighborhoods are amenable at every radius up to 12, so local encodings reduce to CR on a small ball.
5. An MPNN that learns RWPE reproduces the gain of RWPE: the gain comes from inductive bias, not expressivity.

**Finding 1: almost every molecule is amenable.** We run the test of Section 4.3 on every molecule (Table 1). The amenable fraction is our estimate of $p$ (Section 3): $p \approx 99.4\%$ over the three datasets, so by Corollary 1 the loss floor of any MPNN is at most $0.6\%$ of the largest conditional variance of the target on a non-amenable CR class. On these benchmarks, the expressive power of a plain MPNN is not what limits performance. AQSOL is the exception, at $91.4\%$, and the per-graph test shows where the gap comes from: $94.6\%$ of its training molecules are amenable, but only $71.8\%$ of its test molecules (Table 5). The test split does not follow the distribution of the training split, and a per-graph statistic detects it before any model is trained.

**Finding 2: separating the dataset is not enough.** Zopf (2022) shows that CR fingerprints distinguish nearly all graphs in many standard datasets. On molecules the result is even stronger: CR separates every pair of molecules in each dataset, so $P_2$ holds everywhere. Yet about $21{,}000$ molecules are not amenable. Each has a confuser outside the dataset, and a new molecule from the

Table 2: Amenability of rooted $k$-hop balls (all splits). "Amen." is the fraction of balls, with the center marked, that are amenable; "Tree" the fraction that are trees, which are amenable trivially.

| | ZINC-250k | | AQSOL | | PCQM4Mv2 | |
|---|---|---|---|---|---|---|
| $k$ | Amen. | Tree | Amen. | Tree | Amen. | Tree |
| 1 | 100.00% | 99.19% | 100.00% | 99.60% | 100.00% | 98.45% |
| 2 | 100.00% | 80.66% | 99.98% | 93.21% | 99.99% | 81.16% |
| 4 | 99.96% | 16.40% | 99.71% | 38.69% | 99.95% | 29.54% |
| 8 | 99.83% | 1.30% | 98.64% | 23.38% | 99.88% | 11.27% |
| 12 | 99.73% | 0.28% | 98.21% | 21.68% | 99.87% | 9.78% |
| Balls | 69.3M | | 2.07M | | 585.1M | |
| Time | 7.0 min | | 14.1 s | | 28.6 min | |

same source can break $P_2$ (Proposition 1). A collision count reports these datasets as solved by CR. Amenability shows that confusers do occur in real molecular data, and measures how often.

**Finding 3: node-level separation needs $t_{\mathrm{CR}}$, and a random MPNN reaches it.** Separating graphs takes $k_2(\mathcal{D}) = 5$ to 7 CR iterations (Table 1, Figure 2a), and a randomly initialized MPNN gives a unique graph embedding to 87% to 99% of the molecules (random GNN separation, a graph-level measure). As argued in Section 3.1, this depth is too shallow for node-level separation, which needs $t_{\mathrm{CR}}(\mathcal{D})$ (Theorem 4): 11 to 12 iterations, about twice $k_2$, but well within reach of standard MPNNs. On ZINC-250k, a random MPNN with at most 11 layers already separates the nodes of every molecule as well as stable CR does, and most molecules need only 2 to 4 (Figure 2b, Figure 2c). The depth of an MPNN should be chosen from $t_{\mathrm{CR}}$, not from graph-level separation.

**Finding 4: local neighborhoods are amenable too.** We apply the same test to local neighborhoods. For every node $v$ and every radius $k \leq 12$, we take the $k$-hop ball $B_k(v)$, the subgraph induced by the nodes within distance $k$ of $v$, mark $v$ with an extra label, and test the result for amenability (Table 2). From $k = 4$ on, most balls contain a ring, so the test is not trivial. Yet at every radius, at least 99.7% of balls are amenable on ZINC-250k and PCQM4Mv2, and 98.2% on AQSOL: the local structure of molecules is almost never hard for CR.

This matters for local encodings. Many encodings of $v$ depend only on its marked ball: cycle counts through $v$, rooted substructure counts (Bouritsas et al., 2023), and RWPE with $\ell$ steps, which depends on $B_{\lfloor \ell/2 \rfloor + 1}(v)$. On an amenable ball, CR run on the ball identifies it up to isomorphism, so each of these encodings is a function of one CR computation on a small graph. This is consistent with the RWPE experiment below, where an MPNN regresses RWPE almost exactly.

The test scales. Checking all 656M balls of the three datasets takes 36 minutes on a 12-core laptop, 150k to 340k balls per second, and an independent implementation agrees on a random sample of 378k balls.

Table 3: Empirical evidence that RWPE adds no expressivity beyond 1-WL on ZINC-250k. Every experiment is done with 4 different random seeds and we report the mean and standard deviation.

(a) A 1-WL message-passing model can recover RWPE features almost exactly on ZINC-250k

| | Loss (MSE) | Explained Variance |
|---|---|---|
| Train | $(8 \pm 3) \times 10^{-7}$ | $99.92\% \pm 0.08\%$ |
| Test | $(4 \pm 4) \times 10^{-6}$ | $99.53\% \pm 0.07\%$ |

(b) ZINC-250k regression MAE for a 1-WL GNN with different positional encodings.

| Positional Encodings | MAE Loss |
|---|---|
| No PE | $0.087 \pm 0.008$ |
| 1-WL Approx of RWPE | $0.051 \pm 0.004$ |
| RWPE | $0.050 \pm 0.004$ |

**Finding 5: RWPE helps through inductive bias, not expressivity.** Our theoretical and empirical results so far indicate that 1-WL expressivity is sufficient for molecular graphs. Yet in practice,

vanilla MPNNs underperform models augmented with positional or structural encodings, most notably RWPE (Dwivedi et al., 2022b). The standard explanation appeals to expressivity: RWPE is provably more powerful than 1-WL, and the additional discriminative information is assumed to drive the empirical gain. This explanation cannot hold on amenable graphs: there, RWPE is a function of the CR colors, and by Corollary 3 it carries no information that CR has not already extracted. Since $99.4\%$ of ZINC-250k is amenable, we test this prediction directly.

In Table 3a we show that a standard MPNN trained to regress RWPE reaches an $R^2 > 0.99$. The information carried by RWPE is therefore already accessible to an MPNN from the raw graph, not added by it. We then use this pre-trained GNN as a 1-WL approximation of RWPE and compare it to the true RWPE. A GINE augmented with RWPE as input features outperforms a vanilla GINE, as expected. However, a GINE using the 1-WL approximation of RWPE matches the performance of the RWPE-augmented model as shown in Table 3b.

Together, these results suggest that the role of RWPE on molecular tasks is not to inject information beyond 1-WL but to provide a useful inductive bias, which is consistent with our amenability findings.

**What this means for the field.** On molecular benchmarks, a gain from a more expressive architecture is not evidence that the extra expressivity is needed: on $99.4\%$ of molecules, no encoding carries information that CR has not already extracted (Corollary 3). We suggest reporting the amenable fraction next to every graph benchmark. It takes minutes on millions of graphs, it bounds what any MPNN can lose, and it tells whether an expressivity claim can be tested on that dataset at all. Where gains remain, as with RWPE, they should be studied as inductive bias.

## 6 Limitations

We rely on RWPE as a concrete beyond-1-WL augmentation. However, our framework is agnostic to the specific mechanism used to break 1-WL: it can equally serve to diagnose *why* a given higher-order GNN (subgraph-based, $k$-WL-based, or substructure-counting) helps on a particular dataset. A thorough empirical study of any individual such architecture is a substantial effort in its own right, and we consider it out of scope for the present paper. We expect such an analysis would show that the performance gains stem less from expressivity itself than from co-occurring effects such as reduced over-smoothing and over-squashing.

