What Expressivity Theory Misses: Message Passing Complexity for GNNs

Niklas Kemper **Tom Wollschläger** **Stephan Günnemann**
School of Computation, Information and Technology & Munich Data Science Institute
Technical University of Munich
{niklas.kemper, t.wollschläger, s.guennemann}@tum.de

Abstract

Expressivity theory, characterizing which graphs a GNN can distinguish, has become the predominant framework for analyzing GNNs, with new models striving for higher expressivity. However, we argue that this focus is misguided: First, higher expressivity is not necessary for most real-world tasks as these tasks rarely require expressivity beyond the basic WL test. Second, expressivity theory's binary characterization and idealized assumptions fail to reflect GNNs' *practical* capabilities. To overcome these limitations, we propose Message Passing Complexity (MPC): a continuous measure that quantifies the difficulty for a GNN architecture to solve a given task through message passing. MPC captures practical limitations like over-squashing while preserving theoretical impossibility results from expressivity theory, effectively narrowing the gap between theory and practice. Through extensive validation on fundamental GNN tasks, we show that MPC's theoretical predictions correlate with empirical performance, successfully explaining architectural successes and failures. Thereby, MPC advances beyond expressivity theory to provide a more powerful and nuanced framework for understanding and improving GNN architectures.

1 Introduction

From weather forecasting to drug design, Graph Neural Networks (GNNs) have shown remarkable success across diverse applications. However, the seminal works by Morris et al. [36] and Xu et al. [48] revealed a key limitation of standard Message Passing Neural Networks (MPNNs): Their ability to distinguish non-isomorphic graphs can be bounded by the Weisfeiler-Leman (WL) graph isomorphism test [45].

In response, significant research effort has focused on developing more expressive architectures that surpass the WL test [27]. The underlying hypothesis is that increased expressivity translates to better empirical performance, with improved benchmark results often attributed to higher expressivity [46]. Isomorphism-based (Iso) expressivity theory, which characterizes the sets of graphs an architecture can distinguish through the WL test and its extensions, has thus become the predominant framework for analyzing MPNNs.

We argue that this focus on iso expressivity is misguided. While it provides valuable impossibility results, we identify two key limitations that prevent it from explaining real-world MPNN performance. First, higher expressivity is often not necessary for real-world tasks: almost all graphs in standard benchmarks are already distinguishable by the basic WL test, making it unclear why higher expressivity would improve performance [54]. Second, iso expressivity theory fails to capture practical model capabilities. It assumes unrealistic conditions like lossless information propagation over unbounded layers, ignoring practical limitations like over-squashing [1]. Moreover, its binary view (can vs. cannot distinguish) offers no insight into the relative difficulty of learning specific real-world tasks.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

To address these limitations, we propose shifting from binary expressivity tests to a continuous complexity measure, MPC, that quantifies the message-passing complexity of *arbitrary* tasks for a given architecture. MPC builds upon a novel probabilistic WL test. It captures practical limitations of MPNNs, such as under-reaching [3] and over-squashing [1], which are known to hinder empirical performance, while preserving impossibility results from iso expressivity theory—effectively narrowing the gap between theory and practice.

Through extensive validation, we show that trends in MPC complexity align with empirical performance across a range of fundamental graph tasks.[1] Notably, success is determined not by iso expressivity but by architectural choices that minimize complexity for specific tasks. For instance, a simple GCN with virtual node outperforms strictly more expressive, higher-order models at long-range tasks, as MPC correctly predicts. By providing a quantitative measure of architectural capabilities for specific tasks, MPC both reveals current model limitations and offers clear optimization targets for future architectural innovations, shifting focus from maximizing expressivity to minimizing task-specific complexity.

In summary, our key contributions are:

- We identify limitations of iso expressivity theory that prevent it from explaining MPNN performance in real-world tasks (§ 3).
- We introduce MPC, a continuous message-passing complexity measure rooted in a novel probabilistic WL test that characterizes task-specific difficulty. MPC captures existing MPNN limitations, such as over-smoothing and under-reaching, while retaining impossibility results from iso expressivity theory (§ 4).
- We extensively validate MPC, showing its consistency with empirical performance and its superiority over classical expressivity theory in explaining real-world MPNN behavior (§ 5).

2 Background

Notation Let $G = (\mathcal{V}_G, \mathcal{E}_G, \boldsymbol{X})$ denote an (undirected) graph with nodes $\mathcal{V}_G$, edges $\mathcal{E}_G$, features $\boldsymbol{X}_v$ for $v \in \mathcal{V}_G$ and adjacency matrix $\boldsymbol{A}$. With $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \mathbf{I}$, define the influence matrix $\boldsymbol{I}_{uv} := \tilde{\boldsymbol{A}}_{uv} / \sum_w \tilde{\boldsymbol{A}}_{uw}$ as the normalized adjacency (with self-loops). Let $d_G(u, v)$ be the shortest path distance between u and v and $N_G(v)$ be the set of neighbors of v. Let $\mathcal{G}$ denote a set of graphs and $\mathcal{G}^*$ the set of all graphs. Let $\mathcal{G}_\mathcal{V} = \{(G, u) \mid G \in \mathcal{G}, u \in \mathcal{V}_G\}$ denote the set of graph-node pairs. We will mainly consider node-level functions or tasks of the form $f : \mathcal{G}_\mathcal{V} \to \mathbb{R}^k$. For brevity, we will often write $f_v(G)$ for $f(G, v)$. We define $\log(0) = -\infty$ and $\{\!\{.\}\!\}$ is a multiset. Let $\mathrm{WL}^l : \mathcal{G}_\mathcal{V} \to \mathbb{R}^k$ denote the color assignment of the l-th round of the WL test, defined as: $\mathrm{WL}^0_v = \boldsymbol{X}_v$ and $\mathrm{WL}^l_v = \mathrm{HASH}\left(\mathrm{WL}^{l-1}_v, \{\!\{\mathrm{WL}^{l-1}_u \mid u \in N_G(v)\}\!\}\right)$.

Message Passing Framework Standard MPNNs $\mathcal{M}_S$ have hidden representations h^l_v for each node v which are updated iteratively at each layer $l \in \{1, \ldots, L\}$ by aggregating messages $m^l_{w \to v}$ from neighboring nodes w in the graph G:

$$m^l_{w \to v} := \begin{cases} \mathtt{msg}^l_0(h^{l-1}_w) & \text{if } w = v \\ \mathtt{msg}^l_1(h^{l-1}_w) & \text{else} \end{cases} \quad \text{and} \quad h^l_v := \mathtt{upd}^l\left(\mathtt{agg}\left(\{\!\{m^l_{w \to v} \mid w \in N_G(v) \cup \{v\}\}\!\}\right)\right)$$

Here, $\mathtt{msg}^l_0$, $\mathtt{msg}^l_1$, and $\mathtt{upd}^l$ can be arbitrary (learned) functions, often MLPs. Typical choices for the aggregation function $\mathtt{agg}$ are mean or sum. We differentiate between a *model architecture* $\mathcal{M}$ that only specifies which nodes exchange messages (abstracting away from the choice of $\mathtt{msg}$, $\mathtt{upd}$, and $\mathtt{agg}$), and a (learned) *model instantiation* $M \in \mathcal{M}$ with fully specified $\mathtt{msg}$, $\mathtt{upd}$ and $\mathtt{agg}$ functions. Standard MPNNs $\mathcal{M}_S$ perform message passing directly on the input graph G. In contrast, more recent architectures propagate messages on a transformed message passing (MP) graph $\tilde{G} = t(G)$ [42]. They introduce modifications like additional virtual nodes [18], rewired edges [41], or higher-order graphs [46]. For simplicity, we mainly focus on standard MPNNs $\mathcal{M}_S$ in the main text, and defer a general framework encompassing architectures $\mathcal{M}$ with arbitrary MP graphs $\tilde{G}$ to App. B.1.

[1] Find our implementation at https://www.cs.cit.tum.de/daml/message-passing-complexity/

3 Limitations of Expressivity Theory

MPNN architectures differ in their theoretical capacity to solve graph tasks, such as detecting specific substructures [53]. Most prior expressivity theory captures differences by considering an architecture's ability to distinguish non-isomorphic graphs relative to a reference isomorphism test α [22, 5, 46, 37]. Formally:

Definition 3.1 (Iso Expressivity)**.** Let α be a graph isomorphism test. An architecture $\mathcal{M}$ is at least as expressive as α if $\exists M \in \mathcal{M}$ such that $\forall G, G' \in \mathcal{G}^*$:

$$M(G) = M(G') \implies \alpha(G) = \alpha(G'). \tag{1}$$

Contrarily, $\mathcal{M}$ is at most as expressive as α if $\forall M \in \mathcal{M}$ and $\forall G, G' \in \mathcal{G}^*$

$$\alpha(G) = \alpha(G') \implies M(G) = M(G'). \tag{2}$$

Standard MPNNs $\mathcal{M}_S$ are at most as expressive as the WL test [36, 48]. In response, recent GNN research has focused on developing architectures that surpass the WL test in expressivity. Their strong performance on benchmarks such as ZINC [5, 8, 16, 46] is often motivated by and attributed to their higher iso expressivity [46, 8, 16]. This builds on the premise that theoretical iso expressivity correlates with empirical performance. Consequently, this line of expressivity research rests on two fundamental assumptions, both of which we show to be problematic (Fig. 1):

1. *Iso expressivity theory accurately describes the practical capabilities of trained MPNNs.* The theoretical higher expressivity of higher-order MPNNs translates to a higher practical capability to learn more complex tasks, contributing to improved empirical performance.
2. *Iso expressivity beyond WL is important for real-world performance.* The limited expressivity of standard MPNNs restricts their performance on real-world tasks.

We first critique Assumption 1, showing that iso expressivity theory relies on idealized assumptions and provides incomplete insights about practical MPNN capabilities. Positive expressivity results only guarantee the existence of a model instance within the architecture that is more expressive than α. These proofs typically rely on a maximally expressive model M^* with injective upd functions and an unbounded number of layers [5]: conditions under which every node can access the complete information from all other nodes. This idealized setup ignores fundamental limitations of real-world MPNNs. In practice, message passing is lossy: information is often bottlenecked (over-squashing [1]), blurred (over-smoothing [33]), or simply out of reach due to shallow depth (under-reaching [3]). As a result, real-world MPNNs often fail to propagate information even between nearby nodes, let alone replicate the idealized behavior assumed in expressivity theory. Second, expressivity theory provides strictly binary results [38]: can vs. cannot distinguish, with no indication of difficulty. This binary lens cannot account for the wide variation in empirical performance between architectures of equal expressivity on the same task. For example, adding a virtual node does not increase iso expressivity [46] but often leads to performance gains on long-range tasks [9]. Additionally, *between* theoretically solvable tasks, iso expressivity theory provides no insight into their relative difficulty. Some tasks might be trivially learnable in practice while others may be practically impossible to learn with finite data and training time. In summary, the assumptions behind positive expressivity results diverge sharply from the realities of real-world MPNNs, and the binary view can offer only limited insights for practice. Our empirical results in § 5 reinforce this, showing that even highly expressive MPNN architectures struggle with elementary tasks like maintaining initial node features. This highlights a gap between positive iso expressivity results and practical performance.

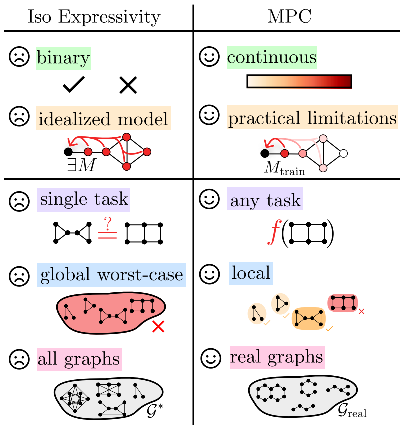


Figure 1: Limitations of iso expressivity vs. benefits of MPC. Top: Iso expressivity gives an idealized, binary view that misses GNNs' practical capabilities. Bottom: Limited expressivity rarely restricts real-world performance as it focuses on worst-case graphs and tasks.

We now challenge Assumption 2, asking whether negative expressivity results reflect meaningful limitations in practice. Iso expressivity theory considers only the hardest possible task: distinguishing all non-isomorphic graphs. However, many real-world applications, such as social network analysis or relational learning, often require only aggregating (local) information. In these cases, the inability to distinguish certain non-isomorphic graphs does not constitute a meaningful limitation in practice. By focusing exclusively on this worst-case task, expressivity theory overlooks the specific requirements of practical problems. Second, expressivity theory makes global statements over the set of graphs $\mathcal{G}^*$, demonstrating a lack of expressivity through single, carefully constructed counterexamples [46, 22, 5]. However, these counterexamples are rare: the probability of encountering WL equivalent graphs in random graphs approaches 0 [2], and real-world graphs often carry rich node features that further break WL equivalences. Hence, the mere existence of theoretical counterexamples provides limited insight into a model's practical performance on large, diverse datasets. Lastly, iso expressivity theory is graph-family agnostic. Its results are derived for the set of all possible graphs $\mathcal{G}^*$, whereas real-world applications typically involve restricted graph families. For example, molecular datasets primarily contain planar graphs with bounded degrees. Tasks infeasible over all graphs may become solvable for standard MPNNs within these restricted graph families [13, 40]. Moreover, as shown in Tab. 1, the basic *WL test already distinguishes almost all graph pairs in popular benchmarks* across different domains [54]. Therefore, iso expressivity theory cannot explain why expressivity beyond the WL test would benefit these real-world tasks or account for the performance differences between architectures.

Our analysis shows that iso expressivity theory, while valuable for establishing theoretical limitations, provides limited insights for practical MPNN applications. It assumes idealized conditions and focuses on theoretical worst cases, leading to a disconnect with practice: high iso expressivity does not imply good performance (contradicting Assumption 1), and limited iso expressivity does not imply poor performance (contradicting Assumption 2). While alternative expressivity approaches such as logic-based characterizations address certain limitations of iso expressivity (see § 7), none can characterize the varying degrees of practical learning difficulty encountered in real-world tasks. These findings suggest that pursuing higher expressivity alone may be misguided, highlighting the need for a framework that captures theoretical limitations *and* practical learning challenges.

4 Message Passing Complexity

To move beyond the limitations of expressivity theory, we propose our continuous, task-specific message passing complexity (MPC). Unlike iso expressivity theory, which only asks whether an MPNN can distinguish certain graphs in theory, MPC quantifies *how difficult* it is for messages to propagate through a graph *to solve a given task*. By accounting for the inherently lossy information propagation of real-world MPNNs, MPC can explain practical performance trends that expressivity theory misses. While many complexity measures exist in machine learning and theoretical computer science, they fail to address this unique challenge of MPNNs: propagating information effectively across graph structures. MPC specifically isolates this challenge from other general sources of difficulty in machine learning—for instance, learning high-degree polynomials is a well-understood difficulty not specific to graph learning. For clarity, we focus on node-level tasks f_v and standard MPNNs $\mathcal{M}_S$ in this section, with generalizations to broader architectures and tasks provided in App. B.1.

As an initial step, we define a local task-specific complexity measure based on the WL test that considers individual graphs and individual tasks rather than making global worst-case statements over all graphs and tasks. For this, we first need to formalize when one individual function output provides sufficient information to determine another (Fig. 7):

Definition 4.1. Let $\alpha : X \to Y$, $\beta : X \to Z$ be two functions. Let $x \in X$ be fixed. Then, $\beta(x)$ *can be deduced* from $\alpha(x)$, $\alpha(x) \vDash_X \beta(x)$, iff

$$\forall x' \in X : \ \alpha(x) = \alpha(x') \Rightarrow \beta(x) = \beta(x').$$

If $\alpha(x) \vDash_X \beta(x)$ for all $x \in X$, we write $\alpha \vDash_X \beta$ and say α is more *fine-grained* than β.

Intuitively, $\alpha(x) \vDash_X \beta(x)$ means that $\alpha(x)$ provides sufficient information to uniquely determine $\beta(x)$. Using this, we can define a first complexity measure dependent on a specific task f_v and individual graph G. The complexity should be maximal if a task f_v is infeasible for all $M \in \mathcal{M}_S$, i.e., the WL coloring $\mathrm{WL}_v^L(G)$ of v in the graph G contains insufficient information to deduce $f_v(G)$:

Definition 4.2. For $G, v \in \mathcal{G}_\mathcal{V}$, a task f over $\mathcal{G}_\mathcal{V}$ and an L-layer standard MPNN $\mathcal{M}_S$, define

$$\mathrm{WLC}_{\mathcal{M}_S}(f_v, G) = \begin{cases} \infty & \text{if } \mathrm{WL}_v^L(G) \nvDash_{\mathcal{G}_\mathcal{V}} f_v(G) \\ 0 & \text{else.} \end{cases}$$

While WLC considers specific tasks f_v and graphs G, it still inherits a fundamental limitation of iso expressivity: it only distinguishes between possible (complexity 0) and impossible (complexity ∞) tasks. This binary characterization fails to capture the varying difficulties observed in practice. Moreover, as discussed in § 3, the WL test assumes lossless information propagation between nodes, contrasting sharply with the lossy message passing observed in trained real-world MPNNs.

4.1 Weisfeiler & Leman Go Lossy

To extend our complexity measure beyond the binary characterization of the standard WL test, we need to account for the varying difficulty arising from the lossy message passing observed in practice. This difficulty inherently relies on the graph topology, i.e., the difficulty of propagating a message from one node u to another node v depends on 1) the number of L-length walks connecting v and u and 2) the degrees of the nodes on the walk. This can be formalized as the random walk probability $\boldsymbol{I}_{vu}^L$ from v to u with edge probabilities $\boldsymbol{I}$ and has been connected to the amount of gradient information node v receives from u [47] and to the oversquashing phenomenon [19]. Correspondingly, $\boldsymbol{I}_{vu}^L$ serves as a measure of difficulty for the simple task of propagating a message from u to v.

How can we adapt this measure of difficulty to *arbitrary* tasks? For this, we propose lossyWL, a probabilistic variant of the WL test that models the possibility of message loss. This allows us to quantify difficulty even for WL-distinguishable real-world graphs. Specifically, a message $m_{u \to v}^l$ sent from a node u to its neighbor $v \in N_G(u)$ survives with probability $\boldsymbol{I}_{vu}$ (Fig. 2). As a result, the color $\mathrm{lossyWL}_v^l$ of a node v at iteration l becomes a random variable rather than a deterministic value.

