On the Expressivity of Persistent Homology in Graph Learning

Rubén Ballester[1] and Bastian Rieck[2, 3]

[1]Departament de Matemàtiques i Informàtica, Universitat de Barcelona
[2]Technical University of Munich
[3]Helmholtz Munich

Abstract

Persistent homology, a technique from computational topology, has recently shown strong empirical performance in the context of graph classification. Being able to capture long range graph properties via higher-order topological features, such as cycles of arbitrary length, in combination with multi-scale topological descriptors, has improved predictive performance for data sets with prominent topological structures, such as molecules. At the same time, the *theoretical properties* of persistent homology have not been formally assessed in this context. This paper intends to bridge the gap between computational topology and graph machine learning by providing a brief introduction to persistent homology in the context of graphs, as well as a theoretical discussion and empirical analysis of its expressivity for graph learning tasks.

1 Introduction

Graph learning is a highly-active research domain in machine learning, fuelled in large parts by the *geometric deep learning* [14, 15] paradigm as well as the resurgence of new neural network architectures for handling graph data. Methods from computational topology, by contrast, have not yet been applied in this domain at large scales. Even though a large amount of prior work employs topological features to solve graph learning tasks [17, 20, 36, 37, 38, 40, 64, 80, 82, 83, 84], a formal investigation relating expressivity in graph learning and topological machine learning is still lacking.[1] We believe that this is largely a deficit in communication between the communities. This paper provides an introduction to topological methods for graph learning, while also showing new theoretical and empirical results about the *expressivity* of topological graph learning methods. Here, we understand expressivity as a general concept to signify which graph properties can be captured by a method. This includes being aware of certain substructures in graphs [21], for instance, but also being able to distinguish large classes of non-isomorphic graphs [13, 43, 69]. While graph neural networks have demonstrated substantial gains in this area, our paper focuses on topology-based algorithms, aiming to provide a better understanding of their theoretical and empirical properties.

Contributions. Our main theoretical contribution is a full characterisation of the expressivity of *persistent homology* in terms of the Weisfeiler–Leman hierarchy [54, 79]. We prove that persistent homology is *at least as expressive* as a corresponding Weisfeiler–Leman test for graph isomorphism. Moreover, we show that there exist graphs that cannot be distinguished using k-FWL, the *folklore Weisfeiler–Leman algorithm* [53], for a specific number of iterations k but that can be distinguished by persistent homology (with or without access to k-cliques in the graph). Along the way, we also prove new properties of filtrations, i.e. descriptor functions that are commonly employed to obtain topological representations of graphs, hinting at their ability to capture information about graph substructures. We complement our theoretical expressivity discussions by an experimental suite that highlights the capabilities of different filtrations for (i) distinguishing certain types of graphs, (ii) predicting characteristic graph properties, and (iii) serving as a baseline for classification tasks.

[1]A notable exception is recent work by Immonen et al. [42], which analyses the expressivity of topological methods for graph classification tasks.

Preprint. Under review.

Guide for readers. Section 2 briefly summarises the main concepts in graph learning. It should be accessible and informative to all readers. Section 2.1 and Section 2.2, by contrast, may be safely skipped by readers that are already well versed in computational topology. Section 3 outlines advantageous properties of filtrations in the context of graph learning, while Section 4 discusses the expressivity of persistent homology with respect to the Weisfeiler–Leman hierarchy of graph isomorphism tests, extending previous work [40]. Section 5 concludes the paper with an experimental suite that highlights the performance of topological methods in a variety of graph-learning tasks.

2 Background & Notation

We deal with undirected graphs in this paper. An undirected graph G is a pair $G = (V, E)$ of finite sets of n *vertices* and m *edges*, with $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$. As is common practice, we will also refer to an edge using tuple notation, with the understanding that (u, v) and (v, u) refer to the same edge. Moreover, we assume our graphs to be *simple*:[2] they must not contain any multi-edges or self-loops, i.e. edges of the form (v, v). We denote the space of all such graphs by $\mathcal{G}$. Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic* (see Fig. S.1 in the appendix), denoted by $G \simeq G'$, if there is a bijective function $\varphi\colon V \to V'$ that preserves adjacency, i.e. $(u, v) \in E$ if and only if $(\varphi(u), \varphi(v)) \in E'$. The isomorphism φ is thus preserving edges and connectivity. Since φ is bijective, it has an inverse function, which we will denote by φ^{-1}. The problem of figuring out whether two graphs are isomorphic or not is referred to as the *graph isomorphism problem*. Presently, there is no known algorithm that solves this problem in polynomial time—efficient algorithms exist only for special families of graphs [24, 44]. Hence, all subsequently-discussed graph isomorphism tests are perforce limited with respect to their expressivity, i.e. there exist classes of non-isomorphic graphs that they cannot distinguish. In the context of graph isomorphism tests, we will often require the definition of a *multiset*, which is a set whose elements are included with multiplicities. We will denote such a multiset by $\{\!\{\}\!\}$.

Equivariance. Given two graphs G and G' with n nodes, let S_n refer to the permutation group on n letters. An element $\sigma \in S_n$ acts on a graph by permuting the order of vertices, and, by transitivity, the edges. If $G \simeq G'$, there is a permutation $\sigma \in S_n$ such that $\sigma(G) = G'$. Under the assumption that all graphs have the same number of vertices n, we call a function $f\colon V \to \mathbb{R}^n$ *permutation-equivariant* if $f(\sigma(G)) = \sigma(f(G))$ for a permutation σ, with the understanding that σ acts on $\mathbb{R}^n$ by permuting the order of coordinates of the vector.[3] The output of a permutation-equivariant function thus changes with the permutation in a predictable manner.

2.1 Topological Features of Graphs

The simplest kind of topological features to prescribe to graphs are *connected components* and *cycles*. For planar input graphs, i.e. for graphs that can be embedded in the plane such that there are no overlaps between edges, Euler's formula captures structural properties via $V - E + F = 2$, where F counts the number of faces (including the unbounded one) of the planar embedding. While this formula can be used to comment on the feasibility of some problems in graph theory (see Appendix A), its practical utility is limited. However, it turns out that a similar identity relates connected components and cycles in a graph. Formally, referring to the number of connected components as β_0 and the number of cycles as β_1, we have $\beta_1 = m + \beta_0 - n$, where n and m denote the number of vertices and edges, respectively. The two quantities in this formula are also known as the first two *Betti numbers* of a graph, with β_1 also known as the *cyclomatic number* or the *circuit rank* [9, pp. 27–30]. While the expressive power of these two numbers is limited, it can be improved by evaluating them alongside a *filtration*, i.e. a sequence of nested subgraphs of the form $\emptyset \subseteq G_0 \subseteq G_1 \ldots \subseteq G_{k-1} \subseteq G_k = G$. Filtrations typically arise from scalar-valued functions of the form $f\colon G \to \mathbb{R}$, which assign vertices and edges a value. Changes in the Betti numbers can then be tracked over the course of the filtration: given a threshold $a \in \mathbb{R}$, one analyses the subgraph arising from the *pre-image* of $(-\infty, a]$ of f, denoted $f^{-1}((-\infty, a])$. This leads to *persistent Betti numbers*, which are typically summarised in a *persistence diagram*, i.e. a topological descriptor, consisting of tuples $(a_i, a_j) \in \mathbb{R}^2$, with a_i referring to the value at which a topological feature was *created*,

[2]This is primarily an assumption for notational convenience. The concepts described in this paper can also be extended to more complex scenarios.

[3]We only consider functions operating on vertices, but the concept can extended to edges as well.

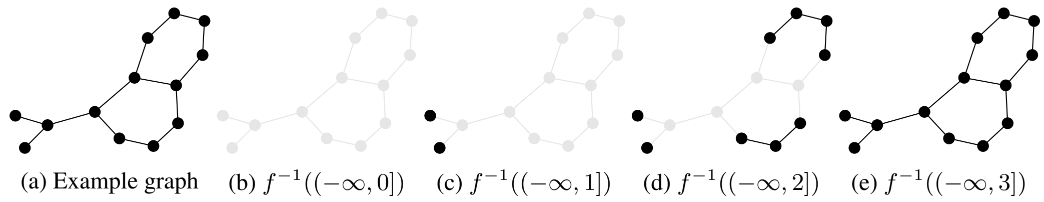

(a) Example graph
(b) $f^{-1}((-\infty, 0])$
(c) $f^{-1}((-\infty, 1])$
(d) $f^{-1}((-\infty, 2])$
(e) $f^{-1}((-\infty, 3])$

FIGURE 1: An example graph and three different steps of a degree-based filtration. The respective caption indicates the pre-image of the corresponding filtration function.

and a_j referring to the value at which a topological feature was destroyed (for instance, because two connected components are being merged). The absolute difference in function values $|a_j - a_i|$ is called the *persistence* of a topological feature; it indicates the prominence or relevance of said feature.

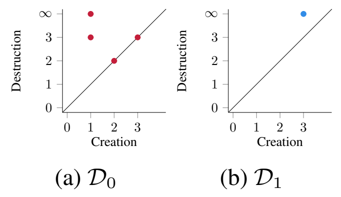


FIGURE 2: Persistence diagrams of the filtration depicted in Fig. 1. Points can have different multiplicities; we show *essential features*, i.e. topological features that persist over the full filtration using an ∞ symbol.

Fig. 1 depicts an example filtration for a simple graph, where we use the degree of each vertex to filter the graph. The calculation of persistent homology along this filtration involves counting the connected components and cycles. We note that these features can only change whenever the filtration function changes. The *critical points* of the degree filtration are thus the unique degrees of vertices occurring in the graph. Fig. 2 shows the persistence diagrams arising from the filtration. The more complex structure of persistence diagrams (in comparison to the simple Betti numbers) already hints at their capabilities in providing expressive graph descriptors.

2.2 Topological Features of Simplicial Complexes

The concepts discussed in Section 2.1 generalise to higher dimensions as well, with the understanding that the connectivity of higher-order structures of a graph—cliques—is being modelled. The resulting framework is referred to as *persistent homology*; it encompasses the calculations on graphs, extending them to simplicial complexes or other types of combinatorial complexes. The nomenclature is used to reflect the idea that the underlying graphs are originating from manifolds, of which the filtration function f measures certain features. A feature of high persistence is thus to be understood as a feature with that occurs over a large range of values of f. In the context of graphs, there are different constructions for obtaining simplicial complexes; we will subsequently use *clique complexes* (in which a k-clique is represented by a $(k-1)$-simplex) because it is (i) straightforward to implement, and (ii) cliques are known to be characteristic graph structures [13]. Persistent homology is typically calculated using matrix reduction reduction algorithms, whose optimisation is still an ongoing topic of research [7]. We refer readers to Otter et al. [59] for a comprehensive introduction of computational strategies. In the general point cloud setting, a paper by Bauer [6] provides a highly-efficient reference implementation while also stating details on combinatorial optimisation strategies. Readers are invited to read Appendix C for a more detailed exposition of concepts in computational topology.

3 Properties of Filtrations

Before we discuss how to create specific filtrations that are useful for graph learning tasks, we first discuss some of their general properties. We briefly expand on the *stability properties* of filtrations, a crucial aspect of research in computational topology [71]. Given two filtrations f, g of the same graph, a seminal result by Cohen-Steiner et al. [22] proves the following bound.

Theorem 1 (Bottleneck stability). *Let f, g refer to filtrations of a graph G, and let $\mathcal{D}_f$ and $\mathcal{D}_g$ denote their respective persistence diagrams. The* bottleneck distance *distance is upper-bounded by* $\mathrm{d_B}(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$, *where* $\|\cdot\|_\infty$ *refers to the supremum norm.*

An extension of this theorem, with $\mathrm{d_B}$ being replaced by the Wasserstein distance, shows that persistence diagrams are also stable in the Lipschitz sense [23]. These stability properties are remarkable because they link a topological quantity with a geometrical one, thus underscoring how persistent homology itself incorporates both geometrical and topological aspects of input data. We continue our enumeration of filtration properties by proving that *any* filtration induced by an equivariant function is well-behaved under graph isomorphism.