## 7 Conclusion

We have introduced a dataset-side view of expressivity, complementing the model-centric 1-WL hierarchy. We have shown that CR separation, the property that collision counts measure, is not the right test of whether a dataset needs more than an MPNN. It depends on the whole dataset, it can break with one new graph, and it says nothing about nodes. Amenability fixes this. It is a property of each graph, so the amenable fraction of a training set estimates how much of the data an MPNN can get wrong and bounds its loss on unseen graphs, and it also guarantees that nodes with different roles get different embeddings.

We give a fast and practical algorithm to test amenability, which checks millions of graphs in minutes. On molecular benchmarks, more than $99\%$ of graphs are amenable, so expressivity is not what limits an MPNN on them, and more than $98\%$ of the $k$-hop neighborhoods around each atom are amenable too, so local encodings add little beyond color-refinement. Our RWPE experiments point the same way: RWPE helps only as an inductive bias, since its 1-WL approximation gives the same gain. We believe these results call for more care in how graph learning explains its progress. When a more expressive architecture performs better on an amenable dataset, the reason must lie elsewhere, and finding it is what will tell us why the architecture works.

## AI USE STATEMENT

In this work, we used generative AI tools to formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses and implement methods. We have not used generative AI tools to generate synthetic data sets, design or provide feedback on research methodology or experiments, clean and reformat datasets, support qualitative and thematic data analysis, interpret results. Additionally, we used generative AI tools to create or modify scientific figures or images, create or edit software code, brainstorming. We have reviewed AI-assisted work. We tested the AI-generated code, we verified the soundness of the theorems. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES


V. Arvind, Johannes Köbler, Gaurav Rattan, and Oleg Verbitsky. Graph isomorphism, color refinement, and compactness, 2015.

Waïss Azizian and Marc Lelarge. Expressive power of invariant and equivariant graph neural networks. In *International Conference on Learning Representations*, 2021.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yu Guang Wang, Pietro Liò, Guido Montúfar, and Michael Bronstein. Weisfeiler and Lehman go cellular: CW networks. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):657–668, January 2023. ISSN 1939-3539. doi: 10.1109/tpami.2022.3154319. URL http://dx.doi.org/10.1109/TPAMI.2022.3154319.

Zhengdao Chen, Soledad Villar, Lei Chen, and Joan Bruna. On the equivalence between graph isomorphism testing and function approximation with gnns. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Vijay Prakash Dwivedi, Chaitanya K. Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *JMLR*, 2022a.

Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. *ICLR*, 2022b.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *ICML*, 2017.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *NeurIPS*, 2020.

Md Shamim Hussain, Mohammed J. Zaki, and Dharmashankar Subramanian. Triplet interaction improves graph transformers: Accurate molecular graph learning with triplet graph transformers. *ICML*, 2024.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Sandra Kiefer, Pascal Schweitzer, and Erkal Selman. Graphs identified by logics with counting. In Giuseppe F. Italiano, Giovanni Pighizzini, and Donald T. Sannella (eds.), *Mathematical Foundations of Computer Science 2015*, volume 9234 of *Lecture Notes in Computer Science*, pp. 319–330, Berlin, Heidelberg, 2015. Springer. doi: 10.1007/978-3-662-48057-1_25.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

Derek Lim, Joshua Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. *ICLR*, 2023.

Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Can classic gnns be strong baselines for graph-level tasks? simple architectures meet excellence. *ICML*, 2025.

Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: higher-order graph neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19. AAAI Press, 2019. ISBN 978-1-57735-809-1.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *ICLR*, 2018.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Markus Zopf. 1-wl expressiveness is (almost) all you need, 2022.

# A MATHEMATICAL DETAILS

## A.1 PROOF OF REMARK 1

Take $G = C_6$, the 6-cycle, and $H = 2K_3$, the disjoint union of two triangles, both with constant node features and no edge features. Both are 2-regular on 6 vertices, so CR assigns the same color to every vertex from the first round on, and the two graphs share the stable fingerprint $\{\!\{c, c, c, c, c, c\}\!\}$.

- Each graph is vertex-transitive, so it has a single orbit and a single CR color: the equivalence $\mathrm{CR}(u) = \mathrm{CR}(v) \iff \mathrm{Orb}(u) = \mathrm{Orb}(v)$ holds trivially. Both $G$ and $H$ are refinable.
- $G$ is connected and $H$ is not, hence $G \ncong H$. Since $G \sim_{\mathrm{CR}} H$, $H$ is a confuser of $G$ (and vice versa), so neither $G$ nor $H$ is amenable.

## A.2 SHARPNESS OF THE THREE CONDITIONS

*Remark* 2 (Sharpness). Neither $P_1$ nor $P_2$ implies $P_3$, and $P_1$, $P_2$ are mutually incomparable.

1. $P_1 \not\Rightarrow P_3$**.** Take $\mathcal{D} = \{C_6\}$. The 6-cycle is vertex-transitive, so its single orbit matches its single stable CR color and $C_6$ is refinable; thus $P_1$ holds. But $C_6$ is not amenable: $2K_3$ is a confuser (Remark 1), so $P_3$ fails.
2. $P_2 \not\Rightarrow P_3$**.** Take again $\mathcal{D} = \{C_6\}$. There is no non-isomorphic pair in $\mathcal{D}$, so $P_2$ holds vacuously. As above, $P_3$ fails.
3. $P_1 \not\Rightarrow P_2$**.** Take $\mathcal{D} = \{C_6, 2K_3\}$. Both graphs are vertex-transitive, hence refinable, so $P_1$ holds. They are non-isomorphic but share the same stable CR fingerprint $\{\!\{c, c, c, c, c, c\}\!\}$ (Remark 1), so $P_2$ fails.
4. $P_2 \not\Rightarrow P_1$**.** Take $\mathcal{D} = \{C_4 \sqcup C_5\}$, the disjoint union of a 4-cycle and a 5-cycle with constant node features. $P_2$ holds vacuously. Every vertex has degree 2 and lives in a regular component, so CR assigns the same color to all 9 vertices. But $\mathrm{Aut}(C_4 \sqcup C_5) \cong D_4 \times D_5$ acts independently on the two cycles, which have different sizes and cannot be swapped, giving two orbits $V(C_4)$ and $V(C_5)$. Thus CR is strictly coarser than the orbit partition and the graph is not refinable: $P_1$ fails.

## A.3 PROOF OF PROPOSITION 1

Write $k_1(\mathcal{D})$ for the smallest $k$ such that $\mathrm{CR}_k$ separates every pair of nodes in different orbits, over all $G \in \mathcal{D}$, with $\min \emptyset = \infty$. $P_1$ holds iff $k_1(\mathcal{D}) < \infty$. We first record a lemma, also used in Theorem 4.

**Lemma 1.** *If $\mathcal{D}$ satisfies $P_3$, then $t_{\mathrm{CR}}(\mathcal{D}) = k_1(\mathcal{D})$.*

*Proof.* $P_3 \Rightarrow P_1$. For each $G \in \mathcal{D}$, amenability implies refinability (Arvind et al., 2015): the stable CR partition equals the orbit partition. The stable partition is reached by round $t_{\mathrm{CR}}(G) \le t_{\mathrm{CR}}(\mathcal{D})$, so $k_1(\mathcal{D}) \le t_{\mathrm{CR}}(\mathcal{D})$ and $P_1$ holds.