Definition 4.3 (lossyWL)**.** For a given graph G, let $m_{u \to v}^l$ be the (potentially lost) message from u to v at iteration l:

$$m_{u \to v}^l = Z_{uv}^l \cdot \mathrm{lossyWL}_u^{l-1}$$

with independent $Z_{uv}^l \sim \mathrm{Bernoulli}(\boldsymbol{I}_{vu})$, indicating whether the message is successfully transmitted. The node colors of the probabilistic WL tests are then recursively defined as:

$$\begin{aligned} \mathrm{lossyWL}_v^0 &= \boldsymbol{X}_v \\ \mathrm{lossyWL}_v^l &= \mathrm{HASH}\left(m_{v \to v}^l, \{\!\{ m_{u \to v}^l \mid u \in N_G(v) \}\!\}\right) \end{aligned}$$

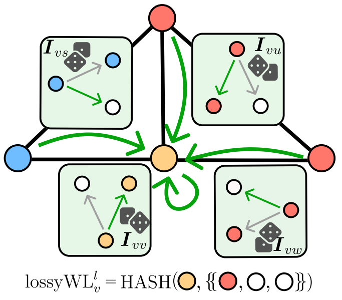


Figure 2: Update step of lossyWL for node v. Every message $m_{a \to v}^l$ (green) survives independently with probability $\boldsymbol{I}_{va}$. lossyWL models the lossy message propagation observed in real-world MPNNs.

Similar to how we can transform the WL test into WLC, we can define our complexity measure, MPC, based on our probabilistic lossyWL test. To do this, we first extend the Definition 4.1 of $\vDash$ to account for probabilistic functions such that $\mathbb{P}[\alpha(x) \vDash \beta(x)]$ represents the probability that $\beta(x)$ can be uniquely deduced from the probabilistic output of $\alpha(x)$ (Fig. 8).

Definition 4.4. Let $\beta : X \to Z$ be a deterministic function and $\alpha : X \to Y$ be a probabilistic function that can be represented as a deterministic function α_s where s is a seed drawn uniformly at random from a (finite) set S. For a fixed $x \in X$, define:

$$\mathbb{P}[\alpha(x) \vDash_X \beta(x)] := \mathbb{P}_{s \in S}[\forall x' \in X, \forall s, t \in S\colon\ \alpha_s(x) = \alpha_t(x') \Rightarrow \beta(x) = \beta(x')]$$

With this in place, we define our message passing complexity: intuitively, if there is a high probability of preserving the information needed for f_v, the task should have low complexity, and vice versa.

Definition 4.5 (MPC)**.** For $G, v \in \mathcal{G}_\mathcal{V}$, a function f over $\mathcal{G}_\mathcal{V}$ and an L-layer MPNN $\mathcal{M}_S$, define

$$\mathrm{MPC}_{\mathcal{M}_S}(f_v, G) = -\log \mathbb{P}[\mathrm{lossyWL}_v^L(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)]$$

MPC is based on the probability that under lossy message passing on G, the output after L layers contains sufficient information to deduce f_v. Intuitively, a high MPC value indicates that the task f_v requires 1) combining information from many nodes through 2) messages of low probability, such as messages through bottlenecks. Conversely, a low MPC value means the task depends on information that remains reliably accessible even under lossy message passing conditions. Note that MPC implicitly also depends on $\mathcal{G}$, which we omit for simplicity. While we focus here on standard MPNNs $\mathcal{M}_S$, MPC generalizes naturally to a wide range of architectures $\mathcal{M}$ by performing lossyWL on modified message passing graphs (see App. B.1). All in all, MPC effectively addresses the discussed limitations of iso expressivity theory, narrowing the gap between theory and practice.

4.2 MPC describes theoretical and practical limitations of MPNNs

Having defined our complexity measure MPC, we now demonstrate that it unifies theoretical expressivity with practical GNN limitations in a single framework. We first establish that MPC preserves impossibility results from iso expressivity theory, then show how it captures practical phenomena such as over-squashing and under-reaching. We begin by characterizing when MPC becomes infinite, which precisely corresponds to tasks that are theoretically impossible for a given architecture $\mathcal{M}$.[2]

Theorem 4.6 (Infeasibility)**.** *The complexity for $G, v \in \mathcal{G}_{\mathcal{V}}$ and function f is* $\text{MPC}_{\mathcal{M}}(f_v, G) = \infty$ *if and only if there exist $G', w \in \mathcal{G}_{\mathcal{V}}$ such that $f_v(G) \neq f_w(G')$ but $M_v(G) = M_w(G')$ for all model instantiations $M \in \mathcal{M}$.*

This shows that MPC becomes infinite precisely when a model architecture fundamentally cannot distinguish between two different nodes that require different outputs. We can also globally characterize the functions an architecture can express:

Lemma 4.7. *There exists no model instantiation $M \in \mathcal{M}$ such that $M_v(G) = f_v(G)$ for all $G, v \in \mathcal{G}_{\mathcal{V}}$ if and only if there exists $G, v \in \mathcal{G}_{\mathcal{V}}$ with* $\text{MPC}_{\mathcal{M}}(f_v, G) = \infty$*.*

By choosing an isomorphism test (like the WL test) as task f, we can recover all impossibility statements from iso expressivity theory (Lemma C.7). Thus, MPC subsumes iso expressivity theory while providing a continuous difficulty measure for possible tasks. Having established when MPC becomes infinite, we now characterize how MPC scales with function granularity: a finer, more discriminative task can never have lower complexity than the coarser task it refines. For example, counting the exact number of cycles in a graph should be more complex than merely detecting whether any cycle exists.

Theorem 4.8 (Function refinement)**.** *Let f be a function that is more fine-grained than g, i.e., $f \vDash_{\mathcal{G}_{\mathcal{V}}} g$. Then for any $G, v \in \mathcal{G}_{\mathcal{V}}$:* $\text{MPC}_{\mathcal{M}}(f_v, G) \geq \text{MPC}_{\mathcal{M}}(g_v, G)$*.*

Like other complexity measures, MPC satisfies compositionality: solving tasks jointly cannot be more complex than solving them separately, and may be easier when tasks share information.

Lemma 4.9 (Task Triangle Inequality)**.** *Let f and g be functions, and $\|$ denote concatenating function outputs. Then for any $G, v \in \mathcal{G}_{\mathcal{V}}$:* $\text{MPC}_{\mathcal{M}}(f_v \| g_v, G) \leq \text{MPC}_{\mathcal{M}}(f_v, G) + \text{MPC}_{\mathcal{M}}(g_v, G)$*.*

However, MPC connects not only to properties of classical complexity and expressivity theory but also captures real-world limitations of MPNNs such as over-squashing and under-reaching. These phenomena, extensively studied as significant constraints on MPNN performance, are overlooked by traditional expressivity theory. We first relate MPC to over-squashing by showing that it is lower bounded in terms of the L-step random walk probability from v to u, $(\boldsymbol{I}^L)_{vu}$, a quantity that also motivated our design of lossyWL in § 4.1.

Lemma 4.10 (Informal version of Lemma C.10)**.** *Consider a task f and $G, v \in \mathcal{G}_{\mathcal{V}}$ where f_v "depends on" information $\boldsymbol{X}_u$ from a node u. Then*

$$\text{MPC}_{\mathcal{M}}(f_v, G) \geq -\log\left((\boldsymbol{I}^L)_{vu}\right).$$

A lower probability $(\boldsymbol{I}^L)_{vu}$ indicates that node v receives less gradient signal from u, making it more susceptible to over-squashing [19, 47]. Our complexity measure captures this: when $(\boldsymbol{I}^L)_{vu}$ is small, tasks requiring information flow from u to v have high complexity. As a simple special case, we consider under-reaching, which occurs when the number of MPNN layers is less than the graph diameter, preventing nodes from receiving information from distant parts of the graph [3, 1]:

[2] We defer all proofs to App. C.

Corollary 4.11. *Consider a task f_v and a graph $G \in \mathcal{G}$ where f_v "depends on" information $\boldsymbol{X}_u$ from a node u outside of the receptive field, i.e. $d_{\tilde{G}}(u,v) > L$. Then* $\text{MPC}_{\mathcal{M}}(f_v, G) = \infty$.

These results show that MPC captures both theoretical expressivity and practical limitations of GNNs, providing a more unified framework that bridges both perspectives.

5 MPC in Practice: Explaining Empirical MPNN Behavior

We now demonstrate that MPC quantitatively explains *empirical* MPNN behavior. Through analysis of fundamental graph tasks—retaining information, propagating information, and extracting topological features—we show that: 1) MPC enables the derivation of meaningful task-specific complexity bounds; 2) trends in MPC complexity correlate strongly with empirical performance, better reflecting the practical behavior of MPNNs than iso expressivity theory; 3) optimal architectures vary by task, with success determined by minimizing task-specific MPC complexity through appropriate inductive biases, not maximizing expressivity. **Experimental Setup:** We verify this for a wide range of architectures, including an MLP baseline (message passing on an empty graph), standard models like GCN [31], GIN [48], GraphSage [25], and a GCN with a virtual node, GCN-VN [18], as well as higher-order models like GSN [8] which incorporates substructure-information as node-feature, FragNet [46] which builds a higher-level graph of fragments and the topological-inspired model CIN [5]. We evaluate on random r-regular graphs $\mathcal{G}$ and show that the results transfer to graphs from the ZINC dataset [15] and the long-range graph benchmark [14]. We provide theoretical complexity bounds and Monte-Carlo simulated complexity values. We compare MPC to the WL-based WLC baseline rather than directly to other expressivity measures, as they have fundamental limitations for task-specific difficulty analysis: they typically provide only global, task-agnostic architecture rankings [53, 52] or impose restrictive assumptions incompatible with our tasks [20] (see § 7).

Retaining information First, we evaluate the task $f_v(G) = \boldsymbol{X}_v$, which tests a model's ability to retain its initial node features. This task is fundamentally important across all domains with informative node features (e.g., atom type in molecules). Although seemingly simple, it is closely linked to the well-studied *over-smoothing* [33] phenomenon, where deeper GNNs lose distinguishability between nodes as their representations converge, making it difficult to recover the original node features [21]. Our complexity measure captures this difficulty as MPC increases at least linearly with depth L:

Lemma 5.1. *Assume the degree $r \geq 2$. Then the expected* MPC *complexity for this task* $\mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G)]$ *grows at least linearly with L, i.e., is in $\Omega(L)$, for all MPNNs $\mathcal{M}$.*

For empirical validation, we train all model architectures on 2000 randomly generated 3-regular graphs with varying numbers of layers L. As shown in Figs. 3 and 9, the complexity measure based on iso expressivity theory, WLC, assigns constant zero complexity regardless of depth L, indicating only theoretical solvability. In contrast, MPC shows perfect negative Spearman correlation with accuracy for most architectures, capturing the increasing difficulty with L. Only CIN and GraphSage maintain perfect accuracy throughout, due to explicit residual connection optimization—an implementation choice our framework abstracts from. These results demonstrate that, unlike binary expressivity measures, MPC accurately quantifies the progressive difficulty of this task for most architectures, capturing important real-world limitations beyond theoretical impossibility statements.

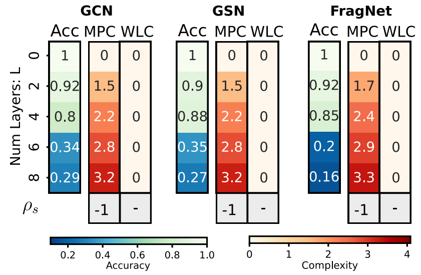


Figure 3: Test accuracy for retaining initial node features compared with complexity measures MPC and WLC. Simulated MPC (in contrast to WL-based WLC) shows perfect negative Spearman correlation ($\rho_s = -1$) with accuracy, capturing increasing difficulty with depth (over-smoothing). Complete results in Fig. 9.

Propagating Information We next analyze the task $f_v(G) = \boldsymbol{X}_u$, where u is a specially marked source node at distance $D = d_G(u,v)$ from target node v. This task directly tests a model's ability to propagate information in relation to the distance D, exposing practical limitations like over-squashing and under-reaching that classical expressivity theory overlooks.

Lemma 5.2. *Assume L is the minimum depth required to solve this task with $\mathcal{M}$. Then the expected complexity* $\mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G)]$ *is $\leq 2\log(n)$ for GCN-VN, while for standard MPNNs it is $\geq D\log(r)$ provided n is sufficiently large given D.*

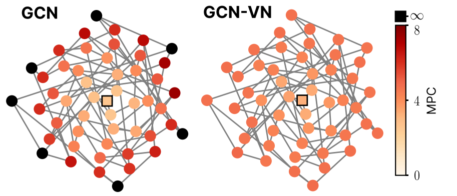


Figure 4: Simulated MPC complexities for propagating features from source nodes u (colored by MPC) to target node v (square). Despite identical iso expressivity, MPC reveals the significant advantage virtual nodes offer for long-range dependencies.

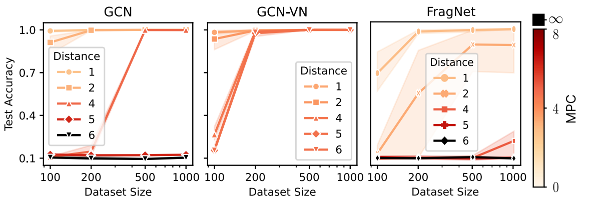


Figure 5: Test accuracy vs. training data size for the propagation task $f_v(G) = \boldsymbol{X}_u$ for different distances D. Colors indicate average simulated MPC for each distance. Higher MPC values reflect greater task difficulty, evidenced by increased sample complexity. All results in Fig. 13; for real-world graphs from lrgb in Fig. 17.

The bounds reveal that MPC captures two critical insights missed by classical expressivity: First, MPC increases with distance D for most MPNNs, explaining why they struggle with long-range dependencies despite theoretical learnability. Second, a virtual node fundamentally changes complexity scaling from $O(D)$ to $O(\log n)$, explaining its empirical advantage despite unchanged expressivity (see Fig. 4). Moreover, MPC also captures under-reaching: by Corollary 4.11, complexity becomes infinite when $d_{\tilde{G}}(u, v) > L$.

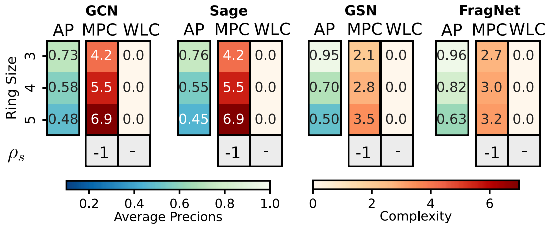
Experiments for models with $L = 5$ layers validate these predictions: MPC correlates negatively with accuracy (Fig. 14) and captures the increased sample complexity with distance (Figs. 5 and 13). Crucially, all models except GCN-VN fail at $D = 5$, showing that high MPC complexity indicates practical limitations even before the theoretical impossibility at $D > 5$. Importantly, these findings—strong MPC-performance correlation and practical failure before theoretical limits—persist in real-world graphs from the long-range graph benchmark (Fig. 17).

In summary, these results demonstrate why MPC offers a more complete and practical understanding of MPNN capabilities than classical expressivity theory. Unlike binary expressivity tests, MPC explains why even highly expressive architectures, such as GSN, FragNet, and CIN, struggle with fundamental tasks like retaining or propagating node information—empirically validating our critique in § 3 that iso expressivity poorly captures practical capabilities. Furthermore, MPC accounts for the success of architectural choices like virtual nodes, which consistently improve performance on real-world long-range tasks [9] despite unchanged iso expressivity [46].

Topological Feature Extraction Our final experiment examines how models extract topological features through cycle detection. We consider random r-regular graphs with unique node labels $\boldsymbol{X}_v \in \{1, \ldots, n\}$, modified to contain a cycle of size s that includes a designated node v. The task f_v is to identify the labels of all nodes in this cycle, jointly testing the model's ability to detect cycles and propagate information across them. We can derive complexity bounds for all considered MPNNs:

Lemma 5.3 (Informal version of Lemma C.13)**.** *Assume L is the minimal depth required to solve this task with architecture $\mathcal{M}$ and that there is only a single cycle in the $\lceil s/2 \rceil$-hop neighborhood of v. Then, the expected* MPC *complexities are:*

For CIN & FragNet: $O(\log(sr))$ *For GSN:* $\leq \lceil s/2 \rceil \log(r+1)$ *For all others:* $\geq s \log(r)$



	GCN			Sage			GSN			FragNet		
Ring Size	AP	MPC	WLC	AP	MPC	WLC	AP	MPC	WLC	AP	MPC	WLC
3	0.73	4.2	0.0	0.76	4.2	0.0	0.95	2.1	0.0	0.96	2.7	0.0
4	0.58	5.5	0.0	0.55	5.5	0.0	0.70	2.8	0.0	0.82	3.0	0.0
5	0.48	6.9	0.0	0.45	6.9	0.0	0.50	3.5	0.0	0.63	3.2	0.0
ρ_s		-1	-		-1	-		-1	-		-1	-

Figure 6: Average precision (AP) for the ring task compared with complexity values MPC and WLC (MPC values for SAGE & GCN are lower bounds). MPC (in contrast to WLC) is negatively Spearman correlated with AP and captures the advantage of cycle-oriented inductive biases in GSN and FragNet. See also Figs. 18 and 23.

These bounds reveal that MPC captures crucial architectural differences missed by iso expressivity: standard MPNNs have complexity scaling linearly with cycle size, while CIN and FragNet achieve logarithmic scaling, capturing their cycle-oriented inductive biases. Experiments on random 4-regular graphs confirm these theoretical predictions, with trends in MPC complexity matching empirical performance (see Figs. 6 and 18; in relation to dataset size: Fig. 19).

Because of the unique node labels, all (at least WL expressive) architectures are theoretically able to solve this task, yet performance varies dramatically. While iso expressivity theory cannot capture these differences, MPC provides a clear explanation: GSN, CIN and FragNet's architectural bias towards cycles reduces their message-passing complexity for this task, providing them with a performance advantage. This insight extends to real-world graphs from the ZINC molecular regression benchmark [15] (Fig. 6), where the identification of molecular rings is crucial. As shown in Tab. 2, architectures with lower ring detection complexity consistently achieve better performance on ZINC on both the synthetic and the real task. The superior performance of higher-order MPNNs like GSN, FragNet or CIN cannot be explained by their greater iso expressivity—as discussed in § 3, almost all graphs in ZINC are already WL-distinguishable—but by their cycle-oriented inductive biases that reduce the complexity of learning molecular structures.

Overall, our results demonstrate that trends in MPC consistently correlate with empirical performance across all tested tasks, offering a more fine-grained, complete, and practical understanding of MPNN capabilities than iso expressivity theory's idealized binary characterization. Crucially, we find that no single architecture performs best across all tasks—performance depends not on iso expressivity but on the alignment between a model's inductive biases and the task requirements. While the global task-agnostic architecture rankings of most existing expressivity measures fail to capture this variation, MPC quantifies the alignment between architecture and task—explaining why virtual nodes excel at long-range tasks or why cycle-aware designs help for ring-transfer. This provides task-specific guidance for architectural design that pure expressivity theory cannot offer.