Theorem 2. *Let $G \simeq G'$ be two isomorphic graphs, and φ be the respective isomorphism. For any equivariant filtration f, the corresponding persistence diagrams are equal.*

Theorem 2 is a consequence of the more general principle of *functoriality*. It also shows that it is impossible to 'adversarially' pick an equivariant filtration function that leads to a non-zero topological dissimilarity between two non-isomorphic graphs. Dropping this condition incurs a substantial loss of expressive power [42]. The theorem thus guarantees that persistent homology is compatible with equivariant function learning, pointing towards the utility of hybrid models that leverage different types of structural properties of graphs. To finish this discussion of general properties, we remark that the calculation of persistent homology carries information about the *diameter* (the length of the longest shortest path) and *girth* (the length of the shortest cycle) of a graph. While the theoretical upper bounds are not tight (see Appendix D for more details), our empirical analysis in Section 5.3 shows that persistent homology captures more than 'just' topological information about a graph. Our work thus corroborates recent results in the setting of point clouds, where persistence diagrams permit inferring additional properties about the input data [16, 48, 73].

4 The Weisfeiler–Leman Hierarchy

Having discussed the properties of specific filtrations, we now analyse the expressivity of persistent homology in the context of the Weisfeiler–Leman hierarchy of graph isomorphism tests. 1-WL, also known as *colour refinement*, constitutes a simple method for addressing the graph isomorphism problem. It is the backbone of graph expressivity research; readers are referred to Morris et al. [53] for a comprehensive survey of 1-WL and its higher-order variants. We follow the terminology of this article and briefly introduce all relevant concepts. Formally, 1-WL proceeds by iteratively calculating a node colouring function $C_i^{(1)} \colon V \to \mathbb{N}$. The output of this function depends on the neighbours of a given node. For a vertex v at iteration $i > 0$, we have

$$C_i^{(1)}(v) := \mathtt{RELABEL}\left(\left(C_{i-1}^{(1)}(v), \left\{\!\!\left\{C_i^{(1)}(u) \mid u \in \mathcal{N}(v)\right\}\!\!\right\}\right)\right), \tag{1}$$

where RELABEL refers to an injective function that maps the tuple of colours to a unique colour, i.e. a unique number. The algorithm is initialised by either using existing labels or the degree of vertices.[4] After a finite number of steps, the colour assignments generated using Eq. (1) stabilise. If two graphs give rise to different colour sequences, the graphs are guaranteed to be non-isomorphic. The 1-WL test is computationally easy and constitutes an upper bound for the expressivity of many graph neural network (GNN) architectures [54, 79]. In other words, if 1-WL cannot distinguish two non-isomorphic graphs, GNNs will also not be able to distinguish them. It is already a known result that *any* 1-WL colouring can be reproduced by creating a special filtration [40]; we restate this result as Theorem 4 in the appendix. The implication is that persistent homology is *at least as expressive* as 1-WL because there is a filtration that distinguishes all the graphs 1-WL can distinguish. Appendix E shows examples of graphs that can *only* be distinguished via their topological features, thus proving that a topological perspective is strictly more expressive than 1-WL. Since 1-WL is also unable to distinguish between graphs with different triangle counts or graphs with cycle information, it was generalised to include information about labelling *tuples* of k nodes (as opposed to only labelling a single node), leading to a hierarchy of algorithms. The variant we shall subsequently describe is also known as the *folklore Weisfeiler–Leman algorithm* [53]. It can be shown that there are non-isomorphic graphs that cannot be distinguished by k-FWL, but that can be distinguished by $(k+1)$-FWL.[5] Following Morris et al. [53], k-FWL is based on the idea of assigning colours to

[4]Note that Eq. (1) does not recognise an ordering of labels. Initialising 1-WL with a constant value thus leads to the *same* colouring—up to renaming—after the first iteration.

[5]There are also other variants, for instance the *oblivious Weisfeiler–Leman algorithm*. It slightly differs in the way tuples are being relabelled, but a paper by Grohe [34] shows that the variant is essentially as powerful as k-FWL (with a minor shift in indices). The reader is referred to Morris et al. [53] and the references therein for an extended discussion of these aspects.

subgraphs as opposed to assigning colours to *vertices*. To achieve this, k-FWL operates on k-tuples of vertices; for iteration $i = 0$, two tuples $\mathbf{v} = (v_1, \ldots, v_k)$ and $\mathbf{w} = (w_1, \ldots, w_k)$ are assigned the same colour if the map $v_j \mapsto w_j$ induces a graph isomorphism between the subgraphs induced by $\mathbf{v}$ and $\mathbf{w}$, respectively. For subsequent iterations with $i > 0$, we relabel the tuples similar to 1-WL, i.e.

$$C_i^{(k)}(\mathbf{v}) := \texttt{RELABEL}\left(\left(C_{i-1}^{(k)}(\mathbf{v}), \left\{\!\!\left\{ C_i^{(k)}(\phi_1(\mathbf{v}, u)), \ldots, C_i^{(k)}(\phi_k(\mathbf{v}, u)) \mid u \in \mathcal{N}(v) \right\}\!\!\right\}\right)\right), \quad (2)$$

where $\phi_j(\mathbf{v}, u) := (v_1, \ldots, v_{j-1}, u, v_{j+1}, \ldots, v_k)$ refers to the function that replaces the jth element of the k-tuple $\mathbf{v}$ with u. This induces a neighbourhood relation between tuples and just as in the case of 1-WL, we run the algorithm until the assigned colours of tuples stabilise for one graph. Similarly, if the colour sequences of two graphs differ, the graphs are non-isomorphic. As a generalisation of previous work [40], we can show that *any* k-FWL colouring can be reproduced with a specific filtration, thus proving that persistent homology is *at least* as expressive as k-FWL.

Theorem 3. $^{(*)}$ *Given k-FWL colourings of two graphs G and G' that are different, there exists a filtration of G and G' such that the corresponding persistence diagrams in dimension $k-1$ or dimension k are different.*

Theorem 3 does not result in a *constructive proof* since the higher-order setting is more complex. In contrast to the result for 1-WL, we also see that Theorem 3 needs to make use of *two* types of persistence diagrams. Moreover, while the filtration used in the proof does not necessarily lead to differentiable persistence diagrams (because of non-unique function values), it is possible to rectify this in practice (see Lemma 2 in the Appendix). Both Theorem 4 and Theorem 3 may not be completely satisfactory because they only show the *existence* of such a filtration, but make no claims about the expressivity of existing filtrations. We aim to provide a more learning-theoretic approach of this issue in future work. Moreover, we would ideally want to extend Theorem 3 to state that persistent homology is *strictly* more expressive than k-FWL. This is not as straightforward as for $k = 1$, since we would have to construct families of non-isomorphic graphs that require $(k+1)$-FWL to be distinguished but that can be distinguished already with lower-dimensional persistent homology. Currently, we can provide one such counterexample, described in Table 1, consisting of the 4×4 rook's graph and the Shrikhande graph. With an appropriate filtration, persistent homology can distinguish these two graphs *without* requiring more than vertices and edges, whereas 2-FWL is unable to distinguish them. We leave a more general result for future work.

5 Experiments

The previous sections discussed the theoretical properties of filtrations. Here, we analyse their *empirical* performance. We first analyse the *expressivity* of five different filtrations by letting them distinguish non-isomorphic graphs. This is followed by experiments on graph property prediction. Please refer to Appendices F and G for additional expressivity and graph-classification experiments.

Experimental setup. We use different data sets containing strongly-regular graphs [52], minimal Cayley graphs, as well as benchmark data sets for graph-learning tasks [78, BREC] In the following, we will use five different filtrations for each graph:

1. A *degree* filtration (denoted by **D**), i.e. $v \mapsto \deg(v)$. The degree filtration is the most basic non-trivial filtration of a graph, showing nevertheless surprising empirical performance in graph classification tasks [37, 57, 66].
2. A filtration based on the eigenvalues of the *undirected graph Laplacian* (denoted by **L**), i.e. $v \mapsto \lambda_v$, where λ_v indicates the eigenvalue of the undirected graph Laplacian corresponding to vertex v. The graph Laplacian is known to capture characteristic properties of a graph; in the context of persistent homology it is often used in the form of a *heat kernel signature* [17].
3. A filtration based on the *Ollivier–Ricci curvature* [58] (denoted by O) in the graph, setting $v \mapsto -1$ and $(u, v) \mapsto \kappa(u, v)$, with κ denoting the Ollivier–Ricci curvature
$$\kappa(u, v) := 1 - \mathrm{W}_1(\mu_u^\alpha, \mu_v^\alpha), \quad (3)$$
where W_1 denotes the first Wasserstein distance,[6] and $\mu_u^\alpha, \mu_v^\alpha$ denote *probability measures* based on a lazy random walk in the graph, i.e. $\mu_u^\alpha(u) := \alpha$, indicating the probability of staying at the same vertex, $\mu_u^\alpha(v) := (1-\alpha)^1/_{\deg(u)}$ for a neighbour v of u, and $\mu_u^\alpha(\cdot) := 0$ otherwise. The

[6]This metric is also known as the *Earth Mover's Distance* [47]. The Wasserstein distance is a fundamental concept in optimal transport; the monograph by Villani [76] contains a comprehensive introduction to this topic.

TABLE 1: Success rate (↑) for distinguishing pairs of *strongly-regular graphs* when using different filtrations at varying expansion levels of the graph (denoted by k). 2-FWL cannot distinguish between any of these pairs [10].

Data	$k=1$					$k=2$					$k=3$				
	Filtration														
	D	O	F	L	V	D	O	F	L	V	D	O	F	L	V
16622	0.00	**1.00**	0.00	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
251256	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.90**	**0.90**	**0.90**	**0.90**	**0.90**
261034	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.20	0.76	0.20	0.93	0.93	0.93	**0.98**	0.93
281264	0.00	0.83	0.00	0.00	0.00	0.00	**1.00**	0.00	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
291467	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.77**	**0.77**	**0.77**	**0.77**	**0.77**
351668	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.95	0.95	0.95	**0.99**	0.95
351899	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.81**	**0.81**	**0.81**	**0.81**	**0.81**
361446	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.02	0.83	0.02	0.92	0.92	0.92	**0.99**	0.92
401224	0.00	0.00	0.00	0.00	0.00	0.93	0.93	0.93	0.99	0.93	0.94	0.94	0.94	**0.99**	0.94

probability measures in Eq. (3) μ_u^α may be adjusted; recent work investigates the utility of this perspective [26, 72]. We set $\alpha = 0$ for our subsequent experiments, thus obtaining a non-lazy random walk, and leave the investigation of the impact of other values for future work.

4. A filtration based on the *augmented Forman–Ricci curvature* [68] (denoted by **F**), where we again set $v \mapsto -1$ and $(u, v) \mapsto \mathcal{F}(u, v)$, with $\mathcal{F}(u, v) := 4 - \deg(u) - \deg(v) + 3\,|\mathcal{N}(u) \cap \mathcal{N}(v)|$.
5. A *Vietoris–Rips* filtration (denoted by **V**) over the metric space defined by the shortest-path distance between its nodes [1].

After picking a filtration (except for the Vietoris–Rips one), we expand the graph by filling in all $(k+1)$-cliques, with filtration value for a clique σ given recursively by the maximum filtration value of its proper subcliques, i.e. $\sigma \mapsto \max_{\tau \subsetneq \sigma} f(\tau)$, and calculating persistent homology up to dimension k. Hence, for $k = 1$, we leave the graph 'as-is,' making use of connected components and cycles only.[7] Our persistent homology calculations result in a set of persistence diagrams for each graph, which we compare in a pairwise manner using the bottleneck distance described by Eq. (12). We consider two graphs to be different whenever the distance between their persistence diagrams is $> 1 \times 10^{-8}$, i.e. above machine precision. This setup has the advantage that no additional classifier is required; when dealing with data sets of pairs of non-isomorphic graphs, we may thus simple *count* the number of non-zero distance pairs, showing the utility of persistent homology as a powerful baseline for graph expressivity analysis.