$t_{\mathrm{CR}}(\mathcal{D}) = k_1(\mathcal{D})$. The inequality $k_1 \le t_{\mathrm{CR}}$ is the previous bullet. For the converse we use two standard facts: (i) CR refinement is monotone, so $\mathrm{CR}_{k+1}$ refines $\mathrm{CR}_k$ for every $k$ (injectivity of HASH ensures cells can split but never merge); (ii) the orbit partition refines $\mathrm{CR}_t$ for every $t$. Fact (ii) follows by induction on $t$: $\mathrm{CR}_0(v) = X(v) = X(\varphi(v))$ for every automorphism $\varphi$; if $\mathrm{CR}_t \circ \varphi = \mathrm{CR}_t$, then $\mathrm{CR}_{t+1}(\varphi(v))$ has the same arguments inside HASH as $\mathrm{CR}_{t+1}(v)$ since $\varphi$ permutes $N(v)$ to $N(\varphi(v))$ and preserves edge features. Fix $G \in \mathcal{D}$ and suppose $\mathrm{CR}_k(G) = \mathrm{Orb}(G)$ at some round $k$ (refinability within $k$ rounds). Then by (i), $\mathrm{CR}_{k+1}$ refines $\mathrm{CR}_k = \mathrm{Orb}$; by (ii), Orb refines $\mathrm{CR}_{k+1}$. The two refinements together force $\mathrm{CR}_{k+1} = \mathrm{Orb} = \mathrm{CR}_k$, so CR is stable from round $k$ onwards: $t_{\mathrm{CR}}(G) \le k$. Taking $k$ to be the smallest round at which $\mathrm{CR}_k(G) = \mathrm{Orb}(G)$, then the supremum over $G \in \mathcal{D}$, gives $t_{\mathrm{CR}}(\mathcal{D}) \le k_1(\mathcal{D})$. □

*Proof of Proposition 1. Part 1.* $P_3 \Rightarrow P_1$ is the first step of the proof of Lemma 1.

$P_3 \Rightarrow P_2$. For $G, H \in \mathcal{D}$ with $G \ncong H$, amenability of $G$ gives $\mathrm{CR}(G) \neq \mathrm{CR}(H)$. Both fingerprints are stable by round $t_{\mathrm{CR}}(\mathcal{D})$, so $k_2(\mathcal{D}) \le t_{\mathrm{CR}}(\mathcal{D})$.

$k_1(\mathcal{D}) \geq k_2(\mathcal{D})$. At round $t_{\mathrm{CR}}(\mathcal{D}) = k_1(\mathcal{D})$ all fingerprints are stable, and amenability gives non-isomorphic graphs distinct stable fingerprints; hence $k_2 \leq k_1$.

*Part 2.* Every graph in the support of $\mu$ is amenable, so every graph of $\mathcal{D}_{\mathrm{train}} \cup \mathcal{D}_{\mathrm{test}}$ is amenable. The union satisfies $P_3$ by definition, and Part 1 applies to it. For the failure of $P_2$, let $H$ be a confuser of $G$ (Section 2.1), for instance $G = C_6$ and $H = 2K_3$ (Remark 1), and let $\mu$ be uniform on $\{G, H\}$. Sample one training graph and one test graph. $\mathcal{D}_{\mathrm{train}}$ is a singleton and satisfies $P_2$ trivially. With probability $1/2$ the test graph is the other member of the pair, and the union violates $P_2$. □

### A.4 Universal approximation on amenable data

**Theorem 4** (Universal approximation on amenable data)**.** *Let $\mathcal{D} \subseteq \mathcal{G}_n$ be amenable and write $t = t_{\mathrm{CR}}(\mathcal{D})$.*

1. Graph level. *For every continuous permutation-invariant $f : \mathcal{D} \to \mathbb{R}$ and every $\varepsilon > 0$, a GIN with $t$ message-passing layers (injective sum aggregation, sum readout) and a 3-layer MLP readout $M$ achieves $\sup_{G \in \mathcal{D}} |M(G) - f(G)| < \varepsilon$. A 2-layer readout suffices when $\mathcal{X}$ and $\mathcal{X}_E$ are finite.*
2. Node level, and exactly $t$. *For every $G \in \mathcal{D}$ and every $\mathrm{Aut}(G)$-invariant $f : V(G) \to \mathbb{R}$, a GIN with $t$ layers and a node-level 2-layer readout reproduces $f$ exactly. No GIN with fewer than $t$ layers does so on all of $\mathcal{D}$.*

We rely on the equivalence between graph isomorphism testing and function approximation established by Chen et al. (2019).

**Definition 3** (GIso-discriminating, augmented class, Chen et al., 2019)**.** A class $\mathcal{C}$ of permutation-invariant functions on graphs is *GIso-discriminating* on a set $\mathcal{S}$ if for all non-isomorphic $G, H \in \mathcal{S}$ there exists $h \in \mathcal{C}$ with $h(G) \neq h(H)$. The *augmented class* $\mathcal{C}^{+L}$ consists of all functions $G \mapsto \mathrm{NN}(h_1(G), \ldots, h_d(G))$ with $h_i \in \mathcal{C}$ and NN an $L$-layer feedforward network.

**Theorem 5** (Chen et al. 2019)**.** *Let $\mathcal{C}$ be a class of continuous permutation-invariant functions on graphs with node features in a compact $\mathcal{X}$.*

1. *If $\mathcal{C}$ is universally approximating, it is GIso-discriminating.*
2. *If $\mathcal{C}$ is GIso-discriminating, then $\mathcal{C}^{+3}$ is universally approximating; if $\mathcal{X}$ is finite, $\mathcal{C}^{+2}$ suffices.*

The fixed-dataset statement comes first.

**Theorem 6** (Universal approximation under $P_2$)**.** *Let $\mathcal{D} \subseteq \mathcal{G}_n$ satisfy $P_2$, fix any $k \geq k_2(\mathcal{D})$, and let $f : \mathcal{D} \to \mathbb{R}$ be continuous and permutation-invariant. For any $\varepsilon > 0$ there exists a GIN with $k$ message-passing layers (injective sum aggregation, sum readout) and a 3-layer MLP readout $M$ such that $\sup_{G \in \mathcal{D}} |M(G) - f(G)| < \varepsilon$. A 2-layer readout suffices when $\mathcal{X}$ and $\mathcal{X}_E$ are finite.*

*Proof.* Let $\mathcal{C}_k$ be the class of scalar functions computable by a GIN with $k$ message-passing layers, injective aggregation, and sum readout. Each $h \in \mathcal{C}_k$ is continuous and permutation-invariant, and computes a coloring at least as fine as $\mathrm{CR}_k$ (Theorem 1). Since $k \geq k_2(\mathcal{D})$, $\mathrm{CR}_k$ separates non-isomorphic pairs in $\mathcal{D}$, hence $\mathcal{C}_k$ is GIso-discriminating on $\mathcal{D}$. By Theorem 5, $\mathcal{C}_k^{+3}$ is universally approximating on $\mathcal{D}$ ($\mathcal{C}_k^{+2}$ in the finite-feature case). Finally, since GIN aggregation and update act coordinate-wise on hidden states, $d$ scalar GINs with shared combinatorial structure can be packaged as a single GIN with $d$-dimensional hidden states and the same $k$ message-passing layers, and the post-hoc NN becomes the MLP readout. □

**Part 1.** Let $\mathcal{D} \subseteq \mathcal{G}_n$ be amenable and $t = t_{\mathrm{CR}}(\mathcal{D})$. By Proposition 1, $\mathcal{D}$ satisfies $P_2$ with $k_2(\mathcal{D}) \leq t \leq n$, so Theorem 6 applies with $k = t$. Taking $k = n$ instead gives a depth fixed in advance of $\mathcal{D}$, so the same architecture covers every amenable subset of $\mathcal{G}_n$ at once.