6 Limitations & Future Work

While we have demonstrated that our complexity measure closely aligns with empirical performance for many tasks, it abstracts away from the parametrization and implementation of the upd and agg functions and the optimization process as discussed in § 5. Additionally, our MPNN framework does not include models with attention. However, we repeat all our experiments with the most common attention-based MPNNs in App. D.1, showing similar empirical trends to the MPNNs that we consider. Secondly, exact target functions in real-world applications are usually unknown. Still, we can analyze two types of proxy tasks: fundamental capabilities required across all domains (like retaining information) and domain-specific operations (like cycle detection). As shown in Tab. 2, the insights from these proxy tasks can translate to real-world performance. Third, exact complexity values can be difficult to compute for sophisticated tasks. However, as our experiments demonstrate, theoretical bounds can provide valuable practical insights. Finally, the complexity values should not be interpreted in isolation to determine whether an architecture can solve a given task, as they only characterize the *message-passing* complexity of a task. Instead, they are most useful for identifying trends and comparing architectures across a single dimension of freedom.

These limitations point to promising extensions of the MPC framework. MPC could be extended to incorporate additional sources of practical difficulty, such as feature noise (App. B.3), providing a more complete view of empirical task complexity. Furthermore, the framework could be modified to capture aggregation and update-specific effects by incorporating non-uniform or learnable message weights Z_{uv} (as in attention-based models), enabling analysis of how different update functions affect message passing complexity.

More broadly, we hope that MPC will enable more principled architectural design: By quantifying fine-grained task complexities, MPC can reveal *practical* limitations of current architectures and guide the development of models with low complexity *for specific capabilities* relevant to target domains. As a concrete example, consider substructure identification. Our analysis in § 5 shows that all considered MPNNs without additional pre-processing steps exhibit very high complexity for this task, even when it is theoretically solvable. This suggests that architectural modifications alone may be insufficient, pointing toward studying expressive preprocessing steps or positional encodings —as successfully employed by FragNet [46] and CIN [5]— that directly reduce task complexity. More generally, this exemplifies how MPC can shift architectural design from maximizing universal expressivity toward strategically minimizing complexity for domain-specific requirements.

7 Related Work

Iso-expressivity frameworks. The predominant approach for analyzing GNNs compares their ability to distinguish non-isomorphic graphs relative to the WL test and its extensions [36, 48, 37]. This has motivated extensive research on developing higher-order architectures that surpass standard MPNNs in expressivity [37, 5, 22, 46]. However, as discussed in § 3, iso-expressivity theory relies on idealized assumptions and provides only binary characterizations, limiting its practical relevance.

Alternative expressivity characterizations. Beyond iso-expressivity, several frameworks theoretically characterize MPNN capabilities through specific tasks. Zhang et al. [53] rank architectures by the set of substructures they can recognize, while Zhang et al. [52] compare models through their ability to solve graph biconnectivity. Wang and Zhang [44] analyze spectral MPNNs through their ability to learn polynomial filters. Logic-based approaches [3, 23] characterize learnable functions through fragments of first-order logic, providing more nuanced insights by considering, for example, the effects of different activation functions [29]. However, these expressivity characterizations share key limitations with iso expressivity: they remain binary (can/cannot solve) and assume lossless information propagation, limiting their insights for real-world MPNNs.

Continuous graph distances. To move beyond binary expressivity, some works have proposed continuous graph similarity measures, including tree-based [12], graphon-based [7], and Wasserstein-based distances [10]. However, these metrics are task-agnostic and architecture-independent, limiting their relevance for explaining practical GNN performance.

Practical GNN limitations. A parallel line of research studies individual *practical* limitations of GNNs, such as over-smoothing [33], under-reaching [3], and over-squashing [1, 19, 41]. Most relevant to our approach is the work by Giovanni et al. [20], which derives expressivity limitations from over-squashing theory. They show tasks become impossible when the required "mixing" between nodes (measured via the Hessian) exceeds what MPNNs can generate. Like MPC, this can capture practical impossibilities arising from over-squashing. However, their framework considers only pairwise interactions with restrictive assumptions on the task (twice differentiable tasks, not dependent on graph topology).

While these works study individual theoretical or practical limitations of GNNs, MPC is the only framework that captures both theoretical expressivity and practical limitations while allowing analysis of arbitrary tasks.

8 Conclusion

We show that classical expressivity theory cannot explain MPNN performance in real-world settings. To narrow the gap between theory and practice, we propose MPC: a continuous complexity measure that quantifies the message-passing difficulty of tasks for different architectures. By building upon a novel probabilistic variant of the WL test, MPC retains all impossibility results from iso expressivity theory while capturing practical limitations like over-squashing and under-reaching. Our extensive validation on fundamental MPNN tasks reveals that trends in MPC complexity correlate with empirical performance, explaining phenomena that iso expressivity theory cannot address. Notably, our analysis indicates that the success of (higher-order) MPNNs often stems from low task complexity rather than increased iso expressivity. This perspective shifts focus from maximizing expressivity to minimizing task-specific MPC complexity, providing clear directions for architectural innovation.

Acknowledgments

We would like to thank Nicholas Gao and Aman Saxena for proofreading this manuscript, and the anonymous reviewers for their constructive comments. This research is supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy with funds from the Hightech Agenda Bayern.

References

[1] U. Alon and E. Yahav. On the bottleneck of graph neural networks and its practical implications. In *ICLR*, 2021.

[2] L. Babai, P. Erdös, and S. M. Selkow. Random graph isomorphism. *SIAM J. Comput.*, 9(3): 628–635, 1980.

[3] P. Barceló, E. V. Kostylev, M. Monet, J. Pérez, J. L. Reutter, and J. P. Silva. The logical expressiveness of graph neural networks. In *ICLR*, 2020.

[4] F. Bause, F. Jogl, P. Indri, T. Drucks, D. Penz, N. M. Kriege, T. Gärtner, P. Welke, and M. Thiessen. Maximally expressive gnns for outerplanar graphs. *Trans. Mach. Learn. Res.*, 2025, 2025.

[5] C. Bodnar, F. Frasca, N. Otter, Y. Wang, P. Liò, G. F. Montúfar, and M. M. Bronstein. Weisfeiler and lehman go cellular: CW networks. In *NeurIPS*, pages 2625–2640, 2021.

[6] C. Bodnar, F. Frasca, Y. Wang, N. Otter, G. F. Montúfar, P. Lió, and M. M. Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 1026–1037. PMLR, 2021.

[7] J. Böker, R. Levie, N. Huang, S. Villar, and C. Morris. Fine-grained expressivity of graph neural networks. In *NeurIPS*, 2023.

[8] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45 (1):657–668, 2023.

[9] C. Cai, T. S. Hy, R. Yu, and Y. Wang. On the connection between MPNN and graph transformer. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 3408–3430. PMLR, 2023.

[10] S. Chen, S. Lim, F. Mémoli, Z. Wan, and Y. Wang. Weisfeiler-lehman meets gromov-wasserstein. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 3371–3416. PMLR, 2022.

[11] Z. Chen, L. Chen, S. Villar, and J. Bruna. Can graph neural networks count substructures? In *NeurIPS*, 2020.

[12] C. Chuang and S. Jegelka. Tree mover's distance: Bridging graph metrics and stability of graph neural networks. In *NeurIPS*, 2022.

[13] R. Dimitrov, Z. Zhao, R. Abboud, and İ. İ. Ceylan. Plane: Representation learning over planar graphs. In *NeurIPS*, 2023.

[14] V. P. Dwivedi, L. Rampásek, M. Galkin, A. Parviz, G. Wolf, A. T. Luu, and D. Beaini. Long range graph benchmark. In *NeurIPS*, 2022.

[15] V. P. Dwivedi, C. K. Joshi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson. Benchmarking graph neural networks. *J. Mach. Learn. Res.*, 24:43:1–43:48, 2023.

[16] J. Feng, L. Kong, H. Liu, D. Tao, F. Li, M. Zhang, and Y. Chen. Extending the design space of graph neural networks by rethinking folklore weisfeiler-lehman. In *NeurIPS*, 2023.

[17] M. Fey and J. E. Lenssen. Fast graph representation learning with pytorch geometric. *CoRR*, abs/1903.02428, 2019.

[18] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 2017.

[19] F. D. Giovanni, L. Giusti, F. Barbero, G. Luise, P. Lio, and M. M. Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 7865–7885. PMLR, 2023.

[20] F. D. Giovanni, T. K. Rusch, M. M. Bronstein, A. Deac, M. Lackenby, S. Mishra, and P. Velickovic. How does over-squashing affect the power of gnns? *Trans. Mach. Learn. Res.*, 2024, 2024.

[21] J. H. Giraldo, K. Skianis, T. Bouwmans, and F. D. Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *CIKM*, pages 566–576. ACM, 2023.

[22] L. Giusti, T. Reu, F. Ceccarelli, C. Bodnar, and P. Liò. CIN++: enhancing topological message passing. *CoRR*, abs/2306.03561, 2023.

[23] M. Grohe. The descriptive complexity of graph neural networks. *TheoretiCS*, 3, 2024.

[24] A. A. Hagberg, D. A. Schult, and P. J. Swart. Networkx: Network analysis with python. pages 11–15, 2008.

[25] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NIPS*, pages 1024–1034, 2017.

[26] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[27] S. Jegelka. Theory of graph neural networks: Representation and learning. *CoRR*, abs/2204.07697, 2022.

[28] C. I. Kanatsoulis and A. Ribeiro. Counting graph substructures with graph neural networks. In *ICLR*, 2024.

[29] S. Khalife and J. Tonelli-Cueto. Is uniform expressivity too restrictive? towards efficient expressivity of graph neural networks. *CoRR*, abs/2410.01910, 2024.

[30] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[31] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR (Poster)*, 2017.

[32] J. Klicpera, J. Groß, and S. Günnemann. Directional message passing for molecular graphs. In *ICLR*, 2020.

[33] Q. Li, Z. Han, and X. Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*, pages 3538–3545. AAAI Press, 2018.

[34] A. Longa, V. Lachi, G. Santin, M. Bianchini, B. Lepri, P. Lio, F. Scarselli, and A. Passerini. Graph Neural Networks for temporal graphs: State of the art, open challenges, and opportunities, July 2023. URL http://arxiv.org/abs/2302.01018. arXiv:2302.01018 [cs].

[35] A. Loukas. How hard is to distinguish graphs with graph neural networks? In *NeurIPS*, 2020.

[36] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI*, pages 4602–4609. AAAI Press, 2019.

[37] C. Morris, Y. Lipman, H. Maron, B. Rieck, N. M. Kriege, M. Grohe, M. Fey, and K. M. Borgwardt. Weisfeiler and leman go machine learning: The story so far. *CoRR*, abs/2112.09992, 2021.

[38] C. Morris, F. Frasca, N. Dym, H. Maron, İ. İ. Ceylan, R. Levie, D. Lim, M. M. Bronstein, M. Grohe, and S. Jegelka. Position: Future directions in the theory of graph machine learning. In *ICML*, 2024.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[40] P. Pellizzoni, T. H. Schulz, and K. M. Borgwardt. Graph neural networks can (often) count substructures. In *ICLR*, 2025.

[41] J. Topping, F. D. Giovanni, B. P. Chamberlain, X. Dong, and M. M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *ICLR*, 2022.

[42] P. Velickovic. Message passing all the way up. *CoRR*, abs/2202.11097, 2022.

[43] Q. Wang, D. Z. Chen, A. Wijesinghe, S. Li, and M. Farhan. $\mathscr{N}$-WL: A New Hierarchy of Expressivity for Graph Neural Networks. September 2022. URL https://openreview.net/forum?id=5cAI0qXxyv.

[44] X. Wang and M. Zhang. How powerful are spectral graph neural networks. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 23341–23362. PMLR, 2022.

[45] B. Weisfeiler and A. Leman. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia*, 2(9):12–16, 1968.

[46] T. Wollschläger, N. Kemper, L. Hetzel, J. Sommer, and S. Günnemann. Expressivity and generalization: Fragment-biases for molecular gnns. In *ICML*, 2024.

[47] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka. Representation learning on graphs with jumping knowledge networks. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 5449–5458. PMLR, 2018.

[48] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[49] K. Xu, J. Li, M. Zhang, S. S. Du, K. Kawarabayashi, and S. Jegelka. What can neural networks reason about? In *ICLR*, 2020.

[50] T. Xu. Towards a complete logical framework for GNN expressiveness. In *ICLR*, 2025.

[51] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. Salakhutdinov, and A. J. Smola. Deep sets. In *NIPS*, pages 3391–3401, 2017.

[52] B. Zhang, S. Luo, L. Wang, and D. He. Rethinking the expressive power of gnns via graph biconnectivity. In *ICLR*, 2023.

[53] B. Zhang, J. Gai, Y. Du, Q. Ye, D. He, and L. Wang. Beyond weisfeiler-lehman: A quantitative framework for GNN expressiveness. In *ICLR*, 2024.

[54] M. Zopf. 1-wl expressiveness is (almost) all you need. In *IJCNN*, pages 1–8. IEEE, 2022.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Our abstract and introduction accurately reflect our three main claims: 1) traditional expressivity theory has limitations that prevent it from explaining real-world GNN performance, 2) our proposed MPC captures practical limitations of MPNNs while preserving theoretical impossibility results, and 3) MPC correlates with empirical performance where expressivity theory fails to differentiate. In § 3, we thoroughly analyze limitations of expressivity theory. § 4.2 establishes formal connections between MPC and practical phenomena like over-squashing and under-reaching, while proving MPC preserves expressivity theory's impossibility results. § 5 empirically validates that MPC complexity correlates with MPNN performance across multiple tasks.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We provide a dedicated section that discusses key limitations (§ 6), including our abstraction from model parameterization, limited support for attention-based models, challenges with exact complexity computation, and the fact that complexity values should only be compared across a single dimension of freedom.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: All theoretical results include the full set of assumptions and are properly numbered and cross-referenced. For Lemmas 4.10 and 5.3 we provide only an informal version in the main body and defer a formal version to App. C. All proofs of theorems can be found in App. C.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide comprehensive experimental details in App. D.1, including model architectures, hyperparameters, and dataset specifications that allow reproduction of all experiments. An algorithm for simulating MPC complexity values is defined with pseudocode in Algorithm 1. Additionally, we provide our code at `https://www.cs.cit.tum.de/daml/message-passing-complexity/`.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: We provide our code at https://www.cs.cit.tum.de/daml/message-passing-complexity/.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: All experimental details — including the type of optimizer, the model hyperparameters and how they were chosen — are provided in App. D.1. Key experimental parameters are also detailed in the main paper (e.g., number of layers L for retaining information task). For our experiments on benchmark datasets, we use the standard pre-defined splits.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: All performance results are averaged over 3 random seeds. For all lineplots, we use Seaborn's built-in functionality which displays shaded regions representing the 95% confidence interval around the mean.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: All compute resources used for every experiment are detailed in App. D.1.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conforms in every respect with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [NA]

    Justification: Our work is foundational theoretical research on graph neural network expressivity and does not have direct societal impacts. We introduce a complexity measure that helps understand GNN limitations but does not create new capabilities that could be misused.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our work introduces a theoretical complexity measure for GNNs and uses only synthetic graphs and public benchmarks (ZINC, LRGB) that pose no risk for misuse. We release no models or datasets that require safeguards.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We properly cite all datasets (ZINC, LRGB) and software libraries (PyTorch Geometric) used in our work. We include their licenses in our experimental details section (App. D.1).

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: LLMs were not part of the core method development of this work.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Extended Related Work

Existing efforts to address the limitations of iso expressivity theory typically target only one limitation, focusing on either overcoming its task-agnostic nature, the unrealistic assumption of lossless information propagation, or its binary characterization of expressivity.

Task-Specific expressivity While iso expressivity theory is task-agnostic, recent works have proposed studying specific graph-related tasks to get a more fine-grained hierarchy of the expressivity of higher-order MPNNs (beyond the WL test). For example, expressivity has been analyzed in the context of substructure recognition [53] and graph biconnectivity [52].

A separate line of work characterizes the node-level functions computable by MPNNs in terms of fragments of first-order logic [23]: Barceló et al. [3] showed that standard MPNNs with sum aggregation and ReLU activations can uniformly express graded modal logic. Subsequent work analyzed the effect of different activation functions [29], and more recently, these logical characterizations were extended to various higher-order MPNNs [50].

However, all these works only provide *existence* results: they show whether there exists some model instance of an architecture that can solve a task or not. Such results neglect practical limitations like information loss (seem in phenomenons like over-squashing or over-smoothing) occurring in *trained real-world models*. Moreover, the characterizations are inherently binary (solvable or not), offering limited insight into the practical difficulty.

Accounting for Lossy Information Propagation. Some works explicitly account for the lossy information propagation inherent in practical GNNs. Negative results have been derived by considering limited hidden dimensions [35] or by analyzing the maximal "mixing" of node representations [20]. However, these approaches are restricted to specific model architectures and provide binary results, rather than a nuanced understanding of how lossy propagation impacts practical performance in arbitrary tasks.

Beyond Binary Expressivity. To address the binary nature of classical expressivity theory, some works define distances between graphs to represent the difficulty of distinguishing them. Examples include tree-based distances [12], graphon-based distances [7], and Wasserstein-based distances [10]. While these metrics provide a continuous measure, they remain task-agnostic and are independent of model architecture.

Restricted Graph Families. Classical expressivity theory evaluates GNNs on the set of all possible graphs, which may not align with the restricted graph families encountered in real-world applications. Recent works examine expressivity within specific graph families, such as planar graphs [13] and outer-planar graphs [4], providing more relevant insights for certain domains.

Higher-order MPNNs and variants of the WL Test. A plethora of higher-order MPNNs [5, 6, 46, 36] have been developed to surpass the WL test in iso expressivity. To quantify and compare their expressivity corresponding (higher-order) variants of the WL test [36, 5, 46, 43] have been developed. However, we show in Tab. 1 similar to Zopf [54] that the standard WL test already suffices to distinguish almost all graphs in standard benchmarks. Hence, expressivity theory statements that only focus on expressivity beyond the WL test cannot offer explanations for performance differences of models that are at least WL expressive on these popular benchmarks.

Many of these higher-order methods, along with an overview of WL variants, are comprehensively surveyed in Morris et al. [37].

Table 1: Fraction of graphs with unique WL hashes (ignoring isomorphic graphs). Similar to Zopf [54].