5.1 Strongly-Regular Graphs and Minimal Cayley Graphs

We start our investigation by analysing *strongly-regular graphs*, which are are known to be extremely challenging to distinguish. 2-FWL, for instance, cannot distinguish *any* of these graphs [53]. Table 1 summarises the performance of our selected filtrations. We first observe that for $k = 1$, i.e. for the original graph without any cliques, few pairs of graphs can be distinguished by the five filtrations. Notably, a curvature-based filtration *is* sufficient to distinguish the two graphs in the 16622 data set, colloquially known as the 4×4 rook's graph and the Shrikhande graph. Distinguishing between these two graphs is usually said to require knowledge about cliques [10], but it turns out that a suitable filtration is sufficient. However, the empirical expressivity of curvature-based filtrations appears limited for $k = 1$, improving only for higher-order clique complexes. The Laplacian filtration, by contrast, exhibits strong empirical performance for $k = 2$ on almost half of the data sets, increasing to near-perfect performance for $k = 3$ in almost all data sets. Vietoris–Rips and degree filtrations obtain the exact same success rates for every k and data set, exhibiting the lowest success rates for $k = 1$ and $k = 2$, and the same ones to the curvature filtrations for $k = 3$. It is clear that knowledge about higher-order cliques helps in driving performance here. Notice that in contrast to other algorithms [10], no additional embedding of the graphs is required; we are comparing 'raw' persistence diagrams directly.

[7] Notice the shift in dimension: a k-simplex has $k+1$ vertices, meaning that persistent homology in dimension k contains information about $(k+1)$-cliques.

TABLE 2: Success rate (↑) for distinguishing pairs of *minimal Cayley graphs* when using five different filtrations at varying expansion levels of the graph (denoted by k). Values for 1-WL are shown as a baseline.

Data		$k=1$					$k=2$				
		Filtration									
	1-WL	D	O	F	L	V	D	O	F	L	V
`cay12`	0.67	0.67	0.71	0.95	0.86	0.00	0.95	0.95	0.95	**1.00**	0.95
`cay16`	0.83	0.83	0.42	0.83	0.58	0.00	0.83	0.92	0.83	**1.00**	0.94
`cay20`	0.61	0.61	0.46	0.61	0.79	0.00	0.61	0.79	0.61	**1.00**	0.89
`cay24`	0.65	0.65	0.82	0.86	0.98	0.00	0.83	0.93	0.86	**1.00**	0.93
`cay32`	0.76	0.76	0.81	0.76	0.90	0.00	0.76	0.94	0.76	**1.00**	0.90
`cay36`	0.69	0.69	0.87	0.84	0.99	0.00	0.84	0.95	0.84	**1.00**	0.94
`cay60`	0.69	0.69	0.90	0.78	1.00	0.00	0.77	0.95	0.78	**1.00**	0.97
`cay63`	0.49	0.49	0.89	0.73	0.88	0.00	0.73	0.93	0.73	**1.00**	0.96

TABLE 3: Success rate (↑) for distinguishing pairs of instances of the *BREC data set* when using different filtrations at varying expansion levels of the graph (denoted k). Due to combinatorial constraints, we did not calculate the Vietoris–Rips filtration for $k=4$. Legend and number of graphs per category: `B` (Basic, 60), `R` (Regular, 100), `E` (Extension, 100), `C` (CFI, 100), `4, 20` (4-Vertex Condition), `D` (Distance-Regular, 20) graphs, respectively and `A` (average over full data set, 400 graphs).

Data	$k=1$					$k=2$					$k=3$					$k=4$			
	Filtration																		
	D	O	F	L	V	D	O	F	L	V	D	O	F	L	V	D	O	F	L
`B`	0.03	0.93	0.87	**1.00**	0.00	0.78	**1.00**	0.98	**1.00**	0.52	0.83	**1.00**	0.98	**1.00**	0.58	0.83	**1.00**	0.98	**1.00**
`R`	0.00	0.42	0.32	0.00	0.00	0.39	0.54	0.50	0.48	0.39	0.85	0.93	0.91	0.93	0.85	0.89	**0.97**	0.95	**0.97**
`E`	0.07	0.76	0.44	0.94	0.00	0.26	0.92	0.59	**1.00**	0.11	0.29	0.92	0.59	**1.00**	0.16	0.29	0.92	0.59	**1.00**
`C`	0.03	0.03	0.03	**0.06**	0.03	0.03	0.03	0.03	**0.06**	0.03	0.03	0.03	0.03	**0.06**	0.03	0.03	0.03	0.03	0.06
`4`	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
`D`	0.00	0.00	0.00	**0.05**	0.00	0.00	0.00	0.00	**0.05**	0.00	0.00	0.00	0.00	**0.05**	**0.05**	0.00	0.00	0.00	0.05
`A`	0.03	0.44	0.33	0.40	0.01	0.29	0.52	0.43	0.54	0.21	0.47	0.67	0.58	0.70	0.40	0.48	0.68	0.59	**0.71**

As an additional class of complex graphs, we analyse *minimal Cayley graphs*, i.e. Cayley graphs that encode a group with a minimal generating set. Minimal Cayley graphs are still a topic of active research in graph theory, with several conjectures yet to be proven [4, 5]. We follow the same experimental setup as described above but also show the performance of 1-WL, calculated via a subtree Weisfeiler–Leman graph kernel [70]. Table 2 shows the results. We observe that the Laplacian filtration is trivially able to distinguish between all these graphs for $k = 2$ and has a strong performance for $k = 1$; this is not surprising since spectra of the graphs are known to be characteristic [49]. The performance of the curvature filtrations also points towards the utility of this formulation in practice. We also observe that Vietoris–Rips filtrations are the ones that most benefit from the availability of higher-order information, with a significant improvement in performance, going from a 0% success rate for $k = 1$ to $> 90\%$ for $k = 2$.

5.2 BREC Data Set

BREC [78] is a novel graph expressivity data set focused on providing a robust, challenging benchmark for graph isomorphism detection, containing particularly difficult graph classes. The data set consists of 400 graphs, divided into 6 different categories, Table 3 provides an overview of them and shows our experimentally-observed success rates. We observe that the Vietoris–Rips filtration is the worst performing filtration for all tested k values, followed by the degree filtration. The most informative curvature-based filtration is the Ollivier–Ricci one, which obtained higher or equal success rates than the Forman–Ricci curvature filtration in all the subsets of data, with strictly higher average success rates for all k values. The most effective filtration overall is the Laplacian filtration for $k = 4$, surpassing almost all the algorithms, including both graph neural networks and classical

TABLE 4: Accuracy (↑) when predicting the properties of graphs in the `ogbg-molhiv` molecular graph data set [41] using different filtrations at varying expansion levels of the graph (denoted by k). Column R contains the average probability of successfully predicting a property at random over all possible values of the property in the data, with the probability of choosing a label being proportional to the number of graphs with that label.

Data		$k = 1$					$k = 2$				
		Filtration									
	R	D	O	F	L	V	D	O	F	L	V
Diameter	0.02	0.09	**0.11**	0.05	0.08	0.07	0.10	0.08	0.06	0.09	-
Girth	0.04	0.00	0.11	0.34	0.46	**0.48**	0.14	0.21	0.33	0.45	-
Radius	0.03	0.16	**0.21**	0.06	0.15	0.14	0.20	0.18	0.10	0.17	-

methods, described in Wang and Zhang [78, Table 2]. The only exception was the N_2 algorithm [61] that obtained a success rate of 74.5%, compared to the 71% obtained by the Laplacian filtration for $k = 4$. Given the fact that N_2 requires knowledge of the isomorphism class of *all* 2-hop-induced subgraphs of a graph, the computational complexity makes it infeasible to apply for many graph sizes in practices [61], whereas the Laplacian filtration remains computable. Overall, these results experiments underscore the high expressivity of persistent homology, making it a strong baseline for graph-learning tasks. Please refer to Appendix F.2 for additional results.

5.3 Predicting Graph Properties

As our final expressivity experiments, we assess the capability of persistent homology to predict graph properties. Here, we focus on the *diameter*, the *radius*, and the *girth*.

Predicting the diameter of random graphs. We predict the diameter of Erdős–Rényi and Watts–Strogatz graphs. For the Erdős–Rényi graphs, we generate $N = 100$ graphs with $n = 100$ vertices and $p = 0.1$. This edge probability corresponds to the critical connectivity regime, for which closed-form solutions of the diameter distribution are not readily available [35]. We assess the utility of persistent homology by specifying a regression task;[8] to this end, we vectorise the persistence diagrams for each filtration using *Betti curves* [57, 66], a simple curve-based topological representation. While this representation is technically a function, we represent it as a histogram of 10 bins and train a *ridge regression classifier* via leave-one-out cross-validation to predict the diameter of each graph. We deliberately focus only on zero-dimensional and one-dimensional persistent homology, i.e. we leave $k = 1$. Using the *mean absolute error* (MAE) for evaluation, we find that Ollivier–Ricci curvature performs best (0.057), followed by the Laplacian spectrum (0.061), and the degree filtration (0.065). We observe similar patterns when calculating $N = 100$ Watts–Strogatz graphs of type $(100, 5, 0.1)$, i.e. we keep the same number of vertices and the same edge rewiring probability p, but connect each node with its 5 nearest neighbours in a ring neighbourhood. Using the same classifier, we again find that Ollivier–Ricci curvature achieves the lowest MAE (0.851), followed by the degree filtration (0.865), and the Laplacian spectrum (1.072).

Predicting graph properties of the `ogbg-molhiv` [41] data set. We predict the maximum radius and diameter among the radii and diameters of the connected components of each graph as well as their girth. See Fig. S.4 for a visualisation of the distribution of these properties across all graphs in the data set. We use a random forest regression model and *persistence images* [2] as its input, computed from the persistence diagrams of the graphs calculated as in the previous sections. The model is trained on the usual train and test splits of the data set, with the final prediction obtained by rounding to the nearest integer.[9] Overall, this is a challenging task, with 24, 15, and 5 labels having more than 100 examples for the diameter, radius and girth, respectively. Thus, the average probabilities of guessing the correct label by random choice is 0.02, 0.03, and 0.04. However, we find that persistent homology, with a suitable filtration, can predict the maximum radius, the maximum diameter, and the girth of the graphs on unseen examples (i.e. on the test data set) in approximately 11%, 20%,

[8] Following Hartmann and Mézard [35], we take the diameter of an Erdős–Rényi graph, which might consist of different connected components, to be the largest diameter of all connected components of the graph.

[9] We predict a girth of ∞ in case the output value is larger than the number of nodes in the graph.

and 48% of the cases, respectively, exhibiting substantially-improved results over a random baseline. For predicting radii and diameters, we find that the Ollivier–Ricci curvature outperforms the other filtrations for $k = 1$ but gets worse for $k = 2$, where the degree filtration performs the best. It is surprising that the Vietoris–Rips filtration is *not* able to predict the radii of the graphs in this data set accurately ($\approx 14\%$ accuracy), being the only filtration that works with explicit distances. However, this filtration proves most effective for predicting the girth, having an accuracy of almost 50% as compared to the 4% of the baseline. These results complement prior work on predicting properties of point clouds [16, 73], showing that topology-based graph-learning approaches carry a large degree of additional information about graphs.

6 Discussion

We discussed various aspects of the computation and provided evidence of the advantageous properties of persistent homology in the context of graph learning. Our primary theoretical insight is that persistent homology is *at least as expressive* as a corresponding 1-WL or k-FWL test, in some cases surpassing their discriminative power.[10] Experiments underscore these theoretical expressivity properties, while also demonstrating that persistent homology is able to capture additional properties of a graph. An inherent limitation of this method is that it crucially hinges on the choice of the *filtration function*, which is used to assess topological features of the graph. In light of the performance differences in our experiments, we suggest that future work should focus on elucidating properties of classes of such functions, aiming to strike a balance between expressivity and efficiency. Another limitation involves the calculation *per se*, which requires costly clique-finding operations and is thus not scalable to large, dense graphs. If node features are present, alternative geometrical-topological approaches could potentially be used [50, 51, 55, 67, 75], necessitating additional research.

In the future, we would like to formally prove which classes of filtrations make k-dimensional persistent homology strictly more expressive than k-FWL (if any). Follow-up research could also focus on identifying other properties (next to the diameter and girth) that can be captured by persistent homology in graph learning. Such research has both theoretical and empirical components; a first step would be a formalisation of which substructures are captured by models [13, 21, 72]. Moreover, the success of the Laplacian filtration at the experimental tasks may hint at new filtrations based on spectral graph theory that provide a trade-off between utility and computational efficiency. We find that this research direction is overlooked by the computational topology research community, with most of the expressivity/stability results focusing on describing the stability of distance-based filtrations under perturbations [18, 19], and few works focusing on graphs [6]. Based on our experiments, we thus envision that persistent homology will constitute a strong baseline for graph-learning applications. As previous work shows, even topology-inspired approaches, making use of concepts such as filtrations, can approximate the performance of highly-parametrised models at a fraction of the computational cost [57]. All insights obtained using such topological methods hint at the overall utility of graph-structural information for graph learning tasks, but it is not clear whether current graph benchmark data actually exhibit such structures [60]. We thus hope that persistent homology and related techniques will also find more applications in *hybrid models*, which are able to incorporate geometrical–topological information about graphs. This is an emerging research topic of crucial relevance since there are now numerous graph data sets that combine geometrical information (node coordinates) with topological information [43].