**Part 2, sufficiency.** Fix $G \in \mathcal{D}$. Amenability implies refinability, and $t \geq t_{\mathrm{CR}}(G)$ by definition of the supremum. By Theorem 1, a GIN with $t$ message-passing layers and injective sum aggregation produces node embeddings $h_v^{(t)}$ with

$$h_u^{(t)} = h_v^{(t)} \iff \mathrm{CR}_t(u) = \mathrm{CR}_t(v),$$

and the induced map on $\mathrm{CR}_t$-classes is injective. Color refinement has stabilized at depth $t$ on $G$, so $\mathrm{CR}_t(u) = \mathrm{CR}_t(v) \iff \mathrm{CR}(u) = \mathrm{CR}(v)$, and refinability gives

$$h_u^{(t)} = h_v^{(t)} \iff \mathrm{Orb}(u) = \mathrm{Orb}(v).$$

The embedding therefore factors through the orbit partition: there is an injection $\bar{h} : V(G)/\mathrm{Aut}(G) \to \mathbb{R}^d$ with $h_v^{(t)} = \bar{h}(\mathrm{Orb}(v))$ for every $v$.

Since $f$ is $\mathrm{Aut}(G)$-invariant, it factors as $f = \tilde{f} \circ \mathrm{Orb}$ for a unique $\tilde{f} : V(G)/\mathrm{Aut}(G) \to \mathbb{R}$. Define $g$ on the finite set $\bar{h}(V(G)/\mathrm{Aut}(G)) \subset \mathbb{R}^d$ by $g(\bar{h}(o)) = \tilde{f}(o)$; this is well defined because $\bar{h}$ is injective. A 2-layer MLP with ReLU activations interpolates any real-valued function on a finite point set exactly, so there exists $M$ with $M(\bar{h}(o)) = \tilde{f}(o)$ for every orbit $o$. Composing GIN with $M$ node-wise,

$$M(G)_v \;=\; M(h_v^{(t)}) \;=\; M(\bar{h}(\mathrm{Orb}(v))) \;=\; \tilde{f}(\mathrm{Orb}(v)) \;=\; f(v).$$

**Part 2, necessity.** Let $k < t$. By Lemma 1, $t = k_1(\mathcal{D})$, so by minimality of $k_1$ there are a graph $G \in \mathcal{D}$ and nodes $u, v \in V(G)$ with $\mathrm{CR}_k(u) = \mathrm{CR}_k(v)$ and $\mathrm{Orb}(u) \neq \mathrm{Orb}(v)$. Take $f$ to be the indicator of $\mathrm{Orb}(u)$, which is $\mathrm{Aut}(G)$-invariant and satisfies $f(u) = 1 \neq 0 = f(v)$. Any MPNN with $k$ message-passing layers assigns $u$ and $v$ the same embedding by Theorem 1, so its node-level output cannot separate them and cannot reproduce $f$. □

*Remark* 3. Part 2 extends to any $\mathrm{Aut}(G)$-invariant target $f : V(G)^m \to \mathbb{R}$ on $m$-tuples of nodes, by composing the node embeddings into tuple embeddings before the MLP readout: under refinability, two tuples $(u_1, \ldots, u_m)$ and $(v_1, \ldots, v_m)$ receive identical tuple embeddings iff they lie in the same $\mathrm{Aut}(G)$-orbit on $V(G)^m$, and the argument above applies verbatim. Taking $m = 2$ recovers a universal approximator for orbit-invariant edge-level targets on $G$.

### A.5 Risk decomposition

*Proof of Theorem 2.* Let $\pi$ denote the pushforward of $\mu$ under $G \mapsto [G]_{\mathrm{CR}}$ and let $\mu_{[G]}$ be the conditional of $\mu$ on the class $[G]$. For any $f \in \mathcal{F}_{\mathrm{MPNN}}$, $f(G)$ depends only on $[G]_{\mathrm{CR}}$; denote this common value by $\tilde{f}([G])$. By the tower property,

$$\mathcal{L}(f) \;=\; \mathbb{E}_{[G]\sim\pi}\Big[\, \mathbb{E}_{H\sim\mu_{[G]}}\, (\tilde{f}([G]) - y(H))^2 \,\Big].$$

The inner expectation is minimized in $\tilde{f}([G])$ by the conditional mean $\mathbb{E}_{H\sim\mu_{[G]}}[y(H)]$, with residual value $\mathrm{Var}_{H\sim\mu_{[G]}}(y(H))$. Taking the outer expectation gives the claim. □

*Proof of Corollary 2.* Every $f \in \mathcal{F}_{\mathrm{MPNN}}$ is constant on each CR class, so $f(H) = f(G)$ for every $H \sim_{\mathrm{CR}} G$, and the average of $f$ over the class of $G$ is $f(G)$. For a parametrized MPNN $f_\theta$, this holds for every $\theta$: by Theorem 1, $f_\theta(H) = f_\theta(G)$ for all $\theta$ whenever $H \sim_{\mathrm{CR}} G$. Two functions of $\theta$ that are equal everywhere have equal gradients, so $\nabla_\theta f_\theta(H) = \nabla_\theta f_\theta(G)$, and averaging over the class gives the second identity. □

### A.6 CR is complete on amenable graphs

**Theorem 7** (CR is complete on amenable graphs)**.** *Let $\mathcal{D}$ be a set of amenable graphs. Then $G \mapsto \mathrm{CR}(G)$ is injective on $\mathcal{D}/\cong$. Amenability is the regime in which WL-1 is as informative as the full isomorphism: anything a permutation-invariant target depends on is already visible through CR.*

*Proof.* Let $G_1, G_2 \in \mathcal{D}$ with $\mathrm{CR}(G_1) = \mathrm{CR}(G_2)$. By definition of $\sim_{\mathrm{CR}}$, this means $G_2 \in [G_1]_{\mathrm{CR}}$. Since $G_1$ is amenable, its CR class consists of a single isomorphism class, hence $G_2 \cong G_1$. Conversely, isomorphic graphs always share the same CR colouring, so $G \mapsto \mathrm{CR}(G)$ descends to a well-defined injection on $\mathcal{D}/\cong$. □

**Corollary 3** (Information content)**.** *For any permutation-invariant target $y$ and any amenable $G$,*

$$I(y(G); \mathrm{CR}(G)) = I(y(G); [G]_{\cong}) = H(y(G)).$$

*No additional positional, or structural encoding can increase the information content of the representation beyond what CR already provides on amenable graphs.*

*Proof.* By Theorem 7, on amenable graphs the map $\mathrm{CR}(G) \mapsto [G]_{\cong}$ is a bijection, so the $\sigma$-algebras generated by $\mathrm{CR}(G)$ and by $[G]_{\cong}$ coincide. Since $y$ is permutation-invariant, $y(G)$ is $\sigma([G]_{\cong})$-measurable, and therefore $H(y(G) \mid \mathrm{CR}(G)) = H(y(G) \mid [G]_{\cong}) = 0$. Hence

$$I(y(G); \mathrm{CR}(G)) \;=\; H(y(G)) - H(y(G) \mid \mathrm{CR}(G)) \;=\; H(y(G)),$$

and identically for $[G]_{\cong}$. □

# B THE AMENABILITY TEST

## B.1 STEPS OF THE AMENABILITY TEST

Theorem 3 yields a direct four-step procedure on a graph with $n$ vertices and $m$ edges.

- **Color refinement:** $O((n+m)\log n)$ - Run CR to obtain the stable partition $P_G$.
- **Cell graph :** $O(n+m)$ - Build $\mathcal{C}(G)$, labeling each cell as homogeneous or heterogeneous and each pair as isotropic or anisotropic.
- **Local checks:** $O(n+m)$ - Match each cell and each anisotropic pair against the allowed shape catalogues.
- **Global checks:** $O(n+m)$- BFS each anisotropic component to verify it is a tree, then check monotonicity and the at-most-one-heterogeneous-cell condition from a minimum-cardinality root.

Steps 2 to 4 visit each edge a constant number of times, so they cost $O(n+m)$; the dominant term is the $\log n$ factor from CR, giving an overall $O((n+m)\log n)$ runtime (Arvind et al., 2015).