Dataset	**Unique WL**
Reddit-Binary	100%
Peptides	100%
Mutag	100%
Enzymes	100%
Protein-dataset	100%
ZINC-subset	100%
ZINC-full	$> 99.99\%$

Real world limitations of GNNs Several fundamental limitations of GNNs have been identified in practical applications. Under-reaching [3] occurs when nodes cannot access information from distant parts of the graph due to insufficient layer depth, effectively limiting the receptive field of each node. Over-squashing [1, 41, 19] describes how graph topology can create bottlenecks in message passing, preventing effective information flow between nodes even when they are theoretically within each other's receptive field. This is distinct from over-smoothing [33], where increasing the number of layers causes node representations to become indistinguishable as all nodes converge to the same representation. These limitations highlight a crucial gap between theoretical expressivity and practical capabilities of GNNs.

Practical power of standard GNNs A few other works highlight a disconnect between theoretical expressivity limitations and the practical capabilities of standard GNNs. While expressivity theory establishes that standard GNNs cannot count any non-trivial substructure [11], recent studies reveal important exceptions. Pellizzoni et al. [40] identify specific conditions on the graph family $\mathcal{G}$ under which standard GNNs can count substructures and demonstrate that many real-world datasets satisfy these conditions for important substructures like small cycles. Similarly, Kanatsoulis and Ribeiro [28] showed that adding randomized node features enables standard GNNs to count small cycles.

Task-Model Alignment. Beyond expressivity, our work aligns with efforts to quantify the compatibility between a model and its target task. For instance, Xu et al. [49] define on a higher level the concept of algorithmic alignment between a general neural network and an algorithm, showing that higher alignment leads to lower sample complexity. Similarly, our proposed complexity measure can be interpreted as quantifying the alignment between a specific graph task and the message passing steps needed to solve it for a given MPNN architecture.

In summary, while existing works address individual limitations of classical expressivity theory, none provide a unified framework addressing all limitations.

B Extending MPC: More Architectures, Graph-Level Functions, and Feature Noise

In the following, we show how MPC can be extended to a wide range of architectures beyond standard MPNNs and to graph-level functions. Additionally, we sketch a potential extension to include additional sources of complexity, such as feature noise. Furthermore, we sketch how one could easily adapt the framework to operate on temporal graphs.

B.1 General Message Passing Framework

In the following, we provide a general message passing framework for MPC that captures a wide range of existing MPNN architectures $\mathcal{M}$.

While standard MPNNs $\mathcal{M}_S$ perform message passing directly on the input graph G, other architectures operate on a transformed message passing graph $\tilde{G} = t(G)$:

$$m^l_{w\to v} := \begin{cases} \mathtt{msg}^l_0(h^{l-1}_w) & \text{if } w = v \\ \mathtt{msg}^l_1(h^{l-1}_w) & \text{else} \end{cases} \quad \text{and} \quad h^l_v := \mathtt{upd}^l(\mathtt{agg}\left(\{\!\!\{m^l_{w\to v} \mid w \in N_{\tilde{G}}(v) \cup \{v\}\}\!\!\}\right))$$

The transformation t defines the MP graph structure $\tilde{G}$ by determining which nodes exchange messages. While standard MPNNs used the input graph G directly as $\tilde{G}$, recent architectures introduce modifications like virtual nodes [18], rewired edges [41], or higher-order graphs [46]. We assume t preserves original nodes $\mathcal{V}_G$ while potentially adding nodes or adding/modifying edges. A general MPNN architecture $\mathcal{M}$ is defined through the transformation t and the number of layers L, characterizing which nodes exchange information. It abstracts away from the specific choice of aggregation method and update function, which is a sensible simplification since many recent models treat the choice of aggregation method as a hyperparameter and use MLPs as update functions (providing universal function approximation capabilities).

This framework captures many existing MPNNs (potentially with some simplifying assumptions):

Standard MPNNs Standard architectures like GCN [31], GraphSage [25], GIN [48] perform message passing directly on the input graph, i.e., $\tilde{G} = t(G) = G$. They differ in aggregation method (mean, sum, degree-normalized mean). Some architectures, like GCN, restrict the possible choices of msg and upd function and do not distinguish self-loops from normal edges, i.e., $\mathtt{msg}_0 = \mathtt{msg}_1$. Note that, as discussed above, our MPNN framework abstracts away these architectural choices.

Virtual nodes, rewiring and additional features Modifications to the input graph such as additional virtual nodes [18], edge rewiring [41], or additional node features (e.g., including substructure information like in GSN [8]) can be directly modeled in our framework by choosing the appropriate transformation function t.

Higher-order MPNNs Higher-order MPNNs that augment the input graph with higher-level structures are also representable. FragNet [46] builds an additional higher-level graph of fragments (for an arbitrary fragmentation scheme). We will focus here on the FragNet model with a fragmentation scheme identifying rings without edge representations. We model this by adding additional fragment nodes connected to all constituent input nodes, with labels corresponding to fragment types. The CIN [5] model follows a similar approach by having representations for all original nodes, edges, and CW-cells (rings). Upper messages are messages from nodes to their corresponding edge representation, and from edges to the corresponding ring representation (if any). Boundary messages are messages between nodes sharing an edge and between edges that are both in the same ring. The initial representation of CW cells is an aggregation of the features of their constituent nodes. We represent edges and CW cells (rings) by additional nodes in $\tilde{G}$. The initial feature of a ring node is the aggregation of all node features of the ring. We have edges between: neighboring original nodes, original nodes and their edge nodes, edge nodes within the same ring, and edge nodes and their rings (if any). Note that we cannot represent in our framework that boundary messages also contain information from the corresponding upper neighborhood. Importantly, both fragNet and CIN use different update functions for different types of messages. This too can not be modeled in our formulation of $\mathcal{M}$; instead, we assume just one update function. While one could, in theory, extend our framework to include different kinds of message updates, we choose against this as our existing framework, with this simplification, already captures the models' empirical performance well.

3D GNNs: DimeNet Next, we sketch how our framework can model GNNs incorporating 3D information, exemplified by DimeNet [32]. This architecture maintains representations for all directed edges and incorporates angular information. In our framework, this can be captured by using the directed line graph $\tilde{G} = (\tilde{V}, \tilde{E})$ of the original graph G as the message passing graph where $\tilde{V} = E$ (the original edges become nodes) and for any $e_1 = (v_j, v_k), e_2 = (v_k, v_i) \in E$, we have $(e_1, e_2) \in \tilde{E}$. The angular and distance features of DimeNet can then be incorporated as edge/node features in this transformed message passing graph.

Given this general message passing framework, we can naturally extend MPC from standard MPNNs $\mathcal{M}_S$ to general MPNNs $\mathcal{M}$ by applying lossyWL to the transformed message passing graph $\tilde{G} = t(G)$ rather than the input graph:

Definition B.1 (MPC)**.** For $G, v \in \mathcal{G}_\mathcal{V}$, a function f over $\mathcal{G}_\mathcal{V}$ and a L-layer MPNN $\mathcal{M}$, define

$$\mathrm{MPC}_\mathcal{M}(f_v, G) = -\log \mathbb{P}[\mathrm{lossyWL}_v^L(t(G)) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)].$$

Analogously, we can extend our baseline WLC to general MPNNs $\mathcal{M}$:

Definition B.2. For $G, v \in \mathcal{G}_\mathcal{V}$, a function f over $\mathcal{G}_\mathcal{V}$ and a L-layer MPNN $\mathcal{M}$, define

$$\mathrm{WLC}_\mathcal{M}(f_v, G) = \begin{cases} \infty & \text{if } \mathrm{WL}_v^L(t(G)) \nvDash_{\mathcal{G}_\mathcal{V}} f_v(G) \\ 0 & \text{else} \end{cases}$$

B.2 Graph-Level Tasks

Additionally, we can generalize MPC to graph-level tasks $f : \mathcal{G} \to \mathbb{R}^k$. A graph-level output of an MPNN architecture $\mathcal{M}$ is learned by aggregating all final node representations h_v^L and transforming the aggregated result. This can be represented in our MPNN framework by having an additional readout node v_{global} that receives messages from all other nodes only in the $L+1$-th layer. MPC is then defined over the graph, node pairs $\{(G, v_{\text{global}}) \mid G \in \mathcal{G}\}$.

B.3 Additional Sources of Complexity: Feature Noise

While MPC is designed to isolate the complexity arising from the message passing topology, it can also be extended to include additional sources of complexity arising in practice, such as feature noise, thereby providing a more unified practical understanding of task-difficulty. We sketch here such a possible extension:

Let $\boldsymbol{X}_v$ be the true feature of node v, and let $\tilde{\boldsymbol{X}}_v$ be the observed noisy feature according to some noise distribution $\tilde{\boldsymbol{X}}_v \sim N_v(\boldsymbol{X}_v)$. Our probabilistic formulation naturally accommodates this additional source of difficulty by modifying the initialization step in Definition 4.3 to incorporate noise:

$$\text{lossyWL}_v^0 = \tilde{\boldsymbol{X}}_v \quad \text{with} \quad \tilde{\boldsymbol{X}}_v \sim N_v(\boldsymbol{X}_v).$$

Notice that this general model of feature noise subsumes missing features as a special case. All in all, this demonstrates a promising direction for incorporating additional practically relevant sources of complexity within a unified framework in future work.

B.4 Temporal Graphs

MPC can be naturally extended to temporal graphs as well. Consider snapshot graphs G^τ at time point τ, as defined in Longa et al. [34]. The complexity for a task f_v then becomes time dependent:

$$\text{MPC}^\tau_{\mathcal{M}}(f_v, G) := -\log \mathbb{P}[\text{lossyWL}_v^L(t(G^\tau)) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)].$$

This extension opens up interesting directions for future work, such as analyzing how the complexity of specific tasks evolves over time as the underlying graph changes.

C Proofs and Extended Theorems

In this section, we present all proofs and extended/formal versions of the theorems in the main body.

Extended Notation Let $\circ$ denote function composition. Additionally, let α^{-1} be the preimage of a function $\alpha : X \to Y$, i.e., $\alpha^{-1}(y) = \{x \in X \mid \alpha(x) = y\}$. If X is clear from the context, we will simply write $\alpha \vDash \beta$ instead of $\alpha \vDash_X \beta$. We define $\text{WL}_v := \text{WL}_v^L$ and $\text{lossyWL}_v := \text{lossyWL}_v^L$, i.e., we omit the layer superscript for the final layer L. Additionally, we say a message $m^l_{u\to v}$ in the lossyWL test *is lost*, if $Z^l_{uv} = 0$, otherwise we say it was successful. Let $M_v(G)$ denote the final output h_v^L of a model at node v. Additionally, we will say a set $\mathcal{S} \subseteq 2^U$ of sets over an universe U is *upward-closed* if $A \in \mathcal{S}$ implies that any $B \subseteq U$ with $A \subseteq B$ is also in $\mathcal{S}$, i.e., $B \in \mathcal{S}$. $\mathbf{1}[\text{condition}]$ denotes the indicator function defined as 1 if the condition is true and 0 else. We use the terms "ring" and "cycle" interchangeably.

We will often represent a probabilistic function τ with domain X and finitely many possible probabilistic outputs in Y (each with rational probability) as a deterministic function $\tau(x, s)$ or $\tau_s(x)$ with a seed drawn uniformly at random from a finite set of seeds S. We will sometimes (slightly abuse notation and) write $\tau(x, s) \vDash_{X\times S} \beta(x)$ for a non-probabilistic function $\beta : X \to Z$ if

$$\forall x' \in X. \forall s' \in S : \tau(x, s) = \tau(x', s') \implies \beta(x) = \beta(x'),$$

i.e., if one can deduce $\beta(x)$ from the output of $\tau(x, s)$. With this notation, we can nicely write Definition 4.4 as

$$\mathbb{P}[\tau(x) \vDash_X \beta(x)] := \mathbb{P}_{s\in S}[\tau(x, s) \vDash_{X\times S} \beta(x)].$$

Additionally, we will define the concept of necessary and sufficient messages that we will use in several proofs.

Definition C.1. We call a set $\mathbb{S}$ of messages (or rather message identifiers (a, b, l)) sufficient for a function f_v on a graph G, if it is possible to deduce f_v from lossyWL_v given that the messages in $\mathbb{S}$ were successful, or more formally:

$$\bigwedge_{(a,b,l)\in\mathbb{S}} Z^l_{ab} = 1 \implies \text{lossyWL}_v(t(G)) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

Definition C.2. We call a set $\mathbb{S}$ of messages necessary for a function f_v on a graph G, if it not possible to deduce f_v from lossyWL_v if any of the messages in $\mathbb{S}$ were lost, or more formally:

$$\exists (a, b, l) \in \mathbb{S} : Z^l_{ab} = 0 \implies \text{lossyWL}_v(t(G)) \nvDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

Lastly, we define the probability of a set $\mathbb{S}$ of messages as the probability that all messages were successful:

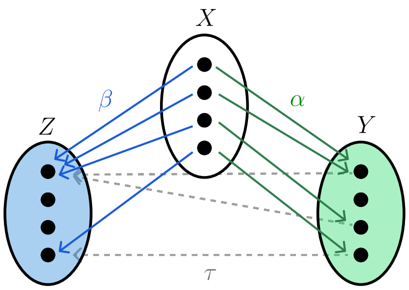


Figure 7: Example for Definition 4.1 showing sets X, Y, Z and functions α, β. We have $\alpha \vDash \beta$ as $\alpha(x) \vDash \beta(x)$ for all $x \in X$. By Lemma C.4, there exists a function τ with $\beta = \tau \circ \alpha$. Contrarily, $\beta \nvDash \alpha$ as there exist $x_1, x_2 \in X$ with $\beta(x_1) = \beta(x_2)$ but $\alpha(x_1) \neq \alpha(x_2)$. Intuitively, this means that function α is more fine-grained than β.

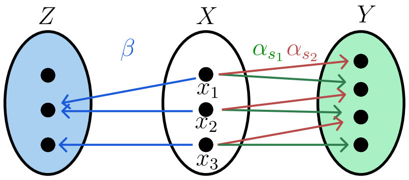


Figure 8: Example for Definition 4.4 showing sets X, Y, Z and a probabilistic function α and a deterministic function β. The probabilistic function α can be represented by two deterministic functions $\alpha_{s_1}, \alpha_{s_2}$ where the seed is drawn uniformly at random. We then have $\mathbb{P}[\alpha(x_1) \vDash \beta(x_1)] = 1$. Intuitively, we can deduce $\beta(x_1)$ for every probabilistic outcome of $\alpha(x_1)$. Whereas for x_2, x_3, we have $\mathbb{P}[\alpha(x_2) \vDash \beta(x_2)] = \mathbb{P}[\alpha(x_3) \vDash \beta(x_3)] = 0.5$.

Definition C.3. Define the probability of a set of messages as

$$\mathbb{P}[\mathbb{S}] := \prod_{(a,b,l) \in \mathbb{S}} \mathbb{P}[Z^l_{ab} = 1].$$

To simplify notation, we will prove all statements in Apps. C.2 to C.5 for $\mathcal{M}_S$. The proofs for general model architectures $\mathcal{M}$ follow completely analogously by performing lossyWL on $\tilde{G} = t(G)$ instead of G, i.e., by replacing G with $t(G)$ and $\mathcal{M}_S$ with $\mathcal{M}$.

C.1 Function Refinement

First, we will now prove the following useful lemma (see also Fig. 7).

Lemma C.4 (Refinement)**.** *If, and only if, $\alpha \vDash \beta$, there exists a function τ such that $\beta = \tau \circ \alpha$.*

Proof. First, we will show that $\alpha \vDash \beta$ implies that there exists a function τ with $\beta = \tau \circ \alpha$. Define $\tau : \alpha(X) \to Z$ as

$$\tau(y) := \beta(x) \text{ for } x \in \alpha^{-1}(y)$$

This function is well-defined because by definition of $\vDash$, we have for all $x_1, x_2 \in \alpha^{-1}(y)$: $\beta(x_1) = \beta(x_2)$. And it follows immediately that $\beta = \tau \circ \alpha$.

Next, we will show that $\alpha \nvDash \beta$ implies that there does not exist such a function τ with $\beta = \tau \circ \alpha$. Assume for the sake of contradiction that τ exists. From $\alpha \nvDash \beta$, it follows that there exist x_1, x_2 with $\alpha(x_1) = \alpha(x_2)$ but $\beta(x_1) \neq \beta(x_2)$. Then

$$\tau(\alpha(x_1)) = \tau(\alpha(x_2)) = \beta(x_2) \neq \beta(x_1) = \tau(\alpha(x_1)).$$

Hence, $\tau(\alpha(x_1)) \neq \tau(\alpha(x_1))$ which is a contradiction. □

C.2 Connection of standard MPNNs to WL

Next, we will show that there exists a model in our standard MPNN framework that is at least as expressive as the WL test. In fact, there exists a model performing exactly the WL test:

Lemma C.5. *There exists a model $M \in \mathcal{M}_S$ that performs exactly the WL test, i.e., for all $G, v \in \mathcal{G}_\mathcal{V}$ $M_v(G) = \mathrm{WL}_v(G)$.*

Proof. We show this inductively over the number of layers L. For $L = 0$ this follows immediately from $\mathrm{WL}_v^0(G) = \boldsymbol{X}_v$ and $M_v^0(G) = \boldsymbol{X}_v$ for any 0-layer model $M^0 \in \mathcal{M}$. For $L = l > 0$, assume that there exists a model M^{l-1} with $M_v^{l-1} = \mathrm{WL}_v^{l-1}$. Then consider a model M^l that executes M^{l-1} in the first $l-1$ layers:

$$m_{w\to v}^l := \begin{cases} \mathtt{msg}_0^l(\mathrm{WL}_w^{l-1}) & \text{if } w = v \\ \mathtt{msg}_1^l(\mathrm{WL}_w^{l-1}) & \text{else} \end{cases} \quad \text{and} \quad h_v^l := \mathtt{upd}^l(\mathtt{agg}\left(\{\!\{ m_{w\to v}^l \mid w \in N_G(v) \cup \{v\} \}\!\}\right))$$

We now need to show that there exist functions $\mathtt{msg}_0^l, \mathtt{msg}_1^l$, $\mathtt{agg}$ and $\mathtt{upd}$ such that for all $G, v \in \mathcal{G}$

$$h_v^l = \mathrm{WL}_v^l = \mathrm{HASH}\left(\mathrm{WL}_v^{l-1}, \{\!\{ \mathrm{WL}_u^{l-1} \mid u \in N_G(v) \}\!\}\right).$$

Zaheer et al. [51] show that one can represent any (permutation-invariant) function $\alpha(X)$ operating on a set X as $\beta\left(\sum_{x\in X} \phi(x)\right)$. As we can differentiate between $\mathtt{msg}_1$, and $\mathtt{msg}_0$ it is easy to see that there also exists functions $\mathtt{msg}_0^l, \mathtt{msg}_1^l$ and $\mathtt{upd}$ such that with sum-aggregation, we have:

$$\begin{aligned} \mathrm{WL}_v^l(G) &= \mathrm{HASH}\left(\mathrm{WL}_v^{l-1}, \{\!\{ \mathrm{WL}_u^{l-1} \mid u \in N_G(v) \}\!\}\right) \\ &= \mathtt{upd}^l\left(\sum_{w \in N_G(v)\cup\{v\}} \mathtt{msg}_{\mathbb{1}[w=v]}^l(\mathrm{WL}_w^{l-1})\right) \\ &= \mathtt{upd}^l\left(\sum_{w \in N_G(v)\cup\{v\}} m_{w\to v}^l\right) \\ &= h_v^l \\ &= M_v^l(G) \end{aligned}$$ □

C.3 When MPC becomes infinite: Connections to iso expressivity theory

We will now prove important theoretical properties of MPC. For this, we will first prove the following useful lemma showing that MPC is infinite precisely for the same tasks and graphs where WLC is infinite:

Lemma C.6. *We have:*

$$\mathbb{P}[\mathrm{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] > 0$$

if and only if

$$\mathrm{WL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

Proof. We will say a message $m_{a\to b}^l$ is visible to $\mathrm{lossyWL}_v$ if there exists a $L - l$ length walk from b to v.