Our theoretical analysis of the properties of persistent homology for graph learning tasks show the potential and benefits of a topology-based perspective. We are confident that additional computational topology concepts will enrich and augment machine learning models, leading to new insights about their theoretical and empirical capabilities. This paper is but a first attempt at elucidating the theoretical utility of computational topology in a graph learning context; advancing the field will require many more insights.

[10] More specifically, previous work [40] showed that persistent homology is *strictly more expressive* than 1-WL. In this article, when tackling $k \geq 2$, we only provide counterexamples that show that there are graphs that can be distinguished by *some* filtration but not by 2-FWL, for instance.

Acknowledgements

This paper was motivated in large parts by discussions with participants of the BIRS 2022 Workshop on ‘Deep Exploration of Non-Euclidean Data with Geometric and Topological Representation Learning.’ The author is also indebted to Leslie O’Bray for comments and discussions that helped substantially improve the main arguments and led to a simplified proof of Theorem 3.

References

[1] H. Adams and B. Coskunuzer. ‘Geometric Approaches to Persistent Homology’. In: *SIAM Journal on Applied Algebra and Geometry* 6.4 (2022), pp. 685–710. DOI: 10.1137/21M1422914.

[2] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier. ‘Persistence Images: A Stable Vector Representation of Persistent Homology’. In: *Journal of Machine Learning Research* 18.8 (2017), pp. 1–35. URL: http://jmlr.org/papers/v18/16-337.html.

[3] V. Arvind, J. Köbler, G. Rattan, and O. Verbitsky. ‘On the Power of Color Refinement’. In: *Fundamentals of Computation Theory*. Ed. by A. Kosowski and I. Walukiewicz. Cham, Switzerland: Springer, 2015, pp. 339–350.

[4] L. Babai. ‘Automorphism Groups, Isomorphism, Reconstruction’. In: *Handbook of Combinatorics*. MIT Press, 1996, pp. 1447–1540.

[5] L. Babai. ‘Chromatic number and subgraphs of Cayley graphs’. In: *Theory and Applications of Graphs*. Ed. by Y. Alavi and D. R. Lick. 1978, pp. 10–22.

[6] U. Bauer. ‘Ripser: efficient computation of Vietoris–Rips persistence barcodes’. In: *Journal of Applied and Computational Topology* 5.3 (2021), pp. 391–423. DOI: 10.1007/s41468-021-00071-5.

[7] U. Bauer, T. B. Masood, B. Giunti, G. Houry, M. Kerber, and A. Rathod. ‘Keeping it sparse: Computing Persistent Homology revisited’. 2022. arXiv: 2211.09075 [cs.CG].

[8] P. Bendich, J. S. Marron, E. Miller, A. Pieloch, and S. Skwerer. ‘Persistent homology analysis of brain artery trees’. In: *The Annals of Applied Statistics* 10.1 (2016), pp. 198–218. DOI: 10.1214/15-AOAS886.

[9] C. Berge. *The Theory of Graphs*. Dover Publications, 2001.

[10] C. Bodnar, F. Frasca, Y. Wang, N. Otter, G. F. Montufar, P. Lió, and M. Bronstein. ‘Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks’. In: *International Conference on Machine Learning (ICML)*. Ed. by M. Meila and T. Zhang. Proceedings of Machine Learning Research 139. PMLR, 2021, pp. 1026–1037.

[11] B. Bollobás. ‘Distinguishing Vertices of Random Graphs’. In: *Graph Theory*. Ed. by B. Bollobás. North–Holland Mathematics Studies 62. Amsterdam, Netherlands: North–Holland, 1982, pp. 33–49. DOI: 10.1016/S0304-0208(08)73545-X.

[12] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. ‘Protein function prediction via graph kernels’. In: *Bioinformatics* 21.suppl 1 (June 2005), pp. i47–i56. DOI: 10.1093/bioinformatics/bti1007.

[13] G. Bouritsas, F. Frasca, S. Zafeiriou, and M. M. Bronstein. ‘Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2023), pp. 657–668. DOI: 10.1109/TPAMI.2022.3154319.

[14] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. ‘Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges’. 2021. arXiv: 2104.13478 [cs.LG].

[15] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. ‘Geometric Deep Learning: Going beyond Euclidean data’. In: *IEEE Signal Processing Magazine* 34.4 (2017), pp. 18–42. DOI: 10.1109/MSP.2017.2693418.

[16] P. Bubenik, M. Hull, D. Patel, and B. Whittle. ‘Persistent homology detects curvature’. In: *Inverse Problems* 36.2 (2020), p. 025008. DOI: 10.1088/1361-6420/ab4ac0.

[17] M. Carrière, F. Chazal, Y. Ike, T. Lacombe, M. Royer, and Y. Umeda. ‘PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures’. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by S. Chiappa and R. Calandra. Proceedings of Machine Learning Research 108. PMLR, 2020, pp. 2786–2796.

[18] F. Chazal, V. De Silva, M. Glisse, and S. Oudot. *The structure and stability of persistence modules*. Vol. 10. SpringerBriefs in Mathematics. Cham, Switzerland: Springer, 2016. DOI: 10.1007/978-3-319-42545-0.

[19] F. Chazal, V. de Silva, and S. Oudot. 'Persistence stability for geometric complexes'. In: *Geometriae Dedicata* 173.1 (2014), pp. 193–214. DOI: 10.1007/s10711-013-9937-z.

[20] Y. Chen, B. Coskunuzer, and Y. Gel. 'Topological Relational Learning on Graphs'. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 27029–27042.

[21] Z. Chen, L. Chen, S. Villar, and J. Bruna. 'Can Graph Neural Networks Count Substructures?' In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 10383–10395.

[22] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. 'Stability of persistence diagrams'. In: *Discrete & Computational Geometry* 37.1 (2007), pp. 103–120. DOI: 10.1007/s00454-006-1276-5.

[23] D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and Y. Mileyko. 'Lipschitz functions have L_p-stable persistence'. In: *Foundations of Computational Mathematics* 10.2 (2010), pp. 127–139. DOI: 10.1007/s10208-010-9060-6.

[24] C. J. Colbourn. 'On testing isomorphism of permutation graphs'. In: *Networks* 11.1 (1981), pp. 13–21. DOI: 10.1002/net.3230110103.

[25] K. Coolsaet, S. D'hondt, and J. Goedgebeur. 'House of Graphs 2.0: A database of interesting graphs and more'. In: *Discrete Applied Mathematics* 325 (2023), pp. 97–107. DOI: 10.1016/j.dam.2022.10.013. URL: https://houseofgraphs.org.

[26] C. Coupette, S. Dalleiger, and B. Rieck. 'Ollivier–Ricci Curvature for Hypergraphs: A Unified Framework'. In: *International Conference on Learning Representations*. 2023. arXiv: 2210.12048 [cs.LG]. URL: https://openreview.net/forum?id=sPCKNl5qDps.

[27] M. Cuturi. 'Sinkhorn Distances: Lightspeed Computation of Optimal Transport'. In: *Advances in Neural Information Processing Systems*. Ed. by C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger. Vol. 26. 2013.

[28] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch. 'Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity'. In: *Journal of Medicinal Chemistry* 34.2 (1991), pp. 786–797. DOI: 10.1021/jm00106a046.

[29] T. K. Dey and Y. Wang. *Computational Topology for Data Analysis*. Cambridge, United Kingdom: Cambridge University Press, 2022. DOI: 10.1017/9781009099950.

[30] H. Edelsbrunner and J. Harer. 'Persistent homology—a survey'. In: *Surveys on discrete and computational geometry: Twenty years later*. Ed. by J. E. Goodman, J. Pach, and R. Pollack. Contemporary Mathematics 453. Providence, RI, USA: American Mathematical Society, 2008, pp. 257–282.

[31] H. Edelsbrunner, D. Letscher, and A. J. Zomorodian. 'Topological persistence and simplification'. In: *Discrete & Computational Geometry* 28.4 (2002), pp. 511–533. DOI: 10.1007/s00454-002-2885-2.

[32] F. Errica, M. Podda, D. Bacciu, and A. Micheli. 'A Fair Comparison of Graph Neural Networks for Graph Classification'. In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=HygDF6NFPB.

[33] R. J. Gardner, E. Hermansen, M. Pachitariu, Y. Burak, N. A. Baas, B. A. Dunn, M.-B. Moser, and E. I. Moser. 'Toroidal topology of population activity in grid cells'. In: *Nature* 602.7895 (2022), pp. 123–128. DOI: 10.1038/s41586-021-04268-7.

[34] M. Grohe. 'The Logic of Graph Neural Networks'. In: *Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 2021, pp. 1–17. DOI: 10.1109/LICS52264.2021.9470677.

[35] A. K. Hartmann and M. Mézard. 'Distribution of diameters for Erdős–Rényi random graphs'. In: *Physical Review E* 97 (3 2018), p. 032128. DOI: 10.1103/PhysRevE.97.032128.

[36] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl. 'Deep Learning with Topological Signatures'. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. 2017, pp. 1634–1644.

[37] C. D. Hofer, F. Graf, B. Rieck, M. Niethammer, and R. Kwitt. ‘Graph Filtration Learning’. In: *International Conference on Machine Learning (ICML)*. Ed. by H. Daumé III and A. Singh. Proceedings of Machine Learning Research 119. PMLR, 2020, pp. 4314–4323. arXiv: 1905.10996 [cs.LG].

[38] C. D. Hofer, R. Kwitt, and M. Niethammer. ‘Learning Representations of Persistence Barcodes’. In: *Journal of Machine Learning Research* 20.126 (2019), pp. 1–45.

[39] D. Horak, S. Maletić, and M. Rajković. ‘Persistent homology of complex networks’. In: *Journal of Statistical Mechanics: Theory and Experiment* 2009.3 (2009), P03034. DOI: 10.1088/1742-5468/2009/03/P03034.

[40] M. Horn, E. De Brouwer, M. Moor, Y. Moreau, B. Rieck, and K. Borgwardt. ‘Topological Graph Neural Networks’. In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=oxxUMeFwEHd.

[41] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. ‘Open Graph Benchmark: Datasets for Machine Learning on Graphs’. In: *arXiv preprint arXiv:2005.00687* (2020).

[42] J. Immonen, A. Souza, and V. Garg. ‘Going beyond persistent homology using persistent homology’. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 63150–63173.

[43] C. K. Joshi, C. Bodnar, S. V. Mathis, T. Cohen, and P. Lio. ‘On the Expressive Power of Geometric Graph Neural Networks’. In: *International Conference on Machine Learning (ICML)*. Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett. Proceedings of Machine Learning Research 202. PMLR, 2023, pp. 15330–15355.

[44] P. J. Kelly. ‘A congruence theorem for trees’. In: *Pacific Journal of Mathematics* 7.1 (1957), pp. 961–968. DOI: pjm/1103043674.

[45] R. Kwitt, S. Huber, M. Niethammer, W. Lin, and U. Bauer. ‘Statistical Topological Data Analysis - A Kernel Perspective’. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. 2015.

[46] T. Leinster. ‘The Euler characteristic of a category’. In: *Documenta Mathematica* 13 (2008), pp. 21–49.

[47] E. Levina and P. Bickel. ‘The Earth Mover’s distance is the Mallows distance: some insights from statistics’. In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. 2001, pp. 251–256. DOI: 10.1109/ICCV.2001.937632.

[48] S. Lim, F. Memoli, and O. B. Okutan. *Vietoris–Rips Persistent Homology, Injective Metric Spaces, and The Filling Radius*. 2020. arXiv: 2001.07588 [math.AT].

[49] L. Lovász. ‘Spectra of graphs with transitive groups’. In: *Periodica Mathematica Hungarica* 6.2 (1975), pp. 191–195.

[50] K. Maggs, C. Hacker, and B. Rieck. ‘Simplicial Representation Learning with Neural k-forms’. In: *International Conference on Learning Representations*. 2024. arXiv: 2312.08515 [cs.LG]. URL: https://openreview.net/forum?id=Djw0XhjHZb.