## B.2 IMPLEMENTATION

We release the amenability algorithm used in our experiments as an open-source Python package. The package additionally provides utilities to:

- extract colorings from attributed graphs;
- run the color refinement algorithm;
- compute exact colorings via the `pynauty` package.

# C EXPERIMENTAL SETUP

## C.1 DATASETS

**ZINC-250k.** ZINC-250k is a curated subset of approximately 250,000 drug-like molecules drawn from the ZINC database of commercially available compounds. Nodes correspond to heavy atoms and edges encode covalent bonds with single, double, triple, or aromatic types. Molecules contain up to roughly 38 heavy atoms. We use $249{,}456$ molecules.

**AQSOL.** AQSOL is a real-world molecular regression benchmark introduced in the updated Dwivedi et al. (2022a), derived from AqSolDB, a curated database of experimentally measured aqueous solubilities. Nodes carry a categorical heavy-atom type, and edges carry a categorical bond type (single, double, triple, aromatic). After filtering out graphs with no edges or missing node features, the benchmark retains $9{,}823$ molecular graphs. A scaffold split is provided in an $8{:}1{:}1$ ratio, yielding $7{,}831$ training, $996$ validation, and $996$ test graphs. Scaffold splitting tests out-of-distribution generalization to structurally novel molecules.

**PCQM4Mv2.** PCQM4Mv2 is the graph-regression benchmark of the OGB Large-Scale Challenge (OGB-LSC) (Hu et al., 2020), derived from the PubChemQC project. Each molecule is labelled with a HOMO–LUMO energy gap computed by density functional theory. Molecules are built from SMILES strings using the standard OGB featurization: 9-dimensional categorical node features (atomic number, chirality, degree, formal charge, etc.) and 3-dimensional edge features (bond type, stereo, conjugation). We use the $3{,}452{,}118$ molecules of the training and validation splits.

### C.2 RWPE EXPERIMENTS

We adopt a standard GINE architecture for the ZINC-250k task. This choice is motivated by Luo et al. (2025), who show that such architectures can reach state-of-the-art performance on ZINC-250k with only minor modifications, namely feed-forward layers, residual connections, and normalization layers. Our final model is a GINE architecture with:

- 12 layers;
- edge features incorporated;
- 80 hidden dimensions;
- 160 feed-forward dimensions;
- sum aggregation;
- 475k parameters in total.

We train this model for 500,000 steps with a batch size of 32 and a learning rate of $10^{-3}$. Each run takes approximately 12 hours on a single GPU. We report the mean and standard deviation over 4 seeds.

For approximating RWPE with MPNNs, we again use a GIN-based architecture; this time, however, the model receives only the graph structure (no node or edge features) as input, and the target for each node is its true RWPE vector. The configuration is:

- jumping knowledge for the prediction head;
- 256 hidden dimensions;
- 512 feed-forward dimensions;
- 24 layers;
- 11M parameters.

We train with a learning rate of $10^{-5}$, a batch size of 256, and for 250,000 training steps. Training takes approximately 12 hours on a single GPU.

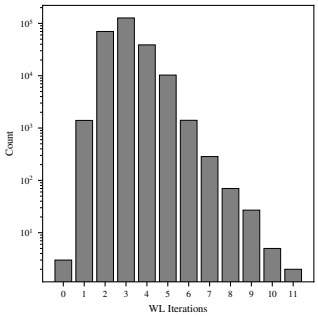
## D ADDITIONAL RESULTS

### D.1 ZINC-250K

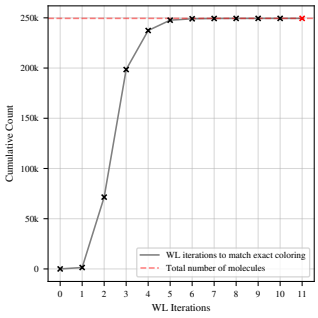
**Dataset-wide properties.** In Table 4 we report the separation and amenability ratio of each split, and in Figure 3 the separation behavior of WL-1 and a random GNN.

Table 4: Separation and amenability statistics on ZINC-250k.

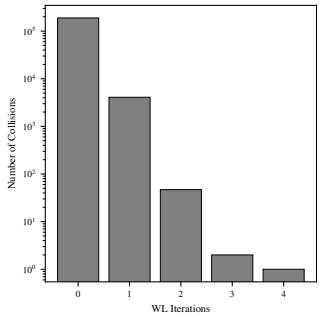
| Split | Random GNN Separation | CR Separation | Amenability |
|---|---|---|---|
| Train | $99.37\%$ | $100.00\%$ | $99.37\%$ |
| Validation | $99.91\%$ | $100.00\%$ | $99.42\%$ |
| Test | $99.98\%$ | $100.00\%$ | $99.48\%$ |



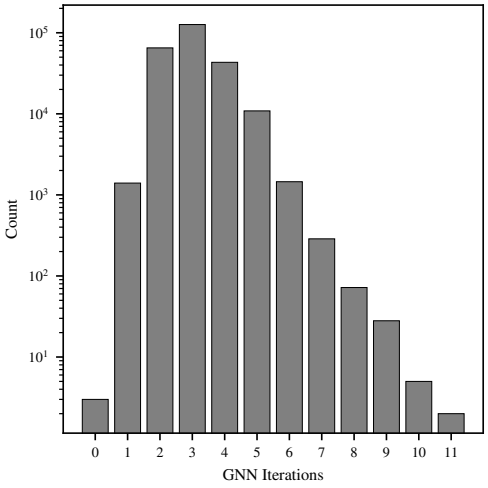

(a) Distribution of the number of WL-1 iterations required to reach the exact graph coloring (log-scale count).



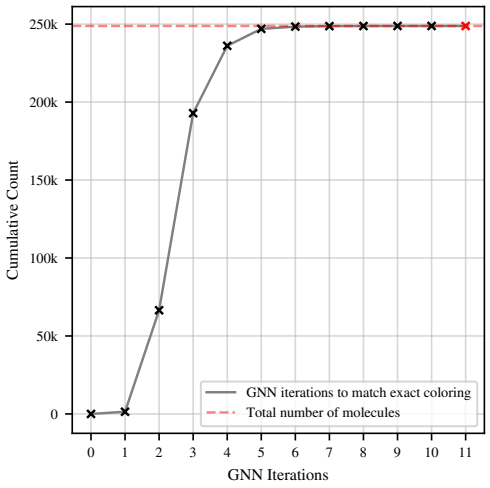

(b) Cumulative number of molecules separated by WL-1 as a function of the iteration count; the dashed line marks the dataset size.




(c) Number of WL-1 fingerprint collisions between molecules as a function of the WL-1 iteration count, over the dataset (log-scale).




(d) Distribution of the number of iterations of a random GNN required to separate every node (log-scale count).




(e) Cumulative number of molecules whose nodes are all separated by the random GNN, as a function of the number of layers; the dashed line marks the dataset size.

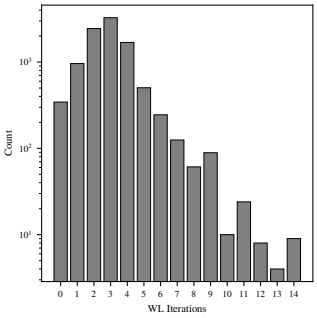
Figure 3: Separation behavior of WL-1 and a random GNN on ZINC-250K. Top row: WL-1 reaches an exact coloring of every molecule within a small number of iterations (a, b), and the residual color collisions decay rapidly with the iteration count (c). Bottom row: at the node level, a random GNN converges in a similar way (d, e). At the graph level it separates fewer molecules than WL-1 (Table 4).

