Diversity Curves for Graph Representation Learning

Katharina Limbeck[1,2], Nadja Häusermann[5], Martin Carrasco[5], Guy Wolf[†,3,4], and Bastian Rieck[†,1,5] ✉

[1] Helmholtz Munich, Munich, Germany
[2] Technical University of Munich, Munich, Germany
[3] Université de Montréal, Montreal, Canada
[4] Mila – Quebec AI Institute, Montreal, Canada
[5] Université de Fribourg, Fribourg, Switzerland
bastian.grossenbacher@unifr.ch

Abstract Graph-level representations are crucial tools for characterising structural differences between graphs. However, comparing graphs with different cardinalities, even when sampled from the same underlying distribution, remains challenging. Unsupervised tasks in particular require interpretable, scalable, and reliable size–aware graph representations. Our work addresses these issues by tracking the structural diversity of a graph across coarsening levels. The resulting graph embeddings, which we denote *diversity curves*, are interpretable by construction, efficient, and directly comparable across coarsening hierarchies. Specifically, we track the *spread* of graphs, a novel isometry invariant that is inherently well-suited for encoding the metric diversity and geometry of graphs. We utilise edge contraction coarsening and prove that this improves expressivity, thus leading to more powerful graph-level representations than structural descriptors alone. Demonstrating their utility over a range of baseline methods in practice, we use *diversity curves* to (i) cluster and visualise simulated graphs across varying sizes, (ii) distinguish the geometry of single-cell graphs, (iii) compare the structure of molecular graph datasets, and (iv) characterise geometric shapes.

Keywords: Unsupervised Representation Learning · Graph Learning.

1 Introduction

Understanding structural differences between graphs is a key step in exploring, describing, and distinguishing graph datasets or distributions. In particular when comparing graphs drawn from the same distribution, suitable graph representations should adapt to differences in scale and capture coarse structural similarities without relying on supervision. However, many existing methods for graph comparison and graph-level representation learning are inherently sensitive to differences in graph sizes [77, 92]. This can negatively impact interpretation

[†] These authors jointly supervised this work.

and downstream performance. We address this issue by investigating to what extent *graph coarsening* techniques can be used to perform structural comparisons between graphs. To this end, we propose *size-aware methods* that are aware of both the cardinality of graphs, as well as their intrinsic size, i.e. their structural diversity across coarsening scales. These size-aware representations aim to address geometric tasks, where graphs have been sampled from the same underlying distribution and share the same inherent structure but vary in size. Our work makes the following **contributions**:

Novel graph representations: We propose *diversity curves* for comparing graphs. Our approach is the first to use coarsening to derive hierarchical graph descriptors and leverages the expressive power of structural notions of diversity to track how graphs evolve across the coarsening hierarchy.
Expressivity guarantees: We prove that edge contraction coarsening improves the expressivity of graph structural comparisons via graph invariants.
Excellent performance: We show that diversity curves excel at geometric tasks, outperforming unsupervised baseline methods across a range of applications. Our methods perform well at clustering simulated graph distributions, distinguishing between trajectory- and cluster-like single-cell graphs, and characterising geometric graphs across varying sizes.

2 Related work

Graph kernel methods are frequently used as baselines for graph-level tasks. However, kernels require computing pairwise similarity, thus limiting their scalability. While more efficient kernels exist [5, 59], they often compromise expressivity. For example, path- or walk-based kernels suffer from size-sensitivity [27], making their embeddings less suitable in unsupervised settings. We observe similar issues with size-sensitivity in *unsupervised graph embeddings* such as graph2vec [58]. Spectral embeddings, such as NetLSD [86] or SpectralZoo [34], aim to characterise graphs via summarising the spectrum of eigenvalues of their normalised Laplacians. In practice, such methods still struggle with clustering graphs generated from the *same* model across varying densities and sizes [77]. Other approaches like FEATHER [71] pool random walks to obtain graph-level embeddings, which implicitly incorporates size.

Graph Neural Networks (GNNs), typically used in supervised settings, can be adapted via contrastive learning to learn graph-level representations for unsupervised tasks [93, 94]. However, their reliance on graph augmentations and comparisons between positive and negative pairs limits their training efficiency and their training objective is sensitive to varying graph sizes. Further, these models do not provide guarantees on which structures are encoded, thus hindering interpretability. Nevertheless, we will investigate the ability of GNNs to characterise geometric graphs across varying scales and upsampling operations [2]. Despite GNNs not specifically focusing on comparing graphs across different cardinalities, we may use their *pooling operators* to reduce graph size while preserving key properties such as connectivity, community structures, or spectral properties

[23]. We will particularly focus on *edge contraction pooling*. Since it provably preserves graph structure [51], it can be used without model training providing interpretable pooling operations that can be applied iteratively. Moreover, pooling is known to improve expressivity [42] and many graphs can even be reconstructed based on their set of possible edge contractions [67].

To obtain interpretable representations, prior work often employs geometry- or topology-based invariants. Methods based on *persistent homology* (PH) are expressive descriptors that enable assessing a graph at multiple scales [1, 76] through *graph filtrations*. This approach was formalised by TopER [82]. However, several filtrations remain expensive to compute while being size-sensitive. We thus see a crucial opportunity to propose size-robust graph embeddings that encode coarse similarities between graphs sampled from the same distribution.