[51] L. Marsh and D. Beers. ‘Stability and Inference of the Euler Characteristic Transform’. Preprint. 2023. arXiv: 2303.13200 [math.ST].

[52] B. McKay. *Collection of strongly-regular graphs*. URL: https://users.cecs.anu.edu.au/~bdm/data/graphs.html.

[53] C. Morris, Y. Lipman, H. Maron, B. Rieck, N. M. Kriege, M. Grohe, M. Fey, and K. Borgwardt. ‘Weisfeiler and Leman go Machine Learning: The Story so far’. 2021. arXiv: 2112.09992 [cs.LG].

[54] 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 Conference on Artificial Intelligence*. 2019. DOI: 10.1609/aaai.v33i01.33014602.

[55] E. Munch. ‘An Invitation to the Euler Characteristic Transform’. Preprint. 2023. arXiv: 2310.10395 [cs.CG].

[56] J. R. Munkres. *Elements of Algebraic Topology*. Menlo Park, California, USA: Addison–Wesley Publishing Company, 1984.

[57] L. O'Bray, B. Rieck, and K. Borgwardt. 'Filtration Curves for Graph Representation'. In: *International Conference on Knowledge Discovery & Data Mining (KDD)*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1267–1275. DOI: 10.1145/3447548.3467442.

[58] Y. Ollivier. 'Ricci curvature of metric spaces'. In: *Comptes Rendus Mathematique* 345.11 (2007), pp. 643–646. DOI: 10.1016/j.crma.2007.10.041.

[59] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington. 'A roadmap for the computation of persistent homology'. In: *EPJ Data Science* 6.1 (2017), 17:1–17:38. DOI: 10.1140/epjds/s13688-017-0109-5.

[60] J. Palowitch, A. Tsitsulin, B. Mayer, and B. Perozzi. 'GraphWorld: Fake Graphs Bring Real Insights for GNNs'. In: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, pp. 3691–3701. DOI: 10.1145/3534678.3539203.

[61] P. A. Papp and R. Wattenhofer. 'A Theoretical Comparison of Graph Neural Network Extensions'. In: *International Conference on Machine Learning*. Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato. Proceedings of Machine Learning Research 162. PMLR, 2022, pp. 17323–17345.

[62] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt. 'A stable multi-scale kernel for topological machine learning'. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 4741–4748. DOI: 10.1109/CVPR.2015.7299106.

[63] B. Rieck. 'Persistent Homology in Multivariate Data Visualization'. PhD thesis. Heidelberg University, 2017. DOI: 10.11588/heidok.00022914.

[64] B. Rieck, C. Bock, and K. Borgwardt. 'A Persistent Weisfeiler–Lehman Procedure for Graph Classification'. In: *International Conference on Machine Learning (ICML)*. Ed. by K. Chaudhuri and R. Salakhutdinov. Proceedings of Machine Learning Research 97. PMLR, 2019, pp. 5448–5458.

[65] B. Rieck, U. Fugacci, J. Lukasczyk, and H. Leitte. 'Clique Community Persistence: A Topological Visual Analysis Approach for Complex Networks'. In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 822–831. DOI: 10.1109/TVCG.2017.2744321.

[66] B. Rieck, F. Sadlo, and H. Leitte. 'Topological Machine Learning with Persistence Indicator Functions'. In: *Topological Methods in Data Analysis and Visualization V*. Ed. by H. Carr, I. Fujishiro, F. Sadlo, and S. Takahashi. Cham, Switzerland: Springer, 2020, pp. 87–101. DOI: 10.1007/978-3-030-43036-8_6. arXiv: 1907.13496 [math.AT].

[67] E. Röell and B. Rieck. 'Differentiable Euler Characteristic Transforms for Shape Classification'. In: *International Conference on Learning Representations*. 2024. arXiv: 2310.07630 [cs.LG]. URL: https://openreview.net/forum?id=MO632iPq3I.

[68] A. Samal, R. P. Sreejith, J. Gu, S. Liu, E. Saucan, and J. Jost. 'Comparative analysis of two discretizations of Ricci curvature for complex networks'. In: *Scientific Reports* 8.1 (2018), p. 8650. DOI: 10.1038/s41598-018-27001-3.

[69] R. Sato. 'A Survey on The Expressive Power of Graph Neural Networks'. 2020. arXiv: 2003.04078 [cs.LG].

[70] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. 'Weisfeiler–Lehman Graph Kernels'. In: *Journal of Machine Learning Research* 12.77 (2011), pp. 2539–2561.

[71] P. Škraba and K. Turner. 'Wasserstein Stability for Persistence Diagrams'. 2020. arXiv: 2006.16824 [math.AT].

[72] J. Southern, J. Wayland, M. Bronstein, and B. Rieck. 'Curvature Filtrations for Graph Generative Model Evaluation'. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 63036–63061.

[73] R. Turkeš, G. Montúfar, and N. Otter. 'On the Effectiveness of Persistent Homology'. In: *Advances in Neural Information Processing Systems*. Ed. by A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho. 2022. URL: https://openreview.net/forum?id=DRjUkfExCix.

[74] K. Turner, Y. Mileyko, S. Mukherjee, and J. Harer. 'Fréchet Means for Distributions of Persistence Diagrams'. In: *Discrete & Computational Geometry* 52.1 (2014), pp. 44–70. DOI: 10.1007/s00454-014-9604-7.

[75] K. Turner, S. Mukherjee, and D. M. Boyer. ‘Persistent homology transform for modeling shapes and surfaces’. In: *Information and Inference: A Journal of the IMA* 3.4 (Dec. 2014), pp. 310–344. DOI: 10.1093/imaiai/iau011.

[76] C. Villani. *Optimal Transport. Old and New*. Grundlehren der mathematischen Wissenschaften 338. Springer, 2009.

[77] N. Wale and G. Karypis. ‘Comparison of Descriptor Spaces for Chemical Compound Retrieval and Classification’. In: *Sixth International Conference on Data Mining (ICDM’06)*. 2006, pp. 678–689. DOI: 10.1109/ICDM.2006.39.

[78] Y. Wang and M. Zhang. *Towards Better Evaluation of GNN Expressiveness with BREC Dataset*. 2023. arXiv: 2304.07702 [cs.LG].

[79] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. ‘How Powerful are Graph Neural Networks?’ In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=ryGs6iA5Km.

[80] Z. Yan, T. Ma, L. Gao, Z. Tang, Y. Wang, and C. Chen. ‘Neural Approximation of Graph Topological Features’. In: *Advances in Neural Information Processing Systems*. Ed. by A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho. 2022. URL: https://openreview.net/forum?id=qwjrO7Rewqy.

[81] P. Yanardag and S. Vishwanathan. ‘Deep graph kernels’. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2015, pp. 1365–1374.

[82] X. Ye, F. Sun, and S. Xiang. ‘TREPH: A Plug-In Topological Layer for Graph Neural Networks’. In: *Entropy* 25.2 (2023). DOI: 10.3390/e25020331.

[83] Q. Zhao and Y. Wang. ‘Learning metrics for persistence-based summaries and applications for graph classification’. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett. Vol. 32. 2019, pp. 9855–9866.

[84] Q. Zhao, Z. Ye, C. Chen, and Y. Wang. ‘Persistence Enhanced Graph Neural Network’. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by S. Chiappa and R. Calandra. Proceedings of Machine Learning Research 108. PMLR, 2020, pp. 2896–2906.

Appendix (Supplementary Materials)

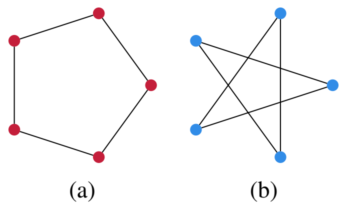


FIGURE S.1: Two isomorphic graphs. The isomorphism can be understood as a relabelling, a shift of node identities, or, most intuitively, as being able to draw both graphs using a single line.

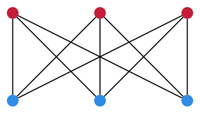


FIGURE S.2: A depiction of the *utilities problem*. Three houses (top row) are supposed to be connected to three different utilities (bottom row; commonly referred to as water, gas, and electricity) without the connection lines crossing one another. The figure shows an incorrect solution in which crossings occur. Using Euler's formula for planar graphs, we can prove that no such solution can exist in the plane. To see this, we set $V = 3$ and $E = 9$, as required by the utilities problem. Euler's formula now states that $F = 8$. However, we can see that F can be *at most* half the number of edges, i.e. $F \leq 4$. This is true because the edges of any face generated by a planar embedding have to alternate between the houses and utilities, respectively, meaning that every face has at least 4 edges, and each edge belongs to exactly 2 faces.

A Using Euler's Formula

Euler's formula is typically used to solve certain problems in graph theory, such as the *utilities problem* described in Fig. S.2. While arguably a contrived example in the context of machine learning, it nevertheless proves the point that a fundamental perspective of structural properties of a graph can well be used to reason about them. The expressive power of such simple computational invariants should not be underestimated, with recent work showing how to assign invariants like the *Euler characteristic* in a more abstract context [46].

B Counting Connected Components

Since the main text deals with higher-order topological features, and such features afford a substantially less intuitive grasp, we want to briefly comment on how to obtain β_0, the number of connected components. As with many problems in computer science, this procedure turns out to be simple if we pick our data structures correctly. Here, we need a *union–find* data structure, also known as a disjoint set forest. This data structure is built on the vertices of a graph and affords two operations, viz. `union` (or `merge`) and `find`. The `merge` operation assigns two vertices to the same connected component, while the `find` operation returns the current connected component of a vertex. Building such a data structure is reasonably easy in programming languages like `Python`, which offer *associative arrays*. Algorithm 1 shows one particular pseudo-code implementation of a simple union–find data structure. The pseudo-code assumes that all operations are changing objects 'in place.' Notice that the `find` operation is implemented implicitly via a lookup in the `merge` function. A proper object-oriented implementation of a union–find data structure should have these two operations in its public interface.

C A Primer in Computational Topology

With the understanding that our readers have different backgrounds, the following section provides a primer of the most relevant concepts in computational topology.

ALGORITHM 1 Using associative arrays to find connected components

```
function GET_CONNECTED_COMPONENTS($V, E$)
    UF $\leftarrow \{\}$
    for $v \in V$ do
        UF[$v$] $\leftarrow v$
    end for
    for $e = (v, w) \in E$ do
        MERGE(UF, $v$, $w$)
    end for
    return $\{v \mid$ UF[$v$] $= v\}$
end function

function MERGE(UF, $v$, $w$)
    if UF[$v$] $\neq$ UF[$w$] then
        UF[$v$] $\leftarrow w$
    end if
end function
```

C.1 Simplicial Homology

The Betti numbers of a graph are actually a special instance of a more generic concept, known as *simplicial homology*. We will see that under this framework, the Betti numbers are the ranks of the zeroth and first homology group, respectively. Simplicial homology is not required in order to understand most of the results of this paper, but an appreciation for some of the concepts will be helpful in understanding connections to other concepts. We try to provide a self-contained introduction to the most relevant concepts and refer to the textbook by Munkres [56] for a more in-depth exposition of these concepts. We start by introducing the central object of algebraic topology—the *simplicial complex*.[11]

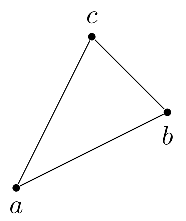
Definition 1 (Simplicial complex)**.** *A* simplicial complex K *is a system of sets that is closed under the subset operation. Thus, for any $\sigma \in \mathrm{K}$ and $\tau \subseteq \sigma$, we have $\tau \in \mathrm{K}$. An element $\sigma \in \mathrm{K}$ with $|\sigma| = k + 1$ is also referred to as a k-simplex. We also express this by writing $\dim \sigma = k$. Moreover, if k is maximal among all the simplices of K, we say that the K is a k-dimensional simplicial complex.*

Note that there is an unfortunate shift in dimensions: a k-simplex has indeed $k + 1$ elements. This convention makes sense when we relate it to the concept of *dimension*. A 0-simplex, i.e. a point or a vertex, should be assigned a dimension of 0. The reader should thus mentally equate the dimension of a simplex with its dimension. The text will aim to quell any confusion about such shifts. The quintessential example of a simplicial complex is a graph $G = (V, E)$. Setting $\mathrm{K} := V \cup E$, we obtain a 1-dimensional simplicial complex. We may calculate additional types of simplicial complexes from a graph, for instance by *expanding* each $(k + 1)$-clique into a k-simplex [39, 65].