## D.2 AQSOL

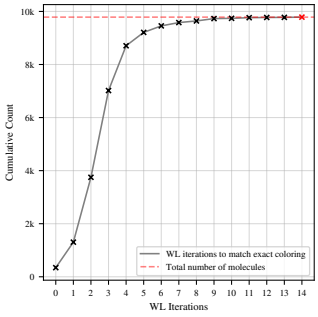
**Dataset-wide properties.** In Table 5 we report the separation and amenability ratio of each split, and in Figure 4 the separation behavior of WL-1 and a random GNN.

Table 5: Separation and amenability statistics on AQSOL.

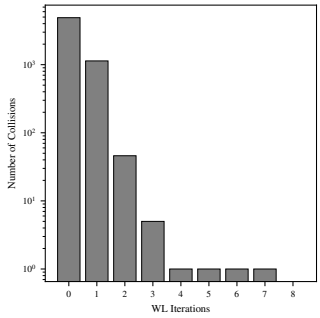
| Split | Random GNN Separation | CR Separation | Amenability |
|---|---|---|---|
| Train | 91.64% | 100.00% | 94.60% |
| Validation | 92.48% | 100.00% | 85.77% |
| Test | 97.90% | 100.00% | 71.77% |



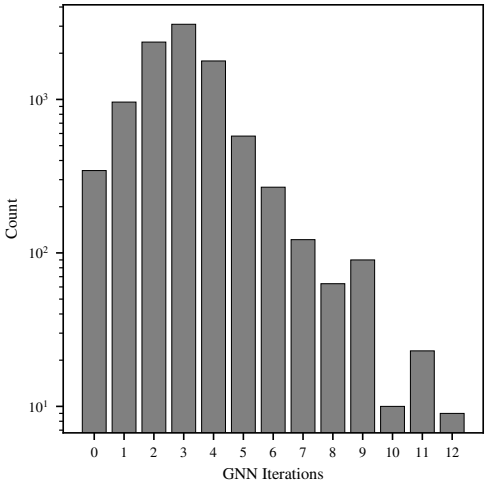

(a) Distribution of the number of WL-1 iterations required to reach the exact graph coloring (log-scale count).



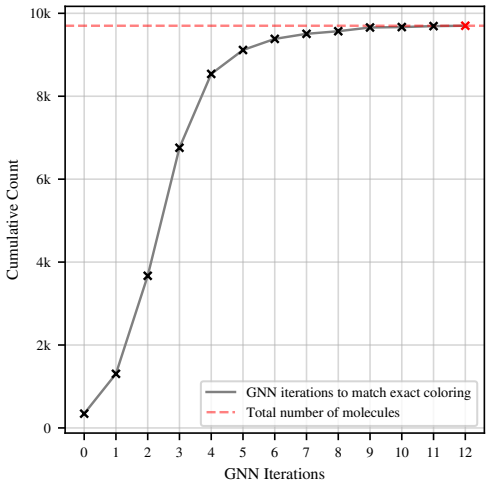

(b) Cumulative number of molecules separated by WL-1 as a function of the iteration count; the dashed line marks the dataset size.




(c) Number of WL-1 fingerprint collisions between molecules as a function of the WL-1 iteration count, over the dataset (log-scale).




(d) Distribution of the number of iterations of a random GNN required to separate every node (log-scale count).




(e) Cumulative number of molecules whose nodes are all separated by the random GNN, as a function of the number of layers; the dashed line marks the dataset size.

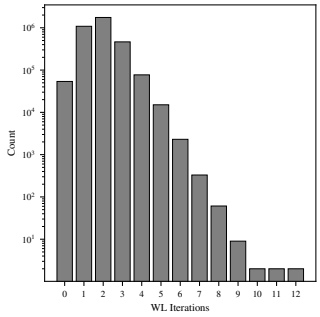
Figure 4: Separation behavior of WL-1 and a random GNN on AQSOL. Top row: WL-1 reaches an exact coloring of every molecule within a small number of iterations (a, b), and the residual color collisions decay rapidly with the iteration count (c). Bottom row: at the node level, a random GNN converges in a similar way (d, e). At the graph level it separates fewer molecules than WL-1 (Table 5).

### D.3 PCQM4Mv2

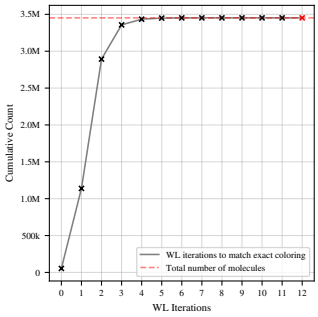
**Dataset-wide properties.** In Table 6 we report the separation and amenability ratio of each split, and in Figure 5 the separation behavior of WL-1 and a random GNN.

Table 6: Separation and amenability statistics on PCQM4Mv2.

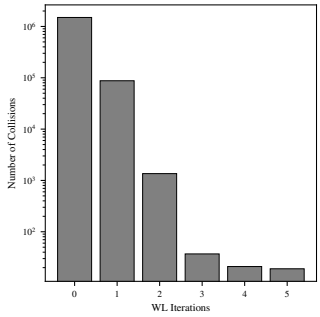
| Split | Random GNN Separation | CR Separation | Amenability |
|---|---|---|---|
| Train | 87.03% | 100.00% | 99.51% |
| Validation | 91.42% | 100.00% | 97.72% |



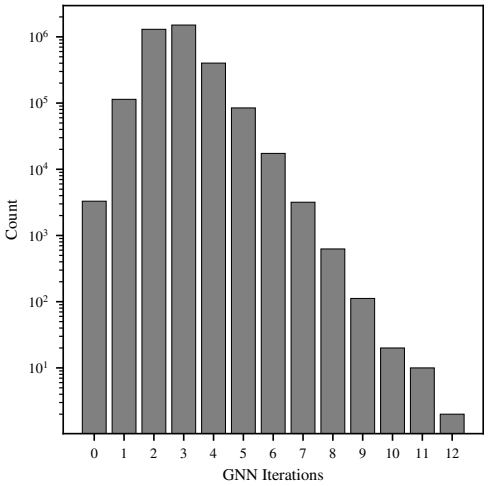

(a) Distribution of the number of WL-1 iterations required to reach the exact graph coloring (log-scale count).



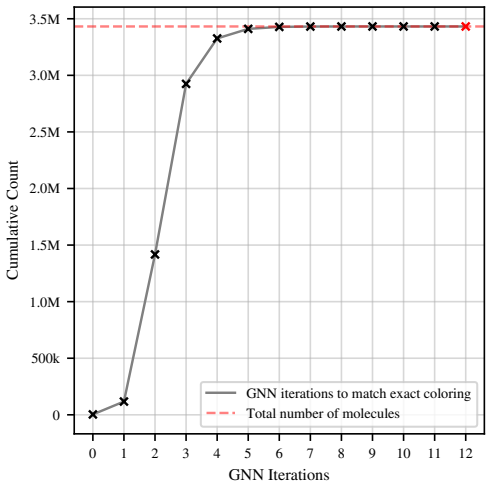

(b) Cumulative number of molecules separated by WL-1 as a function of the iteration count; the dashed line marks the dataset size.




(c) Number of WL-1 fingerprint collisions between molecules as a function of the WL-1 iteration count, over the dataset (log-scale).




(d) Distribution of the number of iterations of a random GNN required to separate every node (log-scale count).




(e) Cumulative number of molecules whose nodes are all separated by the random GNN, as a function of the number of layers; the dashed line marks the dataset size.

Figure 5: Separation behavior of WL-1 and a random GNN on PCQM4Mv2. Top row: WL-1 reaches an exact coloring of every molecule within a small number of iterations (a, b), and the residual color collisions decay rapidly with the iteration count (c). Bottom row: at the node level, a random GNN converges in a similar way (d, e). At the graph level it separates fewer molecules than WL-1 (Table 6).