In this context, the *magnitude* and the *spread* of a graph constitute promising graph invariants that summarise structural diversity. Based on a metric between nodes, both measure diversity as the effective size of a graph, i.e. the number of distinct nodes or communities. Originally meant to provide a mathematical foundation for measuring diversity in a metric space setting [44, 89], these measures have recently seen use in machine learning applications, for instance in evaluating the diversity of latent representations across varying scales [50]. Limbeck et al. [51] further used these invariants to compute geometry-aware edge importance scores for guiding edge contraction pooling towards preserving structural diversity. Both magnitude and spread are highly-correlated and preserve relevant graph properties. In this paper we opt for *spread*, since it is computationally more efficient than *magnitude*, and investigate how to leverage the notion of structural diversity on graphs to learn meaningful graph embeddings.

3 Methods

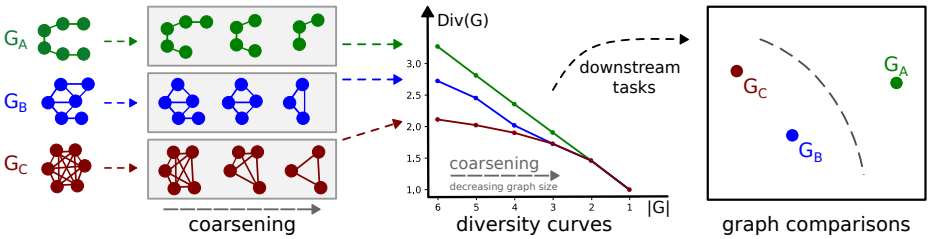

Figure 1: Diversity curves for three graphs. Our framework *coarsens* graphs via iterative edge contractions and tracks structural invariants across the process. We choose structural diversity based on the shortest path metric. *Diversity curves* then represent this diversity as a function of the number of nodes in the coarsened graph. The resulting vector encodes the coarsened geometry of graphs for visualisation, clustering, classification or explanatory tasks.

We propose a framework to compare the structural diversity of graphs across a coarsening process. The resulting diversity curves yield a vectorised representation for each graph, encoding their hierarchical geometry, which can be used for downstream analysis. Figure 1 illustrates the construction of diversity curves, Section 3.1 describes the process and Section 3.2 their properties.

3.1 Diversity curves as graph representations

Graph coarsening as a multiscale process. We first select a set of node numbers, $\mathcal{I} \subset \mathbb{N}$, across which we compare the coarsened geometry of graphs. We coarsen or upsample graphs to these same sizes. As a coarsening operation, we choose edge contraction, because edge contraction pooling was shown to improve expressivity [42], can be applied iteratively giving us direct control over the size of the graphs, is an interpretable operation, and does well at preserving the structure of graphs [51]. We follow the implementation by Limbeck et al. [51]. That is, we compute an importance score for every edge that determines the order in which edges are contracted and is either derived from the edge's importance for the graph's structure as done by Limbeck et al. [51], or selected randomly. Then, the edges with the lowest scores get contracted iteratively, so that adjacent nodes are not merged more than once. We reapply this procedure to coarsen graphs to the desired size. If needed, the node features of any merged nodes are aggregated, for instance by averaging. Formally, a graph $G = (X, E)$ consists of a set of vertices X and a set of edges E. We use the notation $|G| = |X| = n$ for the cardinality of the graph and $m = |E|$ for the number of edges. Furthermore, we define a sequence $\{e_i\}_{1 \le i \le k}$ of edges in G that are to be contracted for some $k \in \{1, \dots, m\}$, and a sequence of coarsened graphs $\{G_i\}_{n-k \le i \le n}$ based on contracting these edges, so that $G_n = G$ and $G_{i-1} = G_i / e_{n-i+1}$. If a dataset contains small graphs, we further employ upsampling strategies, for example by uniformly upsampling a node and reconnecting it to all neighbours of the initial node. This strategy allows us to extend the sequence of coarsened graphs to $\{G_i\}_{n-k \le i \le N}$ for some $N > n$.

Structural diversity for comparing graphs. Structural diversity is inherently well-suited to compare graphs at the same cardinality. By measuring diversity as the *effective number* of distinct nodes in a graph, the *spread* of a graph is an expressive invariant that encodes the graph's geometry based on a choice of distance between nodes. Throughout this work, we consider a graph $G = (X, E)$ to be associated with a metric d, so that (X, d) defines a finite metric space. Given such a graph, the *spread* of its metric space [51, 89], is defined as

$$\mathrm{Div}(G) \coloneqq \sum_{x \in X} \frac{1}{\sum_{y \in X} \exp(-d(x, y))}. \tag{1}$$

Through coarsening, we aim to exploit the inherent link between diversity and the notion of effective size. Spread is more efficient to compute than alternative diversity measures like the magnitude of a metric space [43, 89], while allowing for a flexible choice of geometry determined by the metric on the graph. Further,

spread can be directly compared across graphs as it is invariant to isometries and permutations between nodes [51]. By default, we use shortest-path distances for diversity computations as we find that they perform well across our experiments and are faster to estimate than alternative graph metrics. Section C.8 gives an ablation on the choice of metric on distinguishing graph distributions, validating our choice. Coarsening allows these invariants to be robust to varying node sizes. In fact, coarsening counteracts one of the notable weaknesses of many geometric tools, diversity measures, or path-based summaries, which are known to struggle with directly comparing data of varying sizes [50]. Our framework thus still allows us to use shortest-path distances to make comparisons across coarsened graphs.

Diversity curves as size-aware graph representations. Our final graph representations, which we call *diversity curves*, track the structural diversity of