The simplicial complex on its own is only a set system; to perform calculations with this type of data structure, we need to imbue it with additional operations. One of the most common operations involves defining homomorphisms between the subsets of a simplicial complex K.

Definition 2 (Chain group of a simplicial complex)**.** *Given a simplicial complex K, the vector space generated over $\mathbb{Z}_2$ coefficients whose elements are the k-simplices of K is called the kth* chain group*, denoted by $C_k(\mathrm{K})$. The elements of a chain group are also referred to as* simplicial chains*.*

[11]Technically, we will be working with *abstract simplicial complexes*. A definition of a simplicial complex in terms of convex subsets is also possible, but this necessitates understanding certain nuances that are irrelevant to this paper.



The triangle is a simple simplicial complex, consisting of one 2-simplex, three 1-simplices and three 0-simplices, respectively. The boundary of the 2-simplex is non-zero: we have $\partial_2\{a,b,c\} = \{b,c\} + \{a,c\} + \{a,b\}$. The set of edges, on the other hand, does not have a boundary, i.e. $\partial_1(\{b,c\} + \{a,c\} + \{a,b\}) = \{c\} + \{b\} + \{c\} + \{a\} + \{b\} + \{a\} = 0$, because the simplices cancel each other out.

FIGURE S.3: Calculating the boundaries of a 2-simplex and the boundary of a simplicial chain consisting of 1-simplices. Notice that the boundary of a boundary is always zero. This is a fundamental property of persistent homology. The figure is slightly adapted from Rieck [63].

Elements of the chain group are thus sums of simplices of a compatible dimension. For instance, we may write the sum of all edges of a graph to obtain a valid simplicial chain. Operating over $\mathbb{Z}_2$ coefficients means that $\sigma + \sigma = 0$, the empty chain, for all $\sigma \in \mathrm{K}$.[12] Simplicial chains permit us to define homomorphisms between chain groups, which will ultimately permit us to treat topological questions with tools of linear algebra.

Definition 3 (Boundary homomorphism)**.** *Given $\sigma = (v_0, \dots, v_k) \in \mathrm{K}$, we define the k*th boundary homomorphism *$\partial_k \colon C_k(\mathrm{K}) \to C_{k-1}(\mathrm{K})$ as*

$$\partial_k(\sigma) := \sum_{i=0}^{k} (v_0, \dots, v_{i-1}, v_{i+1}, \dots, v_k), \tag{4}$$

i.e. a sum of simplices with the ith entry—vertex—of the simplex missing, respectively.

It is sufficient to define ∂_k on individual simplices; since it is a homomorphism, it extends to arbitrary simplicial chains. Fig. S.3 shows an example of this calculation. The boundary operator already assigns some algebraic structure to K, but it turns out that we can use it to assign a set of groups to the simplicial complex.

Definition 4 (Homology group)**.** *We define the k*th homology group *of a simplicial complex K as*

$$\mathrm{H}_k(\mathrm{K}) := \ker \partial_k / \operatorname{im} \partial_{k+1}, \tag{5}$$

i.e. a quotient *group that we obtain from the two subgroups $\ker \partial_k$ and $\operatorname{im} \partial_{k+1}$ of C_k.*

The kth homology group of a simplicial complex contains its k-dimensional topological features in the form of *equivalence classes* of simplicial chains, also known as *homology classes*. This rather abstract definition is best understood by an additional simplification step that involves calculating the *rank* of a homology group.

Definition 5 (Betti number)**.** *The rank of the kth homology group $\mathrm{H}_k(\mathrm{K})$ is known as the k*th Betti number*, denoted by β_k.*

Despite the rank being a rather coarse summary, Betti numbers turn out to be of immense utility in comparing different simplicial complexes. We may even reproduce the equation for the cyclomatic number, $\beta_1 = m + \beta_0 - n$, by noting that the *Euler characteristic* $\chi(\mathrm{K}) := \sum_i (-1)^i \, |\{\sigma \mid \dim \sigma = i\}|$ can also be expressed as a sum of alternating Betti numbers, i.e. $\chi(\mathrm{K}) := \sum_i (-1)^i \beta_i$. For a proof of this surprising fact, see e.g. Munkres [56, p. 124]. Using this equivalence, we see that we can calculate β_1 by reshuffling some of the terms, thus also explaining why the equation exhibits alternating signs.

At this point, we have introduced a large amount of algebraic machinery. Changing our focus back to graphs, we may reap some advantageous properties by noting that homology groups are somewhat preserved under graph isomorphism.[13]

[12] Readers familiar with algebraic topology will recognise $\mathbb{Z}_2$ as a deliberate choice of *coefficient field* for the subsequent calculations. Other choices are possible, but the computational topology community predominantly uses $\mathbb{Z}_2$ coefficients in practice, with very few exceptions [33]. However, all the proofs and concepts introduced in this paper apply, *mutatis mutandis*, for other coefficient sets as well.

[13] The reader well-versed in algebraic topology may be aware of this property directly, but we find it useful to mention this fact briefly.

Lemma 1. *Let G, G' be two isomorphic graphs with $\varphi\colon G \to G'$ their corresponding isomorphism. Then the homology groups of G and G' are isomorphic, i.e. $\mathrm{H}_p(G) \simeq \mathrm{H}_p(G')$ for all p.*

Proof. This is a consequence of the *functoriality* of homology, which implies that any function $f\colon G \to G'$ induces a function $f_p\colon \mathrm{H}_p(G) \to \mathrm{H}_p(G')$ on the homology level. This function has the property that it composes in a natural manner with any other function $g\colon G \to G'$, namely $(f \circ g)_p = f_p \circ g_p$. In other words, functoriality implies that the evaluation order does not matter, or, intuitively, that the induced function commutes with function evaluation. In our case, setting $f = \varphi$ and $g = \varphi^{-1}$, this implies that $(\varphi \circ \varphi^{-1})_p = \varphi_p \circ \varphi_p^{-1}$, i.e. $\mathrm{id}_p = \varphi_p \circ \varphi_p^{-1}$, with id_p representing the identity function induced on the homology groups. The same calculations with f, g swapped show that the homology groups are isomorphic, with the isomorphism induced by φ, the isomorphism between G and G'. ■

As a direct corollary, the Betti numbers of G and G' do not change, and in fact, a similar property holds for isomorphic simplicial complexes. This may be seen as a hint about the popularity of simplicial homology in algebraic topology: the framework leads directly to characteristic descriptions that remain invariant under (graph) isomorphism.

C.2 Persistent Homology

Because of their conceptual simplicity—their calculation in low dimensions only involves knowledge about the connected components of a graph—Betti numbers are somewhat limited in their expressivity. Taking any graph $G = (V, E)$, even the addition of a single edge to G will change its Betti numbers, either by merging two connected components (thus decreasing β_0) or by creating an additional cycle (thus increasing β_1). This is a direct consequence of the definition of Euler's formula, which effectively states that the insertion of a new edge $e = (u, v)$ with $u, v \in V$ either causes β_1 to increase by 1 because m changes, or remain the same in case the number of connected components β_0 changes. However, a single edge may only merge two connected components into one, so β_0 may also at most decrease by 1. This indicates that Betti numbers are too coarse to be practically useful in large-scale graph analysis. It is possible to turn Betti numbers into a *multi-scale descriptor* of a graph. This requires certain modifications to the previously-introduced concepts. Similar to Appendix C.1, we will formulate everything in terms of simplicial complexes, again pointing out that this results in a more general description.

Definition 6 (Filtration)**.** *Given a simplicial complex* K, *we call a sequence of simplicial complexes* filtration *if it affords a nesting property of the form*

$$\emptyset = \mathrm{K}_0 \subseteq \mathrm{K}_1 \subseteq \cdots \subseteq \mathrm{K}_{m-1} \subseteq \mathrm{K}_m = \mathrm{K}. \tag{6}$$

Since each element of this sequence is a valid simplicial complex, we can also think of this construction as 'growing' K *by adding simplices one after the other.*

Filtrations arise naturally when building simplicial complexes from point cloud data, but even in the context of graphs, we can imagine filtrations as *filtering* a graph based on some type of data, or function, assigned to its vertices. For instance, we may build a filtration of a graph based on the degree of its vertices, defining K_i to be the subgraph consisting of all vertices satisfying the degree condition, plus all edges whose endpoints satisfy it, i.e.

$$\mathrm{K}_i := \{v \in V \mid \deg(v) \leq i\} \cup \{\{u, v\} \in E \mid \deg(u) \leq i \wedge \deg(v) \leq i\}. \tag{7}$$

Notice that we could also express the second condition more compactly by assigning to each 1-simplex (each edge) the *maximum* of the weight of its vertices. This construction is sometimes also referred to as a *lower-star filtration* since it extends a node-level function to higher-order simplices [29]. Not all filtrations have to be defined on the vertex level; as long as each edge in the filtration is preceded by its vertices, we can also build valid filtrations from functions that are primarily defined on edges.[14]

Setting aside further discussions about how to obtain filtrations for now, filtrations are compatible with the simplicial homology framework introduced above. The boundary operators $\partial(\cdot)$, together with the inclusion homomorphism between consecutive simplicial complexes, induce a homomorphism

[14]In Section 5, we will make use of a filtration defined on edge-based curvature values.

between corresponding homology groups of any filtration of m simplicial complexes. Given $i \leq j$, we write $\iota^{i,j}\colon \mathrm{H}_k(\mathrm{K}_i) \to \mathrm{H}_k(\mathrm{K}_j)$ to denote this homomorphism. This construction yields a sequence of homology groups

$$0 = \mathrm{H}_k(\mathrm{K}_0) \xrightarrow{\iota_k^{0,1}} \mathrm{H}_k(\mathrm{K}_1) \xrightarrow{\iota_k^{1,2}} \dots \xrightarrow{\iota_k^{m-2,m-1}} \mathrm{H}_k(\mathrm{K}_{m-1}) \xrightarrow{\iota_k^{m-1,m}} \mathrm{H}_k(\mathrm{K}_m) = \mathrm{H}_k(\mathrm{K}) \tag{8}$$

for every dimension k. We then define the *kth persistent homology group* as

$$\mathrm{H}_d^{i,j} := \ker \partial_k(\mathrm{K}_i) / (\operatorname{im} \partial_{k+1}(\mathrm{K}_j) \cap \ker \partial_k(\mathrm{K}_i)), \tag{9}$$

containing all topological features—homology classes—created in K_i that still exist in K_j. Following the definition of the ordinary Betti number, we then define the *kth persistent Betti number* to be the rank of this group, leading to $\beta_k^{i,j} := \operatorname{rank} \mathrm{H}_k^{i,j}$. It should be noted that this type of construction makes use of numerous deep mathematical concepts; for the sake of an expository article, we heavily summarise and compress everything to the most pertinent results.

The appeal of persistent homology can be seen when we start to make use of the features it captures. If we assume that our filtration is associated with a set of values $a_0 \leq a_1 \leq \dots \leq a_{m-1} \leq a_m$, such as the function values on the vertices, we can calculate *persistence diagrams*, i.e. simple topological feature descriptors.

Definition 7 (Persistence diagram)**.** *The* k-dimensional persistence diagram *of a filtration is the multiset of points in* $\mathbb{R}^2$ *that, for each pair* i, j *with* $i \leq j$*, stores the tuple* (a_i, a_j) *with multiplicity*

$$\mu_{i,j}^{(k)} := \left(\beta_k^{i,j-1} - \beta_k^{i,j}\right) - \left(\beta_k^{i-1,j-1} - \beta_k^{i-1,j}\right). \tag{10}$$

We will also assign a multiplicity to essential topological features *of the simplicial complex, setting*

$$\mu_{i,\infty}^{(k)} := \beta_k^{i,m} - \beta_k^{i-1,m}, \tag{11}$$

which denotes all features that are still present in the last simplicial complex of the filtration, i.e. in $\mathrm{K}_m = \mathrm{K}$*. The persistence diagram thus contain all the information carried by Betti numbers.*

Persistence diagrams summarise the topological activity of a filtration. Given a persistence diagram $\mathcal{D}$, for any tuple (a_i, a_j), the quantity $\|a_j - a_i\|$ is called the *persistence* of the respective topological feature. Persistence indicates whether a feature, created in some simplicial complex during the filtration, is prominent or not. This notion was originally introduced by Edelsbrunner et al. [31] to analyse the relevance of topological features of a distance function; the terminology is supposed to indicate the prominence of a topological feature. Features with a high persistence are commonly taken to be relevant, whereas features with a low persistence used to be considered as noise; this assumption is changing as, depending on the filtration, low persistence may also just imply 'low reliability.' [8]

Persistence diagrams can be endowed with different metrics and kernels [45, 62], and it is known that the space of persistence diagrams is an Alexandrov space with curvature bounded from below [74]. The most common metric to compare two persistence diagrams is the *bottleneck distance*, defined as

$$\mathrm{d_B}(\mathcal{D}, \mathcal{D}') := \inf_{\eta\colon \mathcal{D} \to \mathcal{D}'} \sup_{x \in \mathcal{D}} \|x - \eta(x)\|_\infty, \tag{12}$$

where η ranges over all bijections between the two persistence diagrams. Eq. (12) is solved using optimal transport; different cardinalities are handled by permitting points in one diagram to be transported to their corresponding projection on the diagonal. Another metric is the *Wasserstein distance*, in which the sup calculation is replaced by a weighted sum over all distances between points in a diagram.