We will first assume that $\mathrm{WL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)$. The proof idea for this direction is that with positive probability lossyWL loses no to v visible message and performs exactly the normal WL test. This coloring will always be different from lossyWL colors with visible lost messages and can, hence, by assumption, be used to deduce f_v.

Formally, notice that with probability > 0 $\mathrm{lossyWL}_v^L$ performs exactly the normal Weisfeiler-Lehman test at node v, i.e., no to $\mathrm{lossyWL}_v$ visible messages are lost ($Z_{uv}^l = 1$ for all visible messages). Now consider such a seed $s_1 \in S$ for which no visible messages are lost. Then for all seeds $s_2 \in S$ and $G_2, w \in \mathcal{G}_\mathcal{V}$ for which messages visible to $\mathrm{lossyWL}_w(G_2)$ are lost, we have

$$\mathrm{lossyWL}_v(G, s_1) \neq \mathrm{lossyWL}_w(G_2, s_2)$$

because of the injectiveness of the HASH function, and only $\text{lossyWL}_w(t(G), s_2)$ has visible lost messages $m = 0$. Hence, we have for any $s_2 \in S$ and $G_2, w \in \mathcal{G}_\mathcal{V}$

$$\text{lossyWL}_v(G, s_1) = \text{lossyWL}_w(G_2, s_2) \implies \text{WL}_v(G) = \text{WL}_w(G_2)$$

as $\text{lossyWL}_v(G, s_1) = \text{lossyWL}_w(G_2, s_2)$ implies that s_2 loses no to w visible messages, i.e., performs exactly the WL test at w as well. Therefore,

$$\text{lossyWL}_v(G, s_1) \vDash_{\mathcal{G}_\mathcal{V} \times S} \text{WL}_v(G).$$

And from the assumption and the transitivity of $\vDash$, it follows that

$$\text{lossyWL}_v(G, s_1) \vDash \text{WL}_v(G) \vDash f_v(G),$$

yielding

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] > 0$$

Now we show that from $\text{WL}_v(G) \nvDash_{\mathcal{G}_\mathcal{V}} f_v(G)$, it follows that $\mathbb{P}[\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] = 0$, i.e., losing messages cannot make the probabilistic WL test more expressive than the deterministic WL test. $\text{WL}_v(G) \nvDash f_v(G)$ implies there is $G', w \in \mathcal{G}_\mathcal{V}$ with $\text{WL}_v(G) = \text{WL}_w(G')$ but $f_v(G) \neq f_w(G')$. Then, for every seed $s_1 \in S$, there exists a seed $s_2 \in S$ such that

$$\text{lossyWL}_v(G, s_1) = \text{lossyWL}_w(G', s_2).$$

(the same messages to WL equivalent nodes in G and G' are lost). Hence by Definition 4.4,

$$\mathbb{P}(\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)) = 0. \qquad \square$$

Using this, we can prove when MPC becomes infinite:

Theorem 4.6 (Infeasibility)**.** *The complexity for $G, v \in \mathcal{G}_\mathcal{V}$ and function f is $\text{MPC}_\mathcal{M}(f_v, G) = \infty$ if and only if there exist $G', w \in \mathcal{G}_\mathcal{V}$ such that $f_v(G) \neq f_w(G')$ but $M_v(G) = M_w(G')$ for all model instantiations $M \in \mathcal{M}$.*

Proof. Assume first that there exists $G, v \in \mathcal{G}_\mathcal{V}$ with $\text{MPC}_{\mathcal{M}_S}(f, G) = \infty$. By definition, this implies that

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] = 0.$$

From Lemma C.6 it follows that

$$\text{WL}_v(G) \nvDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

By definition of $\vDash$ this implies that there exists $G', w \in \mathcal{G}_\mathcal{V}$ with $\text{WL}_v(G) = \text{WL}_w(G')$ but $f_v(G) \neq f_w(G')$. Now it follows from the fact that the WL test upper-bounds the expressivity of standard MPNNs $\mathcal{M}_S$ [36] that also all models $M \in \mathcal{M}$ cannot differentiate G, v from G', w, i.e., $M_v(G) = M_w(G')$.

Now, assume that $\text{MPC}_{\mathcal{M}_S}(f, G) \neq \infty$. By definition, this implies that

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] > 0.$$

From Lemma C.6 it follows that

$$\text{WL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

By definition of $\vDash$ this implies that there *do not* exist $G', w \in \mathcal{G}_\mathcal{V}$ with $\text{WL}_v(G) = \text{WL}_w(G')$ but $f_v(G) \neq f_w(G')$. Now note that by Lemma C.5 there exists a model $M \in \mathcal{M}_S$ performing exactly the WL test. Hence, there exists $M \in \mathcal{M}_S$ such that there does not exist $G', w \in \mathcal{G}_\mathcal{V}$ with $M_v(G) = M_w(G')$ but $f_v(G) \neq f_w(G')$. □

With this, we can also globally characterize which functions an MPNN can express:

Lemma 4.7. *There exists no model instantiation $M \in \mathcal{M}$ such that $M_v(G) = f_v(G)$ for all $G, v \in \mathcal{G}_\mathcal{V}$ if and only if there exists $G, v \in \mathcal{G}_\mathcal{V}$ with $\text{MPC}_\mathcal{M}(f_v, G) = \infty$.*

Proof. If there exists $M \in \mathcal{M}$ such that $M_v(G) = f_v(G)$ for all $G, v \in \mathcal{G}_\mathcal{V}$, it follows directly from Thm. 4.6 that there does not exist $G, v \in \mathcal{G}_\mathcal{V}$ with $\text{MPC}_{\mathcal{M}_S}(f_v, G) = \infty$.

If there does not exist $G, v \in \mathcal{G}_\mathcal{V}$ with $\text{MPC}_{\mathcal{M}_S}(f_v, G) = \infty$, this implies that for all $G, v \in \mathcal{G}_\mathcal{V}$

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] > 0.$$

Hence, by Lemma C.6 for all $G, v \in \mathcal{G}_\mathcal{V}$

$$\mathrm{WL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G).$$

So

$$\mathrm{WL} \vDash_{\mathcal{G}_\mathcal{V}} f.$$

By Lemma C.5 there exists $M \in \mathcal{M}_S$ with $M \vDash_{\mathcal{G}_\mathcal{V}} f$. And by Lemma C.4 a function τ such that for all $G, v \in \mathcal{G}_\mathcal{V}$

$$\tau \circ M_v(G) = f_v(G).$$

But then we can squash the function τ also in the last update layer of M, i.e., $\mathtt{upd}^l_{\text{new}} = \tau \circ \mathtt{upd}^l$. With this, we have found a model M^* such that

$$M^*_v(G) = f_v(G)$$

for all $G, v \in \mathcal{G}_\mathcal{V}$. □

As a special case, we can recover the iso expressivity statements defined in Definition 3.1. For this, we use the graph-level formulation of MPC. Note that all previous theorems generalize completely analogously also to this graph-level version.

Lemma C.7 (MPC preserves impossibility statement of Iso Expressivity)**.** *Let α be a graph isomorphism test. Then $\mathcal{M}$ is at least as expressive as α if and only if there does* not exist *$G \in \mathcal{G}^*$ with* $\mathrm{MPC}_\mathcal{M}(\alpha, G) = \infty$.

Proof. If $\mathrm{MPC}_\mathcal{M}(\alpha, G) = \infty$ for some $G \in \mathcal{G}^*$, then by Thm. 4.6, there exists $G' \in \mathcal{G}^*$ such that $M(G) = M(G')$ for all $M \in \mathcal{M}$ but $\alpha(G) \neq \alpha(G')$. From this follows directly that $\mathcal{M}$ is not at least as expressive as α.

Now assume that there does not exist $G \in \mathcal{G}^*$ with $\mathrm{MPC}_\mathcal{M}(\alpha, G) = \infty$. Then by Lemma 4.7 there exists $M \in \mathcal{M}$ with $M(G) = \alpha(G)$ for all $G \in \mathcal{G}^*$. Therefore, $\mathcal{M}$ is at least as expressive as α. □

C.4 Bounding MPC: Function Refinement & Compositionality

Next, we will prove bounds on MPC relating to how fine-grained a task is and how it can be decomposed into individual subtasks. First, we prove that a more fine-grained function cannot have lower complexity than the coarser function it refines:

Theorem 4.8 (Function refinement)**.** *Let f be a function that is more fine-grained than g, i.e., $f \vDash_{\mathcal{G}_\mathcal{V}} g$. Then for any $G, v \in \mathcal{G}_\mathcal{V}$:* $\mathrm{MPC}_\mathcal{M}(f_v, G) \geq \mathrm{MPC}_\mathcal{M}(g_v, G)$.

Proof. The intuitive idea for this proof is that if the set of successful messages in lossyWL suffices to deduce f, it is also always possible to deduce the more coarse-grained g.

First, note that $\vDash$ is transitive. Then, it follows directly that for every seed $s \in S$ for which $\mathrm{lossyWL}_v(G, s) \vDash_{\mathcal{G}_\mathcal{V} \times S} f_v$ also $\mathrm{lossyWL}_v(G, s) \vDash_{\mathcal{G}_\mathcal{V} \times S} g_v$. Therefore

$$\mathbb{P}[\mathrm{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} f_v(G)] \leq \mathbb{P}[\mathrm{lossyWL}_v(G) \vDash_{\mathcal{G}_\mathcal{V}} g_v(G)].$$

And hence,

$$\mathrm{MPC}_\mathcal{M}(f_v, G) \geq \mathrm{MPC}_\mathcal{M}(g_v, G).$$

□

Now, we will prove the compositional property of MPC.

Lemma 4.9 (Task Triangle Inequality)**.** *Let f and g be functions, and $\|$ denote concatenating function outputs. Then for any $G, v \in \mathcal{G}_\mathcal{V}$:* $\mathrm{MPC}_\mathcal{M}(f_v \| g_v, G) \leq \mathrm{MPC}_\mathcal{M}(f_v, G) + \mathrm{MPC}_\mathcal{M}(g_v, G)$.

Proof. Let L_f be a sufficient set of messages for f_v on graph G. Let $\mathbb{L}_f$ denote the set of all such sufficient sets L_f. Notice that $\mathbb{L}_f$ is upward-closed, i.e., any superset of a $L_f \in \mathbb{L}_f$ is also sufficient and therefore in $\mathbb{L}_f$. This intuitively means that more successful messages cannot hurt. Define analogously $\mathbb{L}_g$ for the task g.

For a set of messages S define $\alpha_f(S) = \mathbf{1}[S \in \mathbb{L}_f]$. Then, α_f is an increasing function over sets, i.e., $S \subseteq S' \implies \alpha_f(S) \leq \alpha_f(S')$, because $\mathbb{L}_f$ is upward-closed. From the Fortuin–Kasteleyn–Ginibre

(FKG) inequality, we then have (where S follows the distributions of the sets of successful messages in lossyWL):

$$\mathbb{E}[\alpha_f(S)\alpha_g(S)] \geq \mathbb{E}[\alpha_f(S)] \cdot \mathbb{E}[\alpha_g(S)]$$

implying

$$\begin{aligned}&\mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G) \wedge \text{lossyWL}_v(G) \vDash g_v(G)]\\ &\quad \geq \mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G)] \cdot \mathbb{P}[\text{lossyWL}_v(G) \vDash g_v(G)]\end{aligned}$$

and thereby

$$\text{MPC}_{\mathcal{M}}(f_v \| g_v, G) \leq \text{MPC}_{\mathcal{M}}(f_v, G) + \text{MPC}_{\mathcal{M}}(g_v, G). \qquad \square$$

C.5 Connection of MPC to Over-squashing and Under-reaching

To formalize Lemma 4.10, we first need to define when a task f_v requires information from another node u.

Definition C.8. We say a task f_v over graphs $\mathcal{G}$ requires (node-feature) information from a node u if there exists $G^1, G^2 \in \mathcal{G}$ that are identical except for the node features $\boldsymbol{X}_u^1 \neq \boldsymbol{X}_u^2$ and

$$f_v(G^1) \neq f_v(G^2).$$

So a task requires information from a node u, if without this information it would be impossible to compute f_v.

With this in place, we can give a formal version of Lemma 4.10:

Lemma C.10. *Consider a task f_v and a graph $G \in \mathcal{G}$ where f_v requires node-feature information from u. Then*

$$\text{MPC}_{\mathcal{M}}(f_v, G) \geq -\log\left((\boldsymbol{I}^L)_{vu}\right).$$

Proof. Intuitively, we will show that in order to deduce f_v from lossyWL$_v$ there needs to exist an L-length walk from u to v where all messages were successful (as f_v requires information from u). The probability of this can be in turn upperbounded by the random walk probability.

Formally, we need to show that

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G)] \leq (\boldsymbol{I}^L)_{vu}.$$

For this we will upper bound $\mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G)]$ by the L step random walk probability from v to u.

For this, let W_{ab} be the event that there exists $x_1, \dots x_{L-1} \in \mathcal{V}_G$ with

$$Z^1_{ax_1} = 1 \wedge Z^2_{x_1x_2} = 1 \wedge Z^3_{x_2x_3} = 1 \wedge \cdots \wedge Z^{L-1}_{x_{L-2}x_{L-1}} = 1 \wedge Z^L_{x_{L-1}b} = 1.$$

We then say Z contains a (L-length) walk from a to b. Additionally, let W^i_{ab} be the event that Z contains a specific walk from a to b with fully specified intermediate nodes x_j. Then the probability of W^i_{ab} is (using the independence of the variables Z):

$$\begin{aligned}\mathbb{P}[W^i_{ab}] &= \mathbb{P}[Z^1_{ax_1} = 1] \cdot \mathbb{P}[Z^2_{x_1x_2} = 1] \cdots \mathbb{P}[Z^L_{x_{L-1}b} = 1]\\ &= \boldsymbol{I}_{x_1a} \cdot \boldsymbol{I}_{x_2x_1} \cdots \boldsymbol{I}_{bx_{L-1}}\end{aligned}$$

This is precisely the probability for a random walk from b to a over all intermediate nodes x_i with edge probabilities $\boldsymbol{I}$. If we now consider all L-length walks $W^1_{ab}, \dots, W^k_{ab}$ from a to b, we get:

$$\begin{aligned}\mathbb{P}[W_{ab}] &\leq \mathbb{P}[W^1_{ab}] + \cdots + \mathbb{P}[W^k_{ab}]\\ &= \mathbb{P}[L\text{-step random walk from } b \text{ to } a \text{ with edge probabilities } \boldsymbol{I}]\\ &= (\boldsymbol{I}^L)_{ba}\end{aligned}$$

Now note that if **not** W_{uv}, i.e., Z contains no walk from u to v, it is by Definition C.8 not possible to deduce $f_v(G)$ from lossyWL$_v(G)$ as v will not receive any information from u.

Hence,

$$\begin{aligned}\mathbb{P}[\text{lossyWL}(G)_v \vDash f_v(G)] &\leq \mathbb{P}[W_{uv}]\\ &\leq (\boldsymbol{I}^L)_{vu}.\end{aligned} \qquad \square$$

With this we can easily prove Corollary 4.11:

Proof. If $d_G(u, v) > L$ there exists no L step walk from v to u on G. Therefore $(\boldsymbol{I}^L)_{vu} = 0$ and by definition of MPC:

$$\text{MPC}_{\mathcal{M}}(f_v, G) = \infty.$$

□

C.6 Complexity bounds in § 5

First, we will prove two useful lemmas to upper and lower bound the MPC complexities:

Lemma C.11. *Let $\mathbb{S}$ be a sufficient set of messages for a function f_v on a graph G. Then*

$$\text{MPC}_{\mathcal{M}}(f_v, G) \leq -\log(\mathbb{P}[\mathbb{S}]).$$

Proof. By Definition C.1 we have:

$$\begin{aligned}\mathbb{P}[\text{lossyWL}_v(t(G)) \vDash f_v(G)] &\geq \mathbb{P}[\bigwedge_{(a,b,l)\in\mathbb{S}} Z^l_{ab} = 1] \\ &= \mathbb{P}[\mathbb{S}]\end{aligned}$$

□

And a corresponding lower bound:

Lemma C.12. *Let $\mathbb{S}$ be a necessary set of messages for a function f_v on a graph G. Then*

$$\text{MPC}(\mathcal{M}, f_v, G) \geq -\log(\mathbb{P}[\mathbb{S}]).$$

Proof. It follows directly from Definition C.2 that

$$\begin{aligned}\mathbb{P}[\text{lossyWL}_v(t(G)) \vDash f_v(G)] &\leq \mathbb{P}[\bigwedge_{(a,b,l)\in\mathbb{S}} Z^l_{ab} = 1] \\ &= \mathbb{P}[\mathbb{S}]\end{aligned}$$

□

Retaining information We will first prove the complexity bound Lemma 5.1 for the task of retaining the initial node feature: $f_v(G) = \boldsymbol{X}_v$.