### D.4 AMENABILITY OF LOCAL NEIGHBORHOODS

For every node $v$ and every radius $k = 1, \ldots, 12$, we test the $k$-hop ball $B_k(v)$ for amenability, with node and edge features and with $v$ marked by an extra label (Table 2). "Balls" counts one ball per node and radius. "Computed" counts the balls actually tested: a ball that already covers its whole molecule at a smaller radius is not tested again. "Molecules fully amenable" is the fraction of molecules whose balls of radius $k$ are all amenable. As a check, we re-tested a random $0.1\%$ of the balls (377,707) with an independent implementation, and found no mismatch. Table 7 gives the running time, and Tables 8, 12 and 16 the results for each dataset and split.

Table 7: Time to test every rooted $k$-hop ball ($k = 1, \ldots, 12$) for amenability, on a laptop with 12 CPUs and 12 workers. "Centers" is the number of nodes; each is the center of one ball per radius.

| Dataset | Split | Molecules | Centers | Balls | Wall time | CPU time | Balls/s |
|---|---|---|---|---|---|---|---|
| AQSOL | train | 7,836 | 120,833 | 1,449,996 | 6.8 s | 1.1 min | 214,021 |
| | validation | 998 | 23,443 | 281,316 | 3.4 s | 21.0 s | 81,798 |
| | test | 999 | 28,635 | 343,620 | 3.9 s | 30.8 s | 88,824 |
| | *total* | 9,833 | 172,911 | 2,074,932 | 14.1 s | 2.0 min | 147,339 |
| ZINC-250k | train | 220,011 | 5,094,322 | 61,131,864 | 6.2 min | 1.14 h | 163,530 |
| | validation | 24,445 | 565,429 | 6,785,148 | 40.1 s | 7.7 min | 169,370 |
| | test | 5,000 | 115,506 | 1,386,072 | 8.7 s | 1.6 min | 158,657 |
| | *total* | 249,456 | 5,775,257 | 69,303,084 | 7.0 min | 1.30 h | 163,983 |
| PCQM4Mv2 | train | 3,378,573 | 47,711,301 | 572,535,612 | 27.8 min | 4.95 h | 343,482 |
| | validation | 73,545 | 1,050,647 | 12,607,764 | 46.5 s | 7.2 min | 271,125 |
| | *total* | 3,452,118 | 48,761,948 | 585,143,376 | 28.6 min | 5.07 h | 341,518 |

Table 8: Rooted $k$-hop ball amenability on AQSOL, all splits.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 172,911 | 172,911 | 100.000% | 99.60% | 100.00% |
| 2 | 172,911 | 170,769 | 99.980% | 93.21% | 99.67% |
| 3 | 172,911 | 167,944 | 99.962% | 52.18% | 99.69% |
| 4 | 172,911 | 161,024 | 99.709% | 38.69% | 96.04% |
| 5 | 172,911 | 147,721 | 99.464% | 30.28% | 95.89% |
| 6 | 172,911 | 130,165 | 99.134% | 26.83% | 95.78% |
| 7 | 172,911 | 109,923 | 98.855% | 24.71% | 95.79% |
| 8 | 172,911 | 90,653 | 98.636% | 23.38% | 95.74% |
| 9 | 172,911 | 73,086 | 98.494% | 22.59% | 95.75% |
| 10 | 172,911 | 58,575 | 98.366% | 22.11% | 95.74% |
| 11 | 172,911 | 47,504 | 98.282% | 21.84% | 95.77% |
| 12 | 172,911 | 39,075 | 98.208% | 21.68% | 95.78% |

Table 9: Rooted $k$-hop ball amenability on AQSOL, train split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 120,833 | 120,833 | 100.000% | 99.73% | 100.00% |
| 2 | 120,833 | 119,205 | 99.993% | 95.32% | 99.89% |
| 3 | 120,833 | 116,524 | 99.988% | 59.99% | 99.94% |
| 4 | 120,833 | 110,530 | 99.780% | 47.42% | 96.94% |
| 5 | 120,833 | 99,294 | 99.611% | 39.44% | 97.10% |
| 6 | 120,833 | 84,852 | 99.388% | 35.77% | 97.24% |
| 7 | 120,833 | 68,965 | 99.231% | 33.41% | 97.31% |
| 8 | 120,833 | 54,448 | 99.090% | 32.00% | 97.31% |
| 9 | 120,833 | 41,878 | 98.988% | 31.13% | 97.31% |
| 10 | 120,833 | 32,056 | 98.905% | 30.60% | 97.31% |
| 11 | 120,833 | 24,950 | 98.870% | 30.27% | 97.33% |
| 12 | 120,833 | 19,830 | 98.849% | 30.07% | 97.35% |

Table 10: Rooted $k$-hop ball amenability on AQSOL, validation split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 23,443 | 23,443 | 100.000% | 99.42% | 100.00% |
| 2 | 23,443 | 23,346 | 99.953% | 86.71% | 98.90% |
| 3 | 23,443 | 23,322 | 99.957% | 32.10% | 99.10% |
| 4 | 23,443 | 22,983 | 99.642% | 16.44% | 93.99% |
| 5 | 23,443 | 22,156 | 99.262% | 8.17% | 92.89% |
| 6 | 23,443 | 20,698 | 98.801% | 5.25% | 92.18% |
| 7 | 23,443 | 18,466 | 98.319% | 3.57% | 92.18% |
| 8 | 23,443 | 15,889 | 98.042% | 2.37% | 92.08% |
| 9 | 23,443 | 13,179 | 97.820% | 1.74% | 92.18% |
| 10 | 23,443 | 10,573 | 97.603% | 1.21% | 92.28% |
| 11 | 23,443 | 8,437 | 97.449% | 1.07% | 92.28% |
| 12 | 23,443 | 6,744 | 97.270% | 1.00% | 92.28% |

Table 11: Rooted $k$-hop ball amenability on AQSOL, test split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 28,635 | 28,635 | 100.000% | 99.20% | 100.00% |
| 2 | 28,635 | 28,218 | 99.951% | 89.64% | 98.80% |
| 3 | 28,635 | 28,098 | 99.860% | 35.63% | 98.40% |
| 4 | 28,635 | 27,511 | 99.466% | 20.07% | 91.09% |
| 5 | 28,635 | 26,271 | 99.012% | 9.73% | 89.39% |
| 6 | 28,635 | 24,615 | 98.338% | 6.78% | 87.89% |
| 7 | 28,635 | 22,492 | 97.706% | 5.32% | 87.49% |
| 8 | 28,635 | 20,316 | 97.210% | 4.18% | 87.09% |
| 9 | 28,635 | 18,029 | 96.962% | 3.62% | 87.09% |
| 10 | 28,635 | 15,946 | 96.714% | 3.41% | 86.89% |
| 11 | 28,635 | 14,117 | 96.483% | 3.28% | 86.99% |
| 12 | 28,635 | 12,501 | 96.270% | 3.19% | 86.99% |

Table 12: Rooted $k$-hop ball amenability on ZINC-250k, all splits.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 5,775,257 | 5,775,257 | 100.000% | 99.19% | 100.00% |
| 2 | 5,775,257 | 5,775,257 | 99.996% | 80.66% | 99.90% |
| 3 | 5,775,257 | 5,775,214 | 99.961% | 31.07% | 99.59% |
| 4 | 5,775,257 | 5,772,401 | 99.957% | 16.40% | 99.32% |
| 5 | 5,775,257 | 5,745,258 | 99.933% | 7.51% | 99.37% |
| 6 | 5,775,257 | 5,631,156 | 99.900% | 3.89% | 99.40% |
| 7 | 5,775,257 | 5,334,919 | 99.864% | 2.19% | 99.44% |
| 8 | 5,775,257 | 4,790,801 | 99.832% | 1.30% | 99.44% |
| 9 | 5,775,257 | 4,042,689 | 99.800% | 0.77% | 99.45% |
| 10 | 5,775,257 | 3,210,756 | 99.770% | 0.47% | 99.45% |
| 11 | 5,775,257 | 2,403,735 | 99.747% | 0.33% | 99.46% |
| 12 | 5,775,257 | 1,684,047 | 99.731% | 0.28% | 99.47% |