D Theorems & Proofs

For the convenience of the reader, we restate all results again before providing their proof.

When working with filtrations in the subsequent proofs, it would be ideal to have filtrations that satisfy *injectivity* on the level of vertices, i.e. $f(v) \neq f(v')$ if $v \neq v'$. Such injective filtrations have the advantage of permitting gradient-based optimisation schemes [37]. The following lemma, first proved in Horn et al. [40], demonstrates that injectivity is not a strict requirement, though, as it is always possible to find an injective filtration function that is arbitrarily close (in the Hausdorff sense) to a non-injective filtration function.

Lemma 2. *For all $\epsilon > 0$ and a filtration function f defined on the vertices, i.e. $f\colon V \to \mathbb{R}^d$, there is an injective function $\tilde{f}\colon V \to \mathbb{R}^d$ such that $\|f - \tilde{f}\|_\infty < \epsilon$.*

Proof. Let $V = \{v_1, \dots, v_n\}$ be the vertices of a graph and $\operatorname{im} f = \{u_1, \dots, u_m\}$ be their images under f. Since f is not injective, we have $m < n$. We resolve non-injective vertex pairs iteratively. For $u \in \operatorname{im} f$, let $V' := \{v \in V \mid f(v) = u\}$. If V' only contains a single element, we do not have to do anything. Otherwise, for each $v' \in V'$, pick a new value from $\mathrm{B}_\epsilon(u) \setminus \operatorname{im} f$, where $\mathrm{B}_r(x) \subset \mathbb{R}^d$ refers to the open ball of radius r around a point x (for $d = 1$, this becomes an open interval in $\mathbb{R}$, but the same reasoning applies in higher dimensions). Since we only ever remove a finite number of points, such a new value always exists, and we can modify $\operatorname{im} f$ accordingly. The number of vertex pairs for which f is non-injective decreases by at least one in every iteration, hence after a finite number of iterations, we have modified f to obtain $\tilde{f}$, an *injective* approximation to f. By always picking new values from balls of radius ϵ, we ensure that $\|f - \tilde{f}\|_\infty < \epsilon$, as required. ■

Theorem 2. *Let $G \simeq G'$ be two isomorphic graphs, and φ be the respective isomorphism. For any equivariant filtration f, the corresponding persistence diagrams are equal.*

Proof. Since $G \simeq G'$, there exists a permutation σ such that $\sigma(G) = G'$. Moreover, since f is equivariant, the image of G under f will just be permutation of vectorial coordinates. This, in turn, implies that all weights of vertices and edges of G are permuted. This does not change the calculation of persistent homology, though, because the *ordering* given by the filtration does not change. The persistence diagrams of G and G' thus coincide. ■

Theorem 3. $^{(*)}$ *Given k-FWL colourings of two graphs G and G' that are different, there exists a filtration of G and G' such that the corresponding persistence diagrams in dimension $k-1$ or dimension k are different.*

Proof. The main idea involves harnessing the colours of k-tuples. We first identify all colours $c_1, c_2, \dots$ with natural numbers $1, 2, \dots$ We then expand G and G' to a simplicial complex that contains all k-tuples as its faces.[15] Moreover, we assign *all* simplices in dimensions less than or equal to $k-2$ a weight of 0. Each $(k-1)$-simplex is assigned its colour according to the respective k-FWL colouring. As a consequence of the *pairing lemma* of persistent homology, every $(k-1)$-simplex is either a creator simplex or a destroyer simplex [30].

We handle the case of *creator simplices* first. Each creator simplex gives rise to an essential persistence pair in the $(k-1)$-dimensional persistence diagram. Each such pair is of the form (i, ∞), where c_i is a colour according to k-FWL.

Each *destroyer simplex*, by contrast, destroys a topological feature created by a $(k-2)$-simplex, i.e. a $(k-1)$-tuples, resulting in a pair of the form $(\cdot, j)$, with again c_j being the corresponding k-FWL colour. This pair is part of a $(k-2)$-dimensional persistence diagram.

By assumption, the k-FWL colours of G and G' are different, so there must exist a colour c whose count is *different* in G and G', respectively. Since the *sum* of all colours arising in the two types of persistence pairs above is the number of colours of k-tuples, there is either a difference in colour counts in the $(k-1)$-dimensional or the $(k-2)$-dimensional persistence diagrams of G and G', showing that they are not equal. ■

Theorem 4. *Given 1-WL colourings of two graphs G and G' that are different, there exists a filtration of G and G' such that their persistence diagrams in dimension 0 are also different.*

Proof. Since the colourings are different, there is an iteration h of 1-WL such that the label sequences of G and G' are different. We thus have at least one colour—equivalently, one label—whose count is different. Let $\mathcal{L}^{(h)} := \{l_1, l_2, \dots\}$ be an enumeration of the finitely many hashed labels at iteration h. We can build a filtration function f by assigning a vertex v with label l_i to its index, i.e. $f(v) := i$, and setting $f(v, w) := \max\{f(v), f(w)\}$ for an edge (v, w). The resulting 0-dimensional persistence diagrams for G and G', denoted by $\mathcal{D}_0$ and $\mathcal{D}'_0$, respectively, now contain tuples of the

[15]When dealing with tuples or simplices, there is always the risk of an off-by-one error. In this theorem, despite dealing with k-FWL, only the $(k-1)$-simplices, which have k vertices, will be relevant.

form (i, j). Moreover, each vertex is guaranteed to give rise to *exactly* one such pair since each vertex creates a connected component in 0-dimensional persistent homology. Letting $\mu^{(i,j)}(\mathcal{D}_0)$ refer to the multiplicity of a tuple in $\mathcal{D}_0$, we know that, since the label count is different, there is *at least* one tuple (k, l) with $\mu^{(k,l)}(\mathcal{D}_0) \neq \mu^{(k,l)}(\mathcal{D}'_0)$. Hence, $\mathcal{D}_0 \neq \mathcal{D}'_0$. ■

We end this section with two results concerning *other* graph properties that are being captured by persistent homology.

Theorem 5. $^{(*)}$ *Given* any *vertex-based filtration f of a graph G with a single connected component such the values of f of the endpoints of edges are* strictly lower *than the values of their corresponding edges, we can upper-bound* $\operatorname{diam}(G)$*, the diameter of G, based on $\mathcal{D}_0$, the resulting persistence diagram in dimension* 0.

Proof. We can provide a procedure to obtain an upper bound d of $\operatorname{diam}(G)$ alongside the calculation of $\mathcal{D}_0$. To this end, we set $d = 0$. While calculating $\mathcal{D}_0$ with Algorithm 1, we check for each edge whether it is a destroyer, or a regular edge. If the edge is a destroyer, we increase d by one; otherwise, we do nothing. This procedure works because $\operatorname{diam}(G)$ is upper bounded by the diameter of the minimum spanning tree of G. Our estimate d counts the number of edges in such a tree. We thus have $\operatorname{diam}(G) \leq d$. The bound is tight for some graphs, such as line graphs. ■

Theorem 6. $^{(*)}$ *Given* any *vertex-based filtration f of a graph G, we can upper-bound the girth of G using the persistence diagrams in dimension* 0 *and* 1.

Proof. Similar to Theorem 5, we can use the calculation of $\mathcal{D}_0$, the persistence diagram in dimension 0, of the filtration function to obtain an upper bound of the girth of the graph. We calculate $\mathcal{D}_0$ with Algorithm 1, checking once again for each edge whether it is a creator or destroyer. If the edge is a destroyer, we stop and set our upper bound of the girth g to be the number of vertices in the two connected components that are being merged by the destroyer edge. Since the addition of the respective edge is guaranteed to create a cycle—the edge is a destroyer—the cycle has to make use of at least two vertices of the respective connected components. Our estimate g thus constitutes an upper bound of the girth of the graph. ■

Theorems 5 and 6 indicate that persistent homology captures more than 'just' topological information about a graph. We substantiate these theorems with empirical results concerning different filtrations and other graph properties in Section 5.3, thus showing how a topological perspective complements and enriches graph-learning tasks.

E Topology and the Weisfeiler–Leman Hierarchy

In the following, we want to briefly extend the argumentation of the expressivity of persistent homology with respect to the Weisfeiler–Leman hierarchy. Since 1-WL is oblivious to certain topological structures such as cycles [3], the existence of graphs with different Betti number counts proves that persistent homology is *strictly more expressive* than 1-WL. For example, consider a graph consisting of the union of two triangles, i.e. . This graph has $\beta_0 = \beta_1 = 2$ since it consists of two connected components and two cycles. If we change the connectivity slightly to obtain a hexagon, i.e. , we obtain a graph with $\beta_0 = \beta_1 = 1$. 1-WL is not able to distinguish between these graphs, but persistent homology can, since the Betti numbers of the graph are still encoded in the persistence diagram as essential features. Note that Theorem 4 does *not* apply to arbitrary filtrations since the theorem requires knowing the correct labels assigned by 1-WL. Finding filtration functions that are able to split graphs in a manner that is provably equivalent to 1-WL remains an open research question.

F Additional Expressivity Experiments

This section contains additional expressivity experiments that had to be excluded from the main paper for reasons of space.

TABLE S.1: Success rate (↑) of distinguishing pairs of *connected cubic graphs* when using five different filtrations at varying expansion levels of the graph (denoted by k). Due to the regularity of each graph, $k = 3$ is omitted since the clique complex of the graph is exactly the same as for $k = 2$. These graphs can be distinguished by 2-FWL but not by 1-WL.

Data	$k = 1$					$k = 2$				
	Filtration									
	D	O	F	L	V	D	O	F	L	V
`cub06`	0.00	**1.00**	**1.00**	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
`cub08`	0.00	0.90	0.70	0.00	0.00	0.90	0.90	0.90	**1.00**	0.90
`cub10`	0.00	0.98	0.66	0.00	0.00	0.81	0.99	0.88	**1.00**	0.85
`cub12`	0.00	0.99	0.64	0.00	0.00	0.80	1.00	0.87	**1.00**	0.87
`cub14`	0.00	0.99	0.62	0.00	0.00	0.79	1.00	0.86	**1.00**	0.89

F.1 Additional Results for Connected Cubic Graphs

We start our supplementary experiments by distinguishing connected cubic graphs, i.e. 3-regular graphs. These graphs cannot be distinguished by 1-WL, but they can be distinguished by 2-FWL [11, 53]. As such, they provide a good example of how different filtrations harness different types of graph information. Table S.1 shows the results. We first observe that the degree filtration is incapable of distinguishing graphs for $k = 1$. This is a direct consequence of the regularity—since the function is constant on the graph, persistent homology cannot capture any variability. This changes, however, when higher-order structures—triangles—are included for $k = 2$. We also observe that the Laplacian-based filtration for $k = 2$ exhibits strong empirical performance; in the absence of additional information, the spectral properties captured by the Laplacian help in distinguishing graphs. The Ollivier–Ricci curvature filtration is performing similarly well also for the same dimension, while it outperforms the Laplacian filtration for $k = 1$. In contrast to the Laplacian-based filtration, it does not require the calculation of eigenvalues, which may be prohibitive for larger graphs.[16] Finally, we see that the Forman–Ricci curvature filtration, a purely combinatorial quantity depending only on local counts, is consistently outperformed by the Ollivier–Ricci curvature for $k = 1$. However, for $k = 1$, the Forman–Ricci filtration outperforms the Laplacian filtration. Finally, the Vietoris–Rips filtration is incapable of distinguishing the graphs for $k = 1$, and performs similarly to the Forman–Ricci curvature for $k = 2$, suggesting that filtrations based on distances are not capable of capturing the graph structure in an optimal way.