Lemma 5.1. *Assume the degree $r \geq 2$. Then the expected* MPC *complexity for this task* $\mathbb{E}_{G,v\sim\mathcal{G}_\mathcal{V}}[\text{MPC}_{\mathcal{M}}(f_v, G)]$ *grows at least linearly with L, i.e., is in $\Omega(L)$, for all MPNNs $\mathcal{M}$.*

Proof. Note that for $r \geq 2$, $\boldsymbol{I}_{ab} < 1$ for all nodes $a, b \in \mathcal{V}_G$ for all model architectures except the MLP. Then, with positive probability, all messages are lost in a layer l, i.e., $Z^l_{a,b} = 0$ for all $a, b \in \mathcal{V}_G$. Hence, only with probability $\phi < 1$ any message is successful in layer l. A necessary condition for retaining the initial node feature is that in every layer, at least one message is successful. Thereby, we have the following lower bound:

$$\mathbb{P}[\text{lossyWL}_v(t(G)) \vDash f(G)] \leq \phi^L.$$

It follows immediately, that

$$\text{MPC}(\mathcal{M}, f_v, G) \geq L\log(1/\phi).$$

□

Propagating Information We will now prove the complexity bound Lemma 5.2 for the task $f_v(G) = \boldsymbol{X}_u$ of propagating information from a source node u to a target node v at distance $D = d_G(u, v)$.

Lemma 5.2. *Assume L is the minimum depth required to solve this task with $\mathcal{M}$. Then the expected complexity* $\mathbb{E}_{G,v\sim\mathcal{G}_\mathcal{V}}[\text{MPC}_{\mathcal{M}}(f_v, G)]$ *is $\leq 2\log(n)$ for GCN-VN, while for standard MPNNs it is $\geq D\log(r)$ provided n is sufficiently large given D.*

Proof. Trivially, for the **MLP baseline** and $D > 0$, we have:

$$\text{MPC}_{\text{MLP}}(f_v, G) = \infty.$$

For **GCN-VN**, notice that each original node in $\tilde{G}$ has degree $r + 2$ (r neighbors in the input graph, the self-loop, and the connection to the virtual node) and the virtual node has degree $1/n$. It is easy to see that for $D = 0$ and $D = 1$, the minimal depth required to solve this task is $L = 0$ and $L = 1$, respectively. Trivially, the MPC complexities for these cases are 0 and $\log(r + 2)$. Both are $\leq 2\log(n)$.

For $D \geq 2$, the minimal depth required to solve this tasks is $L = 2$: One sufficient set of messages consists of the message from u to the virtual node (which has probability of success $1/n$) and from the virtual node to v (which has probability of success $1/(r + 2)$). By Lemma C.11, this implies

$$\text{MPC}_{\text{GCN-VN}}(f_v, G) \leq \log(n(r + 2)).$$

And hence,

$$\text{MPC}_{\text{GCN-VN}}(f_v, G) \leq 2\log(n).$$

For all **other MPNNs**, notice that the probability that $G, v \sim \mathcal{G}_{\mathcal{V}}$ contains a cycle where every node is in the D-hop neighborhood around v approaches 0 as $n \to \infty$ for a given D. Hence, we can choose n such that this probability is $\leq \epsilon$ for any $\epsilon > 0$.

Then, we can bound the expected complexity by (as MPC is always non-negative):

$$\begin{aligned}
&\mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G)] \\
&\geq \mathbb{P}[\text{no cycle around } v] \cdot \mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G) \mid \text{no cycle around } v] \\
&= (1 - \epsilon) \cdot \mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G) \mid \text{no cycle around } v]
\end{aligned}$$

where "no cycle around v" denotes the event that there is no cycle where every node is in the D-hop neighborhood around v.

Next, we need to bound $\mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G) \mid \text{no cycle around } v]$. For this, we will show that $\text{MPC}_{\mathcal{M}}(f_v, G) \geq \log((r + 1)^L)$ for any $G, v \in \mathcal{G}_{\mathcal{V}}$ without a cycle around v. It is easy to see that for all considered MPNNs without a virtual node, the minimal depth required to solve this task is then $L = D$. Then, a necessary set of messages consists of the messages on the only (as there is no cycle around v) D-length path from u to v. Each message on this path has a success probability $\leq 1/(r + 1)$. By Lemma C.12, this implies:

$$\text{MPC}_{\mathcal{M}}(f_v, G) \geq \log((r + 1)^D)$$

Finally, by choosing ϵ small enough, we have

$$\begin{aligned}
&\mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G)] \\
&\geq (1 - \epsilon) \cdot \mathbb{E}_{G,v\sim\mathcal{G}_{\mathcal{V}}}[\text{MPC}_{\mathcal{M}}(f_v, G) \mid \text{no cycle around } v] \\
&\geq (1 - \epsilon) \cdot \log((r + 1)^D) \\
&\geq D\log(r)
\end{aligned}$$

□

Extracting topological information Lastly, we consider the ring transfer task. To simplify notations here, we will consider the complexity of $f_v(G) = \boldsymbol{X}_u$ for a *single node* u on the cycle of size s that v is part of. The tasks involve 1) identifying that u is part of the cycle 2) propagating the node feature information from u to v. We will now prove the complexity bounds in Lemma 5.3:

Lemma C.13. *Assume L is the minimal depth required to solve the task $f_v(G) = \boldsymbol{X}_u$ with architecture $\mathcal{M}$ where u is a node on the cycle of size s that v is part of. Further assume that there is only a single cycle in the $\lceil s/2 \rceil$-hop neighborhood of v. Then, the expected* MPC *complexities are:*

For CIN & FragNet: $O(\log(sr))$ *For GSN: $\leq \lceil s/2 \rceil \log(r+1)$* *For all others: $\geq s\log(r)$*

Proof. **FragNet:** Notice that for FragNet $L = 2$ is the minimal depth required to solve this task. With $L = 2$ a sufficient set of messages involves a message from u to the fragment node that represents this cycle (with message success probability $1/s$) and from the fragment node to v (with message success probability $1/(r + 2)$). Using Lemma C.11, the bound follows immediately.

CIN: Again, for CIN $L = 2$ is the minimal depth required to solve this task. A sufficient set of messages involves a message from the ring node (which contains as initial encoding all node features from the nodes in the ring) to an edge node. And from the edge node to v. Notice that the edge node has degree $s + 2$ (an edge to every other edge node in the cycle, an edge to the ring node and two edges to the original nodes) and v has degree $2r + 1$ (edges to all r neighboring nodes in G, edges to corresponding edge nodes, and self-loop). Hence, we have message success probabilities $1/(s + 2)$ and $1/(2r + 1)$. Again, using Lemma C.11, the bound follows immediately.

GSN: Notice that for GSN the cycle information is already contained in the initial node-feature. So the task is simply to propagate the node feature information to v for a distance of at most $\lceil s/2 \rceil$. Hence, a sufficient set of messages consists of the messages on a path of length $L \leq \lceil s/2 \rceil$ from u to v, each with success probability $1/(r + 1)$. Again, using Lemma C.11, the bound follows immediately.

All other MPNNs: For all other MPNNs, the minimum depth required to solve this task is $L = s$ (for $L < s$, it is impossible to detect the cycle). It is easy to see that in order to identify the cycle at node v, there needs to be a set of successful messages going around the full cycle returning to v. Or more formally, for a cycle consisting of nodes $v, a_1, a_2, \dots, a_{s-1}, v$, we define two possible message traversal patterns:

$$\text{Cycle}_{\circlearrowleft} := Z^1_{va_1} = 1 \wedge Z^2_{a_1a_2} = 1 \wedge Z^3_{a_2a_3} = 1 \wedge \cdots \wedge Z^s_{a_{s-1}v} = 1$$

and

$$\text{Cycle}_{\circlearrowright} := Z^1_{va_{s-1}} = 1 \wedge Z^2_{a_{s-1}a_{s-2}} = 1 \wedge Z^3_{a_{s-2}a_{s-3}} = 1 \wedge \cdots \wedge Z^s_{a_1v} = 1.$$

Additionally, node v must retain its unique node feature throughout all L layers to recognize that the returning messages originated from itself.[3] Hence, a necessary condition for $\text{lossyWL}_v(G) \vDash f_v(G)$ is

$$Z^1_{vv} = 1 \wedge Z^2_{vv} = 1 \wedge \cdots \wedge Z^s_{vv} = 1 \wedge (\text{Cycle}_{\circlearrowleft} \vee \text{Cycle}_{\circlearrowright}).$$

Therefore, we have

$$\mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G)] \leq \mathbb{P}[Z^1_{vv} = 1 \wedge Z^2_{vv} = 1 \wedge \cdots \wedge Z^s_{vv} = 1 \wedge (\text{Cycle}_{\circlearrowleft} \vee \text{Cycle}_{\circlearrowright})].$$

Notice that $\mathbb{P}[Z^l_{ab}] = 1/(r + 1)$ for any message, as every node has degree r and the additonal self-loop. Then, because of the independence of all Z^l_{ab}, we have

$$\begin{aligned}
\mathbb{P}[\text{lossyWL}_v(G) \vDash f_v(G)] &\leq \mathbb{P}[Z^1_{vv} = 1 \wedge Z^2_{vv} = 1 \wedge \cdots \wedge Z^s_{vv} = 1 \wedge (\text{Cycle}_{\circlearrowleft} \vee \text{Cycle}_{\circlearrowright})]. \\
&= \mathbb{P}[Z^1_{vv} = 1 \wedge Z^2_{vv} = 1 \wedge \cdots \wedge Z^s_{vv} = 1] \cdot \mathbb{P}[\text{Cycle}_{\circlearrowleft} \vee \text{Cycle}_{\circlearrowright}] \\
&\leq 1/2 \cdot \mathbb{P}[\text{Cycle}_{\circlearrowleft} \vee \text{Cycle}_{\circlearrowright})] \\
&\leq 1/2 \cdot 2 \cdot \mathbb{P}[\text{Cycle}_{\circlearrowleft}] \\
&= \left(\frac{1}{r + 1}\right)^s.
\end{aligned}$$

While this bound is not tight (we've made several relaxations), it is sufficient to establish the $\geq s \log(r)$ MPC complexity lower bound.

MLP: The complexity is trivially infinite. □

Lastly, we give a short proof why WLC is 0 for this task. For this, we generally show that for graphs with unique node features any (at least WL-expressive) architecture with at least one layer can distinguish all non-isomorphic graphs, and hence solve any task.

Lemma C.14. *Let $\mathcal{G}$ be a family of graphs with unique node features, i.e., for any $G \in \mathcal{G}$, no two nodes $v, w \in \mathcal{V}_G$ have the same node features. Then the WL output after one iteration $\{\!\{\text{WL}^1_v \mid v \in \mathcal{V}_G\}\!\}$ differs for any two non-isomorphic graphs in $\mathcal{G}$.*

Proof. Let $G_1, G_2 \in \mathcal{G}$ be two non-isomorphic graphs. Suppose, for contradiction, that the WL output after one iteration is the same for both graphs, i.e., $\{\!\{\text{WL}^1_v \mid v \in \mathcal{V}_{G1}\}\!\} = \{\!\{\text{WL}^1_w \mid w \in \mathcal{V}_{G2}\}\!\}$ as multisets.

[3] A sufficient condition for this task would be even stronger, requiring all nodes in the cycle to retain their features until the messages through the cycle reach them.

Since all node features in G_1 and G_2 are unique within each graph, the initial WL labels WL_v^0 are distinct for all nodes v in each graph. After one iteration, each WL_v^1 consists of the node's original feature combined with the multiset of its neighbors' features, i.e., $\mathrm{WL}_v^1 = \big(\mathrm{WL}_v^0, \{\!\{\mathrm{WL}_u^0 \mid u \in N(v)\}\!\}\big)$.

Given that $\{\!\{\mathrm{WL}_v^1 \mid v \in \mathcal{V}_{G1}\}\!\} = \{\!\{\mathrm{WL}_w^1 \mid w \in \mathcal{V}_{G2}\}\!\}$, there must exist a bijection $f : \mathcal{V}_{G1} \to \mathcal{V}_{G2}$ such that $\mathrm{WL}_v^1 = \mathrm{WL}_{f(v)}^1$ for all $v \in \mathcal{V}_{G1}$.

This implies that for each $v \in \mathcal{V}_{G1}$: (1) $\mathrm{WL}_v^0 = \mathrm{WL}_{f(v)}^0$ (the original node features match), and (2) $\{\!\{\mathrm{WL}_u^0 \mid u \in N(v)\}\!\} = \{\!\{\mathrm{WL}_w^0 \mid w \in N(f(v))\}\!\}$ (the multisets of neighbor features match).

From (1), since node features are unique within each graph, f maps each node in G_1 to a unique node in G_2 with identical features.

From (2), for each $v \in \mathcal{V}_{G1}$, there must exist a bijection $g_v : N(v) \to N(f(v))$ such that $\mathrm{WL}_u^0 = \mathrm{WL}_{g_v(u)}^0$ for all $u \in N(v)$.

Due to the uniqueness of node features and (1), we must have $g_v(u) = f(u)$ for all $u \in N(v)$. This implies that for all $v \in \mathcal{V}_{G1}$ and $u \in \mathcal{V}_{G1}$: $u \in N(v) \iff f(u) \in N(f(v))$ Therefore, f is an isomorphism between G_1 and G_2, contradicting our assumption that the graphs are non-isomorphic. Thus, the WL output after one iteration must differ for any two non-isomorphic graphs in $\mathcal{G}$. □

D Additional results and Experimental Details

We present the following additional results:

- Extended figures showing *all* considered model architectures across all tasks.
- Robustness analysis demonstrating that MPC predictions hold across different hyperparameter choices (hidden dimensions) for the information retention task.
- Validation on Erdős-Rényi (ER) graphs, confirming that MPC accurately matches performance trends also on a different synthetic graph family.
- Real-world validation on graphs from ZINC [15] and peptides datasets [14] showing that MPC explains performance better than classical expressivity theory for real-world graph structures.
- Results showing that insight on synthetic proxy tasks can mirror performance trends on real-world benchmarks where the precise tasks are unknown.

Additionally, we carefully describe all experimental details and setups.

D.1 Implementation Details and Experimental Setup

We use pytorch [39] and torch-geometric [17] (released under an MIT license) for the implementation of all models. For optimization, we use the Adam optimizer [30]. The training, validation and test sets containing random r-regular graphs are generated using networkx [24]. For all standard MPNNs we use their pytorch geometric implementation. Note that GSN [8] only specifies the additional node features and not the downstream MPNN. We use the most common MPNN, GCN, as downstream MPNN for all experiments for GSN. For CIN [5] we use our own implementation following exactly the method proposed in their paper. Additionally, for FragNet, we use a custom implementation of their FR-WL model without edge representations using a GCN as base MPNN. For CIN, GSN, and FragNet we use a fragmentation scheme identifying every cycle of size at most 6 (unless otherwise noted). For all models we use an initial feature embedding layer and a final output MLP. Additionally, we use BatchNorm [26] for normalization. We found little difference between learning rates in $\{0.001, 0.005, 0.01\}$ for all models and tasks. The shown results are for the learning rate 0.005. We train all models for all settings using three different seeds for a maximum of 50 epochs, showing the average results. All experiments are conducted on NVIDIA GeForce GTX 1080 GPUs with 16GB memory allocation per job. Training times vary by model architecture, ranging from 10-30 minutes. Monte Carlo simulations for complexity calculations run on a single Intel Xeon E5-2630 v4 CPU (2.20GHz) and complete in under 10 seconds. The peptides dataset of the lrgb [14] is released under a CC BY-NC 4.0 license. The ZINC dataset [15] is distributed under a custom license (free to use for everyone).

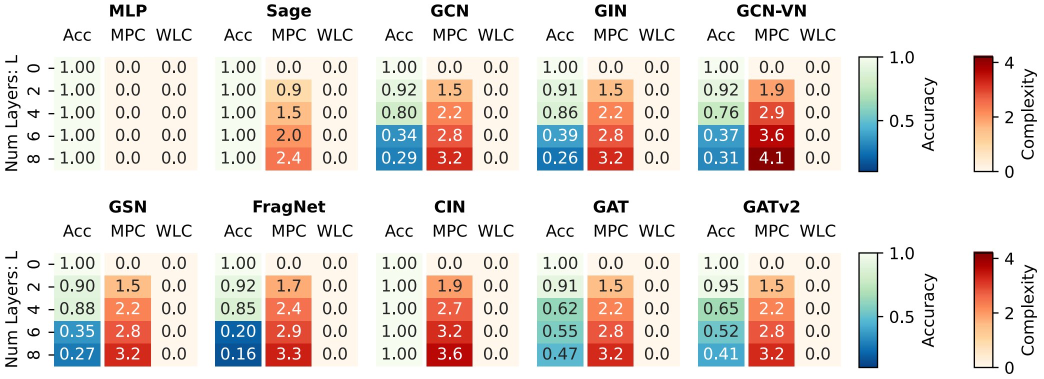
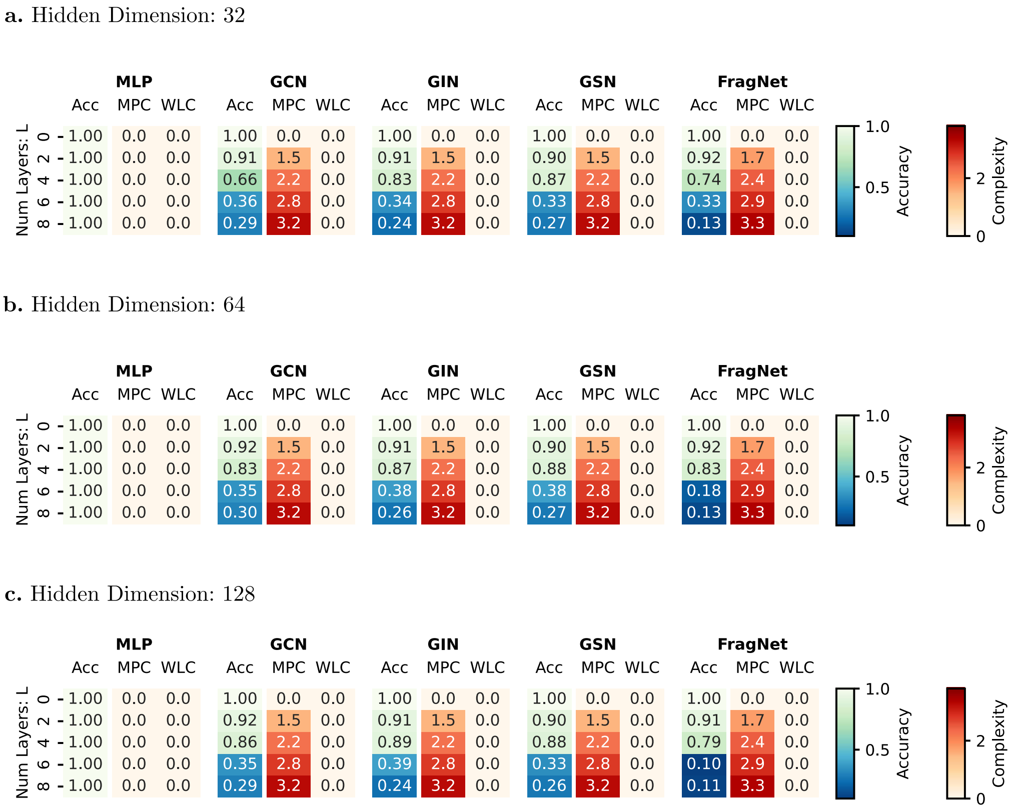


Figure 9: Test accuracy for retaining initial node features compared with complexity measures MPC and WLC for all models. Simulated MPC (in contrast to WL-based WLC) matches trends in empirical accuracy (for all models except for Sage and CIN), capturing increasing difficulty with depth.