Table 13: Rooted $k$-hop ball amenability on ZINC-250k, train split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 5,094,322 | 5,094,322 | 100.000% | 99.19% | 100.00% |
| 2 | 5,094,322 | 5,094,322 | 99.996% | 80.68% | 99.90% |
| 3 | 5,094,322 | 5,094,285 | 99.960% | 31.07% | 99.59% |
| 4 | 5,094,322 | 5,091,828 | 99.956% | 16.39% | 99.31% |
| 5 | 5,094,322 | 5,068,006 | 99.931% | 7.50% | 99.36% |
| 6 | 5,094,322 | 4,967,644 | 99.897% | 3.88% | 99.39% |
| 7 | 5,094,322 | 4,706,600 | 99.861% | 2.18% | 99.43% |
| 8 | 5,094,322 | 4,226,826 | 99.828% | 1.29% | 99.44% |
| 9 | 5,094,322 | 3,566,913 | 99.796% | 0.76% | 99.44% |
| 10 | 5,094,322 | 2,832,882 | 99.766% | 0.46% | 99.44% |
| 11 | 5,094,322 | 2,120,905 | 99.742% | 0.32% | 99.46% |
| 12 | 5,094,322 | 1,486,106 | 99.726% | 0.27% | 99.46% |

Table 14: Rooted $k$-hop ball amenability on ZINC-250k, validation split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 565,429 | 565,429 | 100.000% | 99.17% | 100.00% |
| 2 | 565,429 | 565,429 | 99.996% | 80.44% | 99.91% |
| 3 | 565,429 | 565,424 | 99.966% | 31.00% | 99.66% |
| 4 | 565,429 | 565,123 | 99.962% | 16.38% | 99.35% |
| 5 | 565,429 | 562,395 | 99.946% | 7.55% | 99.46% |
| 6 | 565,429 | 551,061 | 99.918% | 3.95% | 99.47% |
| 7 | 565,429 | 521,968 | 99.889% | 2.26% | 99.51% |
| 8 | 565,429 | 468,660 | 99.860% | 1.35% | 99.51% |
| 9 | 565,429 | 395,506 | 99.830% | 0.82% | 99.51% |
| 10 | 565,429 | 314,203 | 99.800% | 0.52% | 99.52% |
| 11 | 565,429 | 235,391 | 99.776% | 0.38% | 99.52% |
| 12 | 565,429 | 164,962 | 99.760% | 0.33% | 99.53% |

Table 15: Rooted $k$-hop ball amenability on ZINC-250k, test split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 115,506 | 115,506 | 100.000% | 99.24% | 100.00% |
| 2 | 115,506 | 115,506 | 99.996% | 80.78% | 99.90% |
| 3 | 115,506 | 115,505 | 99.951% | 31.44% | 99.48% |
| 4 | 115,506 | 115,450 | 99.971% | 16.56% | 99.48% |
| 5 | 115,506 | 114,857 | 99.953% | 7.70% | 99.50% |
| 6 | 115,506 | 112,451 | 99.920% | 4.05% | 99.50% |
| 7 | 115,506 | 106,351 | 99.887% | 2.28% | 99.50% |
| 8 | 115,506 | 95,315 | 99.861% | 1.41% | 99.50% |
| 9 | 115,506 | 80,270 | 99.838% | 0.87% | 99.50% |
| 10 | 115,506 | 63,671 | 99.823% | 0.58% | 99.54% |
| 11 | 115,506 | 47,439 | 99.815% | 0.43% | 99.54% |
| 12 | 115,506 | 32,979 | 99.813% | 0.39% | 99.58% |

Table 16: Rooted $k$-hop ball amenability on PCQM4Mv2, all splits.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 48,761,948 | 48,761,948 | 100.000% | 98.45% | 100.00% |
| 2 | 48,761,948 | 48,758,176 | 99.994% | 81.16% | 99.91% |
| 3 | 48,761,948 | 48,667,486 | 99.979% | 42.49% | 99.81% |
| 4 | 48,761,948 | 47,460,858 | 99.952% | 29.54% | 99.64% |
| 5 | 48,761,948 | 42,600,846 | 99.925% | 21.31% | 99.63% |
| 6 | 48,761,948 | 33,517,795 | 99.900% | 16.44% | 99.63% |
| 7 | 48,761,948 | 22,940,457 | 99.884% | 13.16% | 99.63% |
| 8 | 48,761,948 | 13,516,410 | 99.875% | 11.27% | 99.63% |
| 9 | 48,761,948 | 6,688,614 | 99.871% | 10.33% | 99.63% |
| 10 | 48,761,948 | 2,717,863 | 99.870% | 9.94% | 99.63% |
| 11 | 48,761,948 | 902,290 | 99.869% | 9.81% | 99.63% |
| 12 | 48,761,948 | 259,076 | 99.869% | 9.78% | 99.63% |

Table 17: Rooted $k$-hop ball amenability on PCQM4Mv2, train split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 47,711,301 | 47,711,301 | 100.000% | 98.44% | 100.00% |
| 2 | 47,711,301 | 47,707,882 | 99.994% | 81.09% | 99.91% |
| 3 | 47,711,301 | 47,621,192 | 99.979% | 42.52% | 99.81% |
| 4 | 47,711,301 | 46,449,908 | 99.954% | 29.56% | 99.66% |
| 5 | 47,711,301 | 41,704,991 | 99.929% | 21.32% | 99.65% |
| 6 | 47,711,301 | 32,811,259 | 99.906% | 16.43% | 99.65% |
| 7 | 47,711,301 | 22,443,867 | 99.890% | 13.14% | 99.65% |
| 8 | 47,711,301 | 13,202,061 | 99.883% | 11.24% | 99.65% |
| 9 | 47,711,301 | 6,508,768 | 99.880% | 10.30% | 99.65% |
| 10 | 47,711,301 | 2,622,832 | 99.879% | 9.91% | 99.65% |
| 11 | 47,711,301 | 853,151 | 99.879% | 9.78% | 99.65% |
| 12 | 47,711,301 | 232,325 | 99.879% | 9.75% | 99.65% |

Table 18: Rooted $k$-hop ball amenability on PCQM4Mv2, validation split.

| $k$ | Balls | Computed | Rooted amenable | Ball is a tree | Molecules fully amenable |
|---|---|---|---|---|---|
| 1 | 1,050,647 | 1,050,647 | 100.000% | 99.15% | 100.00% |
| 2 | 1,050,647 | 1,050,294 | 99.994% | 84.27% | 99.92% |
| 3 | 1,050,647 | 1,046,294 | 99.968% | 41.37% | 99.73% |
| 4 | 1,050,647 | 1,010,950 | 99.873% | 28.64% | 98.77% |
| 5 | 1,050,647 | 895,855 | 99.766% | 20.96% | 98.61% |
| 6 | 1,050,647 | 706,536 | 99.658% | 16.79% | 98.56% |
| 7 | 1,050,647 | 496,590 | 99.575% | 14.11% | 98.55% |
| 8 | 1,050,647 | 314,349 | 99.515% | 12.56% | 98.55% |
| 9 | 1,050,647 | 179,846 | 99.472% | 11.75% | 98.55% |
| 10 | 1,050,647 | 95,031 | 99.442% | 11.36% | 98.55% |
| 11 | 1,050,647 | 49,139 | 99.425% | 11.20% | 98.55% |
| 12 | 1,050,647 | 26,751 | 99.415% | 11.14% | 98.55% |