F.2 Additional Results for the BREC Data Set

Table S.2 shows BREC data set results, itemised by the value of k, while Table S.3 provides a summary and performance comparison of the persistent homology results with other baselines.

F.3 Additional Figures for Graph-Property Prediction Tasks

Fig. S.4 shows the distributions of graph properties that we predict in the main paper in Section 5.3.

G Additional Graph Classification Experiments

To assess the capacity of persistent homology in graph learning tasks, we have designed a set of graph classification experiments that use the filtrations introduced in Section 5 to extract persistence diagrams from graph classification data set to perform inference on them. Two fundamental differences between our approach and the one used in [37] are that, we do not learn an optimal filtration and also we do not use deep learning models to perform the classification, as we are interested into the capacity of persistent homology to perform classification, and not in its combination with neural networks. For this endeavour, we train a Random Forest classifier on the persistent images [2]

[16]Ollivier–Ricci curvature requires solving optimal transport problems, for which highly efficient approximative solvers are available [27].

TABLE S.2: Success rate (↑) for distinguishing pairs of instances of the *BREC data set* when using five different filtrations at varying expansion levels of the graph (denoted by k). Due to combinatorial constraints, we did not calculate the Vietoris–Rips filtration for $k = 4$. Legend and number of graphs per category: `B` (Basic, 60), `R` (Regular, 100), `E` (Extension, 100), `C` (CFI, 100), `4, 20` (4-Vertex Condition), `D` (Distance-Regular, 20) graphs, respectively and `A` (average over full data set, 400).

$k = 1$

Data	*Filtration*				
	D	O	F	L	V
`Basic` (60)	0.03	0.93	0.87	**1.00**	0.00
`Regular` (100)	0.00	0.42	0.32	0.00	0.00
`Extension` (100)	0.07	0.76	0.44	0.94	0.00
`CFI` (100)	0.03	0.03	0.03	**0.06**	0.03
`4-VC` (20)	0.00	0.00	0.00	0.00	0.00
`DR` (20)	0.00	0.00	0.00	**0.05**	0.00
Average (400)	0.03	0.44	0.33	0.40	0.01

$k = 2$

Data	*Filtration*				
	D	O	F	L	V
`Basic` (60)	0.78	**1.00**	0.98	**1.00**	0.52
`Regular` (100)	0.39	0.54	0.50	0.48	0.39
`Extension` (100)	0.26	0.92	0.59	**1.00**	0.11
`CFI` (100)	0.03	0.03	0.03	**0.06**	0.03
`4-VC` (20)	0.00	0.00	0.00	0.00	0.00
`DR` (20)	0.00	0.00	0.00	**0.05**	0.00
Average (400)	0.29	0.52	0.43	0.54	0.21

$k = 3$

Data	*Filtration*				
	D	O	F	L	V
`Basic` (60)	0.83	**1.00**	0.98	**1.00**	0.58
`Regular` (100)	0.85	0.93	0.91	0.93	0.85
`Extension` (100)	0.29	0.92	0.59	**1.00**	0.16
`CFI` (100)	0.03	0.03	0.03	**0.06**	0.03
`4-VC` (20)	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
`DR` (20)	0.00	0.00	0.00	**0.05**	**0.05**
Average (400)	0.47	0.67	0.58	0.70	0.40

$k = 4$

Data	*Filtration*				
	D	O	F	L	V
`Basic` (60)	0.83	**1.00**	0.98	**1.00**	—
`Regular` (100)	0.89	0.97	0.95	0.97	—
`Extension` (100)	0.29	0.92	0.59	**1.00**	—
`CFI` (100)	0.03	0.03	0.03	0.06	—
`4-VC` (20)	**1.00**	**1.00**	**1.00**	**1.00**	—
`DR` (20)	0.00	0.00	0.00	0.05	—
Average (400)	0.48	0.68	0.59	0.71	—

TABLE S.3: Success rate (↑) for distinguishing pairs of instances of the *BREC data set* when using four different filtrations for $k = 4$. Vietoris–Rips filtration not computed for $k = 4$ due to computational constraints. The data sets Regular, 4-Vertex Condition, and Distance Regular, from Table 3 are merged into the All regular data set. Green indicates the best performing algorithm, while orange indicates the second best.

Data	SOTA		Filtration ($k = 4$)			
	3-WL	N_2	D	O	F	L
Basic (60)	1.000	1.000	0.83	1.000	0.983	1.000
All regular (140)	0.36	0.986	0.78	0.84	0.82	0.843
Extension (100)	1.000	1.000	0.29	0.920	0.59	1.000
CFI (100)	0.600	0.00	0.03	0.03	0.03	0.060
Average (400)	0.68	0.745	0.48	0.68	0.59	0.710

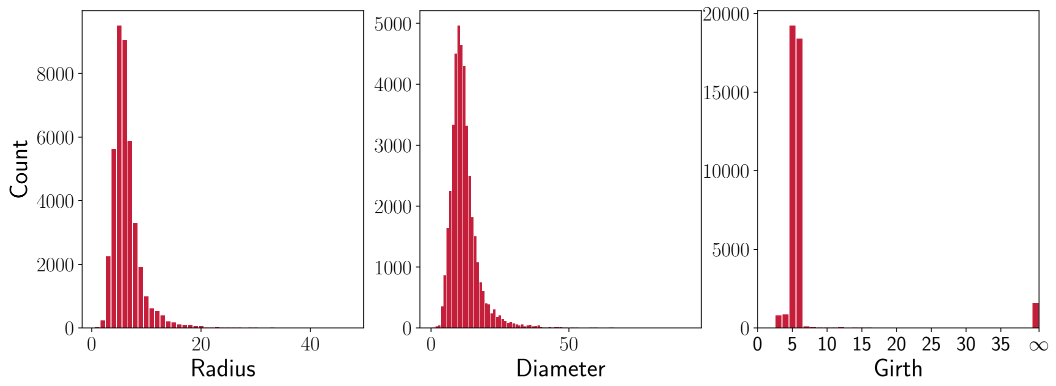


FIGURE S.4: Distribution of maximum radii, maximum diameters, and girths in the ogbg-molhiv molecular graph data set [41]. The values are concentrated on the lower end of the spectrum, with medians of 6, 11, and 5 and maximums of 47, 93, and 36, respectively. The maximum value for the girth is computed without taking into account infinite values.

computed from the persistence diagrams extracted from the input graphs using the previous filtrations up to dimension $k = 2$. We test our approach in the MUTAG [28], IMDB-Binary [81], PROTEINS [12], NC1 [77], and ogbg-molhiv [41] data set. Except for ogb-molhiv, we perform a Stratified 3-Fold experiment and provide the average and standard deviation of the multiple runs. For ogbg-molhiv, we perform only one experiment instance with the official train and test splits. The results are shown in Table S.4. We observe that, although we discard the data set features when using persistent homology, we are able to achieve suitably good results on some of the data sets, such as MUTAG and IMDB-Binary [32], even surpassing state-of-the-art results reported in Hofer et al. [37] and O'Bray et al. [57] for IMDB-Binary, MUTAG, and NCI1. For the ogbg-molhiv, however, we obtain consistent ROC-AUC values of 0.5, which is the same as random guessing. Our hypothesis is that molecular graphs rely on the annotated data set features to perform well, and the isolated topological information is not enough to perform the classification. The success of the experiments suggests that persistent homology is capable of capturing essential structural information about the graphs to classify and thus, suggests that persistent homology can be expressive in practice. We leave a more structured and comprehensive benchmark for persistent homology-like methods for graph-learning tasks to future work.

H Implementation and Hardware details

Our implementation is based on Python 3. We plan on releasing the code under a BSD-3-Clause license. For review purposes, the code has been attached to the supplementary materials. Please use `pip install -r requirements.txt` on the root folder of the code to set up the environment. The experiments were executed on a server with an AMD EPYC 7452 (128) @ 2.350GHz CPU,

TABLE S.4: Graph classification average accuracy (↑) and standard deviation in test for the experiments with data set `IMDB-Binary` (`I`), `PROTEINS` (`P`), `MUTAG` (`M`), and `NCI1` (`N`), and ROC-AUC (↑) for the `ogbg-molhiv` (`O`) test data set. The results are averaged over three runs using a stratified 3-means for all data set except for `ogbg-molhiv`, where only one run is performed. The best results are highlighted in bold. Experiments with the Vietoris–Rips filtration for the `PROTEINS` and `molhiv` are not reported for $k = 2$ due to out-of-resources errors during execution. S_1 and S_2 refer to accuracy of the most effective methods reported in [57] and [37], respectively. The abbreviation DS stands for data set.

				$k = 1$			
DS	SOTA		Filtration				
	S_1	S_2	D	O	F	L	V
`P`	**0.75 ± 0.05**	0.74 ± 0.03	0.70 ± 0.01	0.74 ± 0.01	0.72 ± 0.02	0.72 ± 0.02	0.70 ± 0.00
`I`	0.74 ± 0.04	**0.75 ± 0.05**	**0.75 ± 0.02**	0.71 ± 0.01	0.69 ± 0.03	0.65 ± 0.01	0.68 ± 0.03
`M`	0.87 ± 0.08	-	0.86 ± 0.01	0.87 ± 0.04	**0.90 ± 0.04**	0.79 ± 0.06	0.87 ± 0.01
`N`	-	0.71 ± 0.02	0.68 ± 0.00	0.73 ± 0.00	0.69 ± 0.01	0.67 ± 0.00	0.66 ± 0.01
`O`	-	-	**0.50**	0.50	**0.50**	0.50	0.50
				$k = 2$			
DS	SOTA		Filtration				
	S_1	S_2	D	O	F	L	V
`P`	0.75 ± 0.05	0.74 ± 0.03	0.70 ± 0.02	0.73 ± 0.02	0.70 ± 0.01	0.68 ± 0.01	-
`I`	0.74 ± 0.04	0.75 ± 0.05	0.73 ± 0.03	0.71 ± 0.02	0.72 ± 0.03	0.68 ± 0.03	0.66 ± 0.02
`M`	0.87 ± 0.08	-	0.86 ± 0.01	0.89 ± 0.04	0.89 ± 0.03	0.81 ± 0.04	0.87 ± 0.03
`N`	-	0.71 ± 0.02	0.68 ± 0.00	**0.74 ± 0.01**	0.70 ± 0.00	0.68 ± 0.00	0.69 ± 0.01
`O`	-	-	**0.50**	**0.50**	0.50	0.50	-

503GiB of RAM memory, no GPU acceleration, and Ubuntu 22.04.4 LTS with the 6.5.0-28-generic Linux kernel. Each experiment was executed on a single process. Not completed experiments in the main text were due to process termination by the operating system with the previous constraints.

- To perform the BREC experiments, use the script `BREC_experiments.py`.
- To perform the classification experiments, use the script `classification_tasks.py`.
- To perform the three non-BREC expressivity experiments (cubic, regular, and Cayley graphs), use the script `simple_isomorphism_experiments.py`.
- To perform the graph properties prediction experiments, use the code `property_prediction_experiments.py`.
- To perform the graph properties prediction experiments for the Watts–Strogatz and Erdős–Rényi random graphs, use the code `predict_diameter_random_graphs.py`.

The easiest way to perform the experiments in all the cases is using the flag `-a`, that executes all the experiments without the need of specifying specific parameters about the datasets and filtrations. Once the experiments are performed, the tables from the paper can be generated using the flag `-t` on the same scripts.

I Licenses

We do not redistribute any existing data sets, but briefly mention their licenses here. Expressivity experiments on known graphs make use of an existing database [25], which does not directly specify any licensing requirements but asks for a citation. The BREC data set and the `ogbg-molhiv` data set are distributed under a MIT license. All other graph-classification data sets, i.e. `IMDB-Binary`, `PROTEINS`, `MUTAG`, and `NCI1`, do not specify a license. They are distributed as part of the 'TUDatasets' repository, and can be accessed via `https://chrsmrrs.github.io/datasets/`. We will make our code available under a 3-Clause BSD License.