D.2 Retaining information

For the task $f_v(G) = \boldsymbol{X}_u$, we randomly generate 3-regular graphs with $n = 50$ nodes where each node is randomly assigned to one of ten possible classes. We use a training set of size 2000, a validation set of size 500 and a test set of size 2000. We simulate the complexities using Monte Carlo simulation on 100 different graphs from the test set, and for each graph 1000 trials (with the same method explained in App. D.3). Fig. 9 shows the accuracy and simulated MPC complexity for all considered model architectures. Additionally, Fig. 10 shows that the complexity measure is robust to changes in the hidden dimension hyperparameter, i.e., trends in MPC complexity align with empirical performance also for different hidden dimensions. Moreover, Fig. 11 shows how complexity and accuracy changes for different levels of sparsity.

Erdős–Rényi graphs We repeat the retaining information experiment with Erdős–Rényi (ER) graphs with $n = 50$ nodes and node connection probability $p = 0.06$. Fig. 12 shows that generally, trends in MPC complexity align with trends in empirical performance (except for Sage and CIN, the two models that directly optimize the weight of the residual channel) as well for ER graphs.



a. Hidden Dimension: 32

Num Layers: L	MLP			GCN			GIN			GSN			FragNet		
	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.91	1.5	0.0	0.91	1.5	0.0	0.90	1.5	0.0	0.92	1.7	0.0
4	1.00	0.0	0.0	0.66	2.2	0.0	0.83	2.2	0.0	0.87	2.2	0.0	0.74	2.4	0.0
6	1.00	0.0	0.0	0.36	2.8	0.0	0.34	2.8	0.0	0.33	2.8	0.0	0.33	2.9	0.0
8	1.00	0.0	0.0	0.29	3.2	0.0	0.24	3.2	0.0	0.27	3.2	0.0	0.13	3.3	0.0

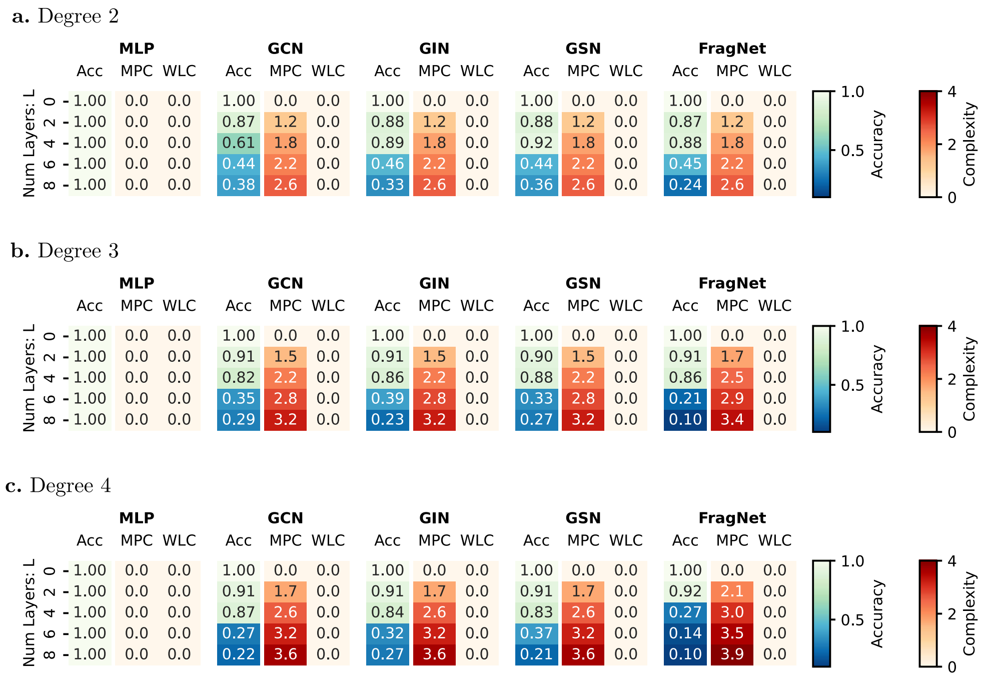
b. Hidden Dimension: 64

Num Layers: L	MLP			GCN			GIN			GSN			FragNet		
	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.92	1.5	0.0	0.91	1.5	0.0	0.90	1.5	0.0	0.92	1.7	0.0
4	1.00	0.0	0.0	0.83	2.2	0.0	0.87	2.2	0.0	0.88	2.2	0.0	0.83	2.4	0.0
6	1.00	0.0	0.0	0.35	2.8	0.0	0.38	2.8	0.0	0.38	2.8	0.0	0.18	2.9	0.0
8	1.00	0.0	0.0	0.30	3.2	0.0	0.26	3.2	0.0	0.27	3.2	0.0	0.13	3.3	0.0

c. Hidden Dimension: 128

Num Layers: L	MLP			GCN			GIN			GSN			FragNet		
	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC	Acc	MPC	WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.92	1.5	0.0	0.91	1.5	0.0	0.90	1.5	0.0	0.91	1.7	0.0
4	1.00	0.0	0.0	0.86	2.2	0.0	0.89	2.2	0.0	0.88	2.2	0.0	0.79	2.4	0.0
6	1.00	0.0	0.0	0.35	2.8	0.0	0.39	2.8	0.0	0.33	2.8	0.0	0.10	2.9	0.0
8	1.00	0.0	0.0	0.29	3.2	0.0	0.24	3.2	0.0	0.26	3.2	0.0	0.11	3.3	0.0

Figure 10: Influence of hidden dimension hyperparameter. Test accuracy for retaining initial node feature task compared with complexity measures MPC and WLC for hidden dimension size **a.** 32, **b.** 64 and **c.** 128 (for a selection of models). Even with larger hidden dimensions, MPNNs face fundamental over-smoothing limitations that MPC captures while WLC does not. MPC consistently predicts performance trends across all hidden dimension choices.



a. Degree 2

Num Layers: L	MLP Acc	MLP MPC	MLP WLC	GCN Acc	GCN MPC	GCN WLC	GIN Acc	GIN MPC	GIN WLC	GSN Acc	GSN MPC	GSN WLC	FragNet Acc	FragNet MPC	FragNet WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.87	1.2	0.0	0.88	1.2	0.0	0.88	1.2	0.0	0.87	1.2	0.0
4	1.00	0.0	0.0	0.61	1.8	0.0	0.89	1.8	0.0	0.92	1.8	0.0	0.88	1.8	0.0
6	1.00	0.0	0.0	0.44	2.2	0.0	0.46	2.2	0.0	0.44	2.2	0.0	0.45	2.2	0.0
8	1.00	0.0	0.0	0.38	2.6	0.0	0.33	2.6	0.0	0.36	2.6	0.0	0.24	2.6	0.0

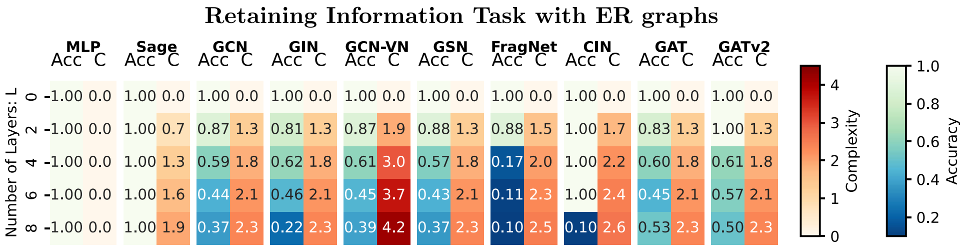
b. Degree 3

Num Layers: L	MLP Acc	MLP MPC	MLP WLC	GCN Acc	GCN MPC	GCN WLC	GIN Acc	GIN MPC	GIN WLC	GSN Acc	GSN MPC	GSN WLC	FragNet Acc	FragNet MPC	FragNet WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.91	1.5	0.0	0.91	1.5	0.0	0.90	1.5	0.0	0.91	1.7	0.0
4	1.00	0.0	0.0	0.82	2.2	0.0	0.86	2.2	0.0	0.88	2.2	0.0	0.86	2.5	0.0
6	1.00	0.0	0.0	0.35	2.8	0.0	0.39	2.8	0.0	0.33	2.8	0.0	0.21	2.9	0.0
8	1.00	0.0	0.0	0.29	3.2	0.0	0.23	3.2	0.0	0.27	3.2	0.0	0.10	3.4	0.0

c. Degree 4

Num Layers: L	MLP Acc	MLP MPC	MLP WLC	GCN Acc	GCN MPC	GCN WLC	GIN Acc	GIN MPC	GIN WLC	GSN Acc	GSN MPC	GSN WLC	FragNet Acc	FragNet MPC	FragNet WLC
0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0	1.00	0.0	0.0
2	1.00	0.0	0.0	0.91	1.7	0.0	0.91	1.7	0.0	0.91	1.7	0.0	0.92	2.1	0.0
4	1.00	0.0	0.0	0.87	2.6	0.0	0.84	2.6	0.0	0.83	2.6	0.0	0.27	3.0	0.0
6	1.00	0.0	0.0	0.27	3.2	0.0	0.32	3.2	0.0	0.37	3.2	0.0	0.14	3.5	0.0
8	1.00	0.0	0.0	0.22	3.6	0.0	0.27	3.6	0.0	0.21	3.6	0.0	0.10	3.9	0.0

Figure 11: Influence of degree r of the random regular graphs. Test accuracy for retaining initial node feature task compared with complexity measures MPC and WLC for hidden dimension size **a.** 32, **b.** 64 and **c.** 128 (for a selection of models). Unlike MPC, MPC captures the increasing difficulty as graph degree increases, demonstrating the importance of considering the graph topology.



Number of Layers: L	MLP Acc	MLP C	Sage Acc	Sage C	GCN Acc	GCN C	GIN Acc	GIN C	GCN-VN Acc	GCN-VN C	GSN Acc	GSN C	FragNet Acc	FragNet C	CIN Acc	CIN C	GAT Acc	GAT C	GATv2 Acc	GATv2 C
0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0	1.00	0.0
2	1.00	0.0	1.00	0.7	0.87	1.3	0.81	1.3	0.87	1.9	0.88	1.3	0.88	1.5	1.00	1.7	0.83	1.3	1.00	1.3
4	1.00	0.0	1.00	1.3	0.59	1.8	0.62	1.8	0.61	3.0	0.57	1.8	0.17	2.0	1.00	2.2	0.60	1.8	0.61	1.8
6	1.00	0.0	1.00	1.6	0.44	2.1	0.46	2.1	0.45	3.7	0.43	2.1	0.11	2.3	1.00	2.4	0.45	2.1	0.57	2.1
8	1.00	0.0	1.00	1.9	0.37	2.3	0.22	2.3	0.39	4.2	0.37	2.3	0.10	2.5	0.10	2.6	0.53	2.3	0.50	2.3

Figure 12: MPC captures practical difficulty across different graph families. Test accuracy and average simulated MPC complexity for retaining initial node features on Erdős–Rényi graphs (50 nodes, connection probability p = 0.06). Models are trained on 2000 graphs. MPC complexity (C) correctly increases with layer depth (WLC would be 0 everywhere), demonstrating that our complexity measure generalizes beyond random regular graphs to capture practical learning difficulties.

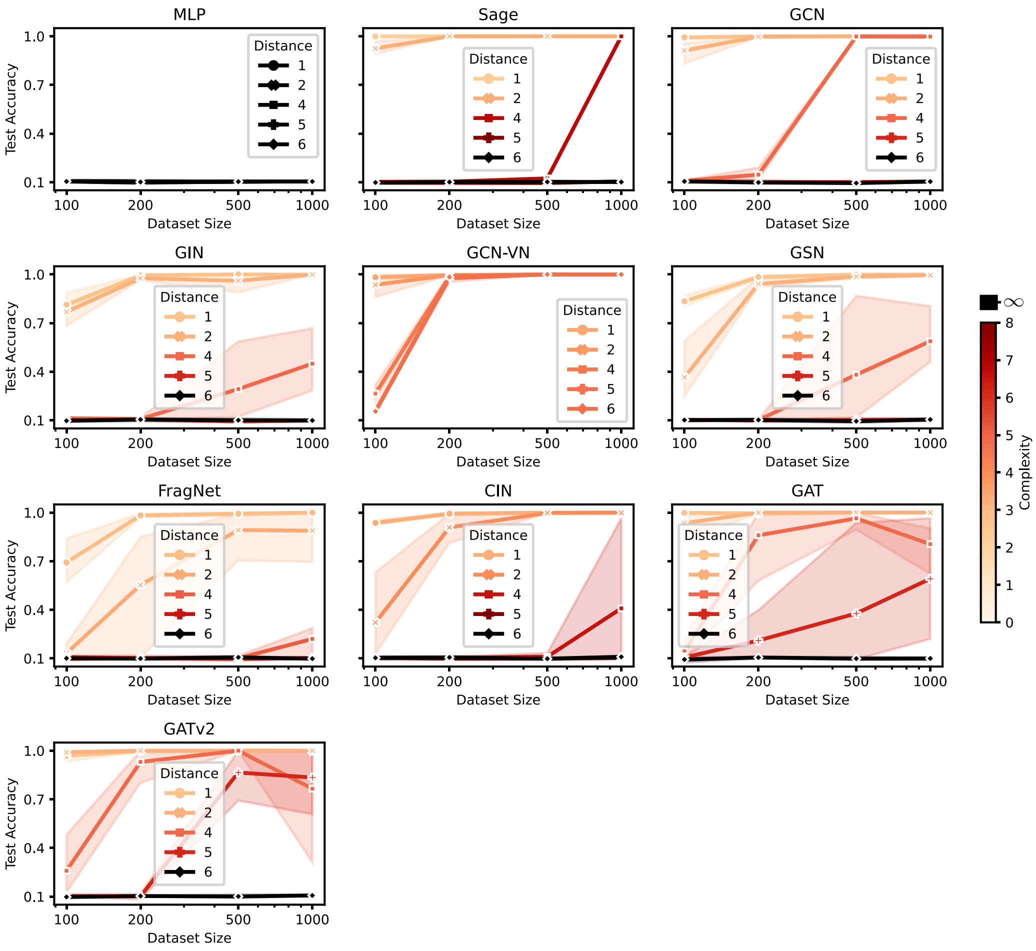


Figure 13: Test accuracy by training data size for the information propagation task $f_v(G) = \boldsymbol{X}_u$ for different distances D. Colored by average simulated MPC complexity per distance. MPC correctly captures the increasing sample complexity with distance and the performance advantage that a virtual node offers for long-range dependencies.

D.3 Propagating Information

For the task $f_v(G) = \boldsymbol{X}_u$, we randomly generate 3-regular graphs with $n = 50$ nodes. In each graph, we randomly select a target node v. For a given distance D, we select a node u with distance D from v. Then, v gets the unique label 0 identifying it as target node, u is randomly assigned a target label in $\{1, \ldots, 10\}$, and all other nodes get a random label from $\{11, \ldots, 20\}$. For each distance D, we use training sets of different sizes and a validation set of size 500 and a test set of size 2000. We simulate the complexities using Monte Carlo simulation on 100 different graphs from the test set, and for each graph 1000 trials. Fig. 13 shows the average test accuracy by training data size and Fig. 14 shows the test accuracy and the complexity for the dataset size 1000.

Erdős–Rényi graphs Again, we repeat the information propagation experiment with ER graphs with $n = 50$ nodes and node connection probability $p = 0.04$. Fig. 15 show that trends in MPC complexity align with trends in empirical performance as well for ER graphs.

Peptides dataset Additionally, we repeat the information propagation task on graphs from the real-world peptides-func/peptides-struct dataset. It comprises larger peptide molecules that require models to consider long-range interactions [14]. Therefore, the information propagation task, testing

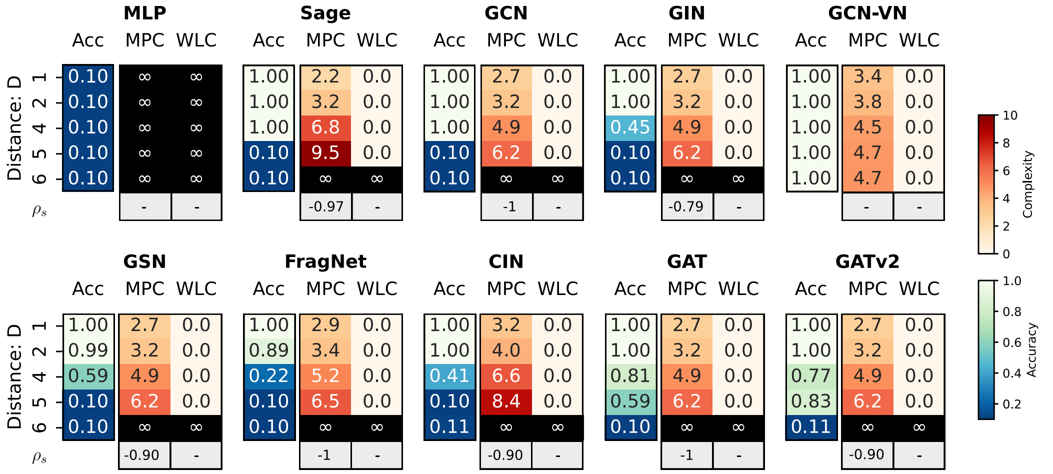


Figure 14: Test accuracy for retaining initial node features compared with complexity measures MPC and WLC for the information propagation task $f_v(G) = \boldsymbol{X}_u$ for all models (for dataset size 1000). Simulated MPC matches trends in empirical accuracy (highly negative Spearman correlated ρ_S), capturing increasing difficulty with distance while preserving impossibility statements from WLC.

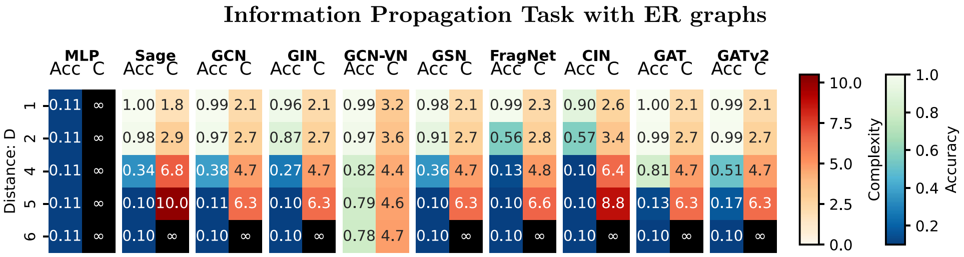


Figure 15: Test accuracy by distance for the information propagation task $f_v(G) = \boldsymbol{X}_u$ (for varying train dataset sizes) and simulated MPC complexity (C) for ER graphs with $n = 50$ nodes and connection probability $p = 0.04$. MPC again matches empirical performance trends, demonstrating that our complexity measure generalizes beyond random regular graphs to capture practical learning difficulties.

the models' ability to exchange information over varying distances, is particularly well-suited for this dataset.

Similar to our approach for random regular and ER graphs, we randomly select one node of each graph in the dataset as target node v. For a given distance D, we randomly select a node u with distance D from v as the source node. Then, v gets the unique label 0 identifying it as target node, u is randomly assigned a label in $\{1, \ldots, 10\}$, and all other nodes get a random label from $\{11, \ldots, 20\}$. We then randomly sample different numbers of graphs from the dataset as train graphs. In summary, this allows us to analyze and isolate the information propagation capability of models on real-world graphs requiring long-range interactions.

Fig. 17 shows again a connection between sample complexity needed to achieve perfect accuracy and MPC complexity, underlining that MPC is also a good predictor of performance for graph topologies occurring in the real world.

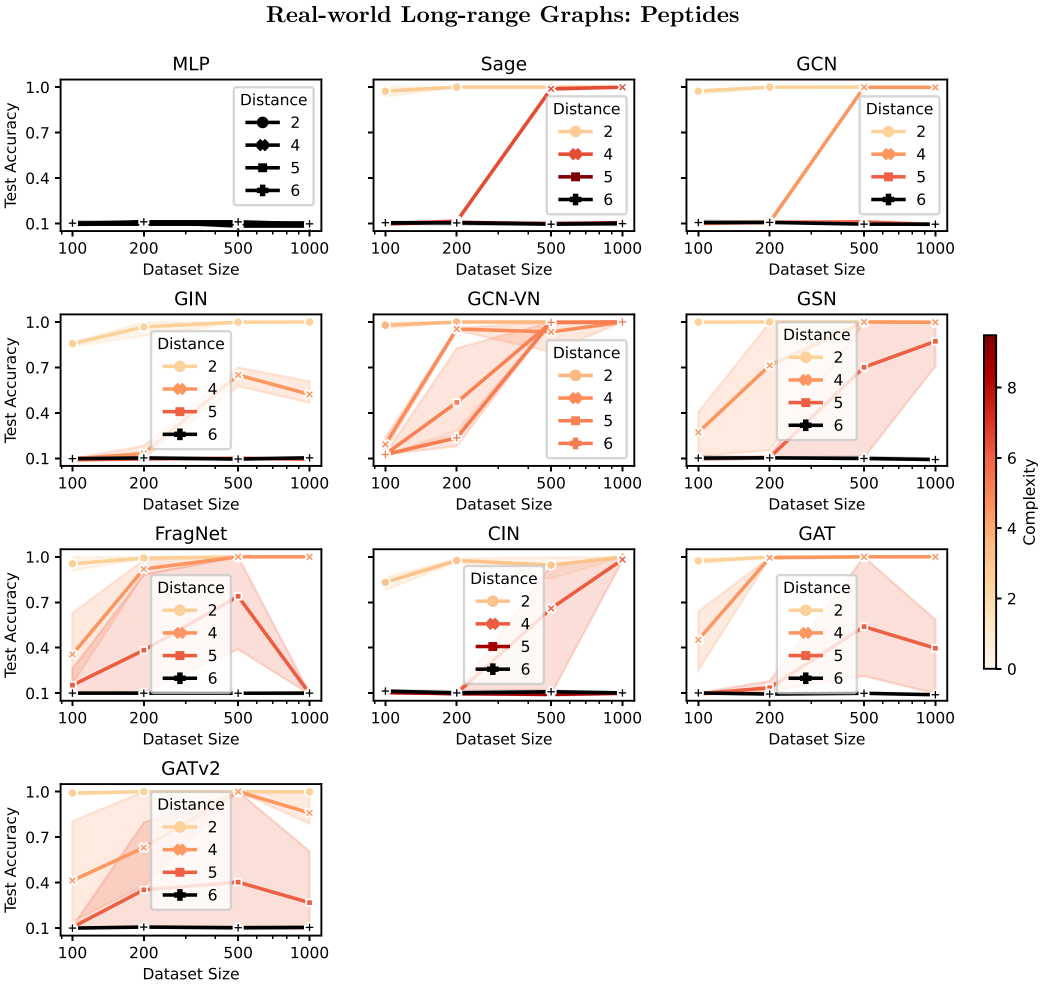


Figure 17: **MPC captures task difficulty on real-world graphs.** Test accuracy by distance for the information propagation task $f_v(G) = \boldsymbol{X}_u$ across varying training dataset sizes, with simulated MPC complexity shown for real-world graphs from the peptides-func/peptides-struct dataset [14]. Higher MPC complexity corresponds to greater learning difficulty, as evidenced by increased sample complexity requirements, demonstrating that MPC effectively captures practical task difficulty beyond synthetic graph families.

Table 2: Comparison of MPC for ring transfer task (for ring size 6) and Mean Absolute Error on prediction penalized logP, both on graphs from ZINC-subset. Trends in MPC on the synthetic ring task match trends on the real-world task, where identifying large rings is important.

	Standard MPNNs			Substructure Encodings	Substructure Graphs	
	GIN	**GraphSage**	**GCN**	**GSN**	**FragNet**	**CIN**
MPC (Ring Task)	7.3	7.3	7.3	3.6	3.2	2.9
MAE (ZINC)	0.53	0.40	0.37	0.12	0.078	0.077

D.4 Extracting topological information

For the ring transfer task, we use randomly generated 4-regular graphs with $n = 50$ nodes conditioned on having a cycle of size s at a node v (and no smaller cycle). Each node is randomly assigned a unique label in $\{1, \dots, 50\}$. The multilabel classification task $f_v(G)$ is then to classify which labels between 1 and 50 are part of the cycle that contains v. For each cycle size, we generate training sets of varying size, a validation set of size 1000 and a test set of size 10000. For GSN, FragNet, and CIN we use a fragmentation scheme identifying cycles of size at most 5. For all models, we use the minimal number of layers with which they can solve all tasks: for FragNet and CIN: 2, for all other models: 5. For CIN and FragNet, we explicitly compute the complexities. For all other models, we use the bounds provided in Lemma 5.3. All results show the binary average precision. Fig. 18 shows the average precision and complexities for this task for the maximal dataset size of 10000. Additionally, Fig. 19 shows the average precision in relation to dataset size.

Erdős–Rényi graphs Again, we repeat the ring transfer task with ER graphs with $n = 50$ nodes and edge probability $p = 0.04$ conditioned on having a ring of size s at node v. The label assignment is done in the same way as for the random regular graphs. Fig. 20 shows that complexities also align with MPC complexities for ER graphs.

ZINC dataset We additionally test the ring transfer capabilities of the models on real-world graphs from the ZINC molecular regression dataset [15]. The ZINC dataset comprises small molecules and the benchmark task is to predict the penalized logP score which involves the number of cycles. Therefore the graphs from the ZINC dataset are well-suited to test our models cycle detection capabilities. For this, we first filter all graphs to contain rings of size s. Second, we randomly choose

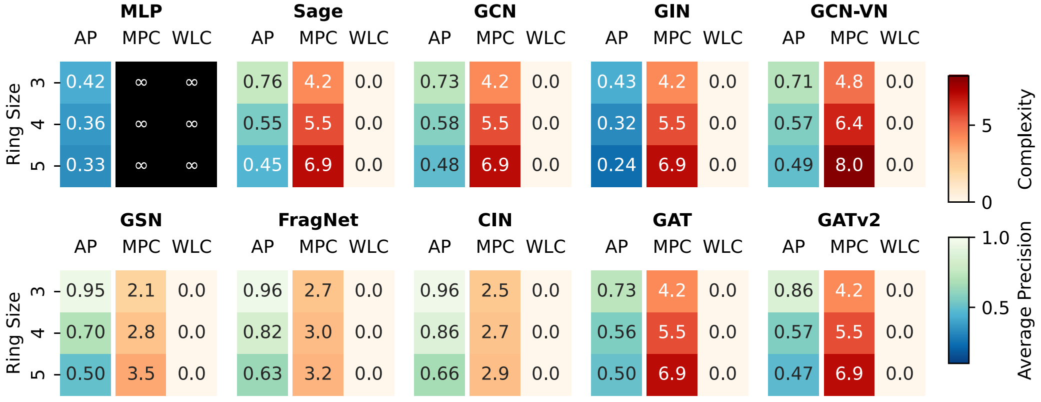


Figure 18: Test average precision for the ring transfer task compared with complexity measures MPC and WLC across all models (dataset size 10000, MPC values for all standard MPNNs are lower bounds). Simulated MPC aligns with empirical accuracy trends, capturing both the increasing difficulty with ring size and the superior performance of GSN, FragNet, and CIN due to their cycle-oriented inductive biases.

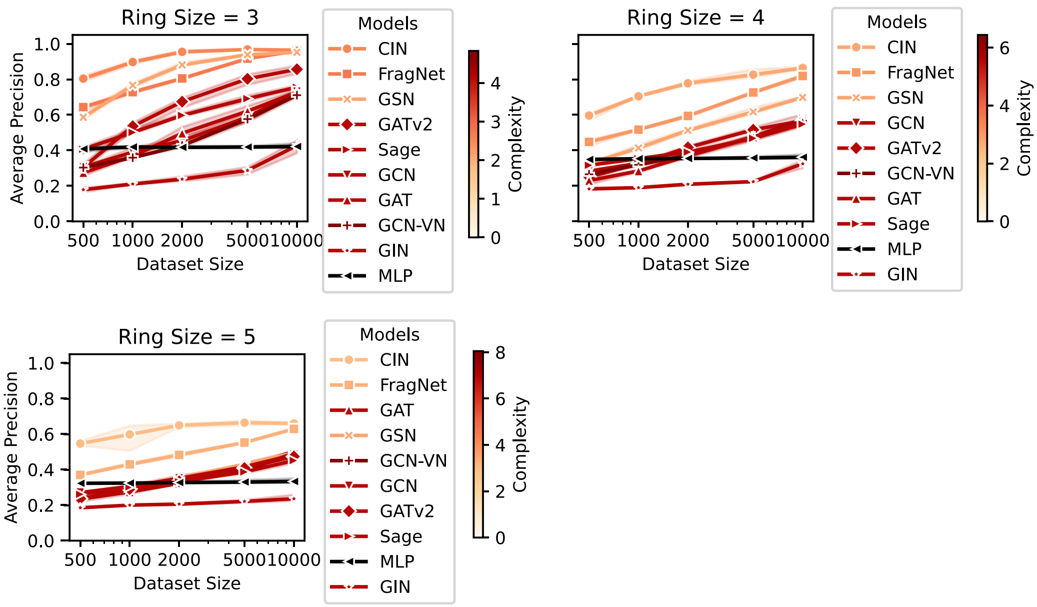


Figure 19: Test average precision and by dataset size for the ring transfer task in relation to dataset size. Models are colored by their simulated MPC complexity.

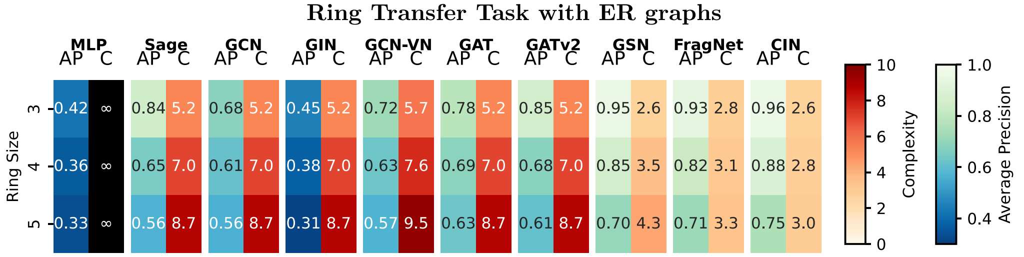


Figure 20: Test average precision and average MPC complexity (C) for the ring transfer task with ER graphs for all models for training data size 10000 (complexity values for standard MPNNs are lower bounds), demonstrating again that MPC matches performance trends for graphs from a different distribution.

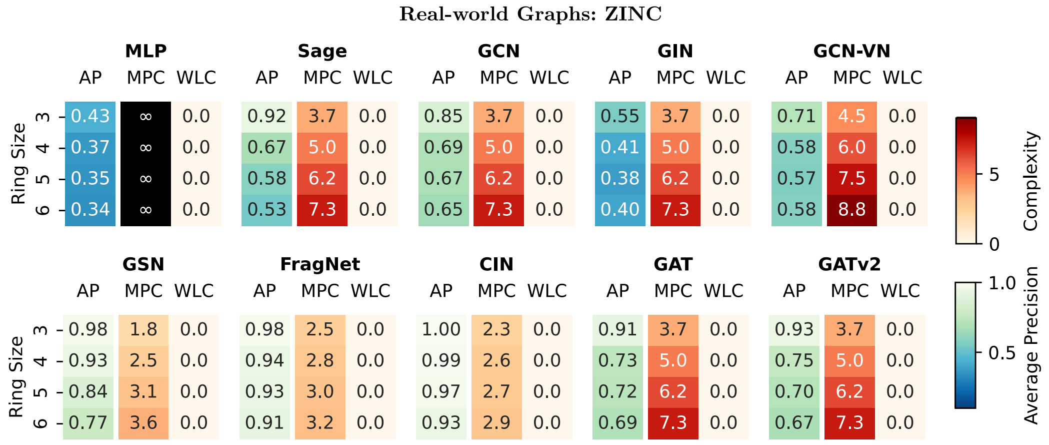


Figure 22: Test average precision compared with complexity measures MPC and WLC for the ring transfer task for real-world graphs from the ZINC dataset, where cycle detection is a crucial subtask (dataset size 10000, complexity values for standard MPNNs are lower bounds). MPC can account for performance differences in this real-world dataset that classical expressivity theory misses: The superior performance of GSN, CIN, and FragNet cannot be explained by their increased iso expressivity(WLC 0 for all MPNNs) but by their reduced MPC due to their cycle-oriented inductive bias.

a node v that is part of cycle s for each graph. The labels are assigned in the same way as for the random regular graphs and the ER graphs.

Figs. 22 and 23 show that trends in MPC complexity align again with empirical performance. Again, the superior performance of GSN, CIN, and FragNet cannot be explained by their increased iso expressivity(WLC 0 for all MPNNs) but by their reduced MPC due to their cycle-oriented inductive bias.

Additionally, we compare results on the synthetic ring transfer task to empirical performance on the standard ZINC molecular property prediction task (penalized logP), where ring identification is crucial for accurate predictions. As shown in Tab. 2, trends in MPC on the synthetic ring task match trends in empirical performance on ZINC: architectures with lower ring detection complexity (GSN, FragNet, CIN) significantly outperform standard MPNNs on both tasks. This demonstrates that MPC analysis of targeted proxy tasks can provide valuable insights into real-world performance, even when exact target functions are unknown or complexity values are computationally infeasible to derive.

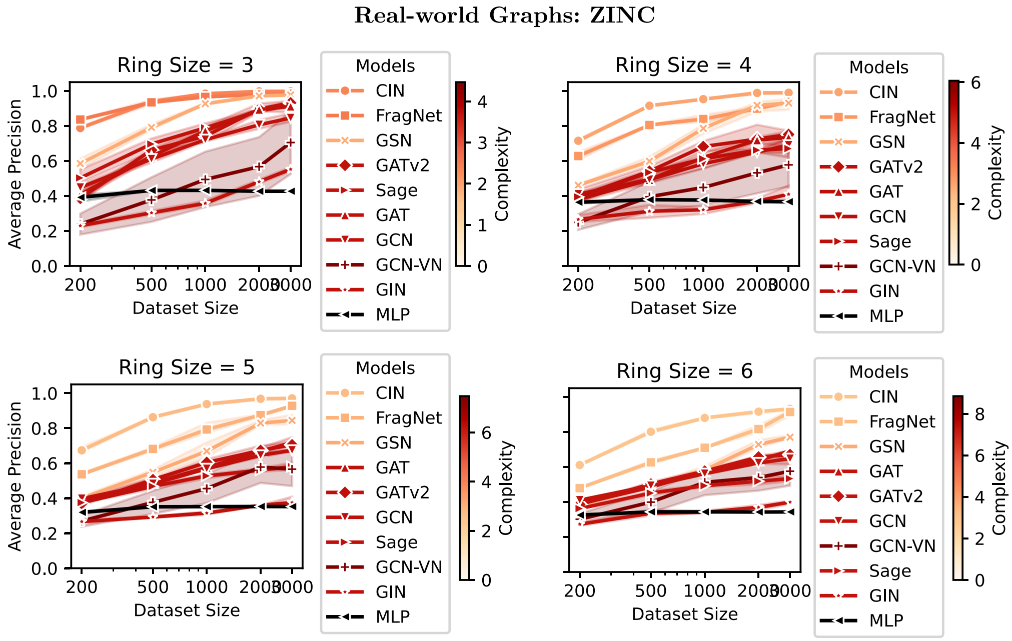


Figure 23: Test average precision compared with complexity measures MPC and WLC for the ring transfer task for real-world graphs from the ZINC dataset in relation to dataset size. MPC captures the performance advantage of the cycle-oriented GSN, CIN, and FragNet across different dataset sizes.

D.5 Simulation of Complexities

We show in Algorithm 1 for the exemplary task of propagating information from a source node u to a target node v how the complexities can be efficiently simulated using Monte Carlo simulation.

Algorithm 1 Propagating Information Simulation

```
Input: Source node $u$, target node $v$, number of trials $T$, graph $\tilde{G}$, edge weights $\boldsymbol{I}$, and maximum steps $L$

success $\leftarrow 0$
for $t = 1$ to $T$ do
    active $\leftarrow u$
    for $s = 1$ to $L$ do
        newActive $\leftarrow \emptyset$
        for $v \in$ active do
            for $u \in N_{\tilde{G}}(v) \cup \{v\}$ do
                if Random$(0, 1) < \boldsymbol{I}_{vu}$ then
                    newActive $\leftarrow$ newActive $\cup \{u\}$
                end if
            end for
        end for
        active $\leftarrow$ newActive
    end for
    if $v \in$ active then
        success $\leftarrow$ success $+ 1$
    end if
end for

Output: $-\log(\text{success}/T)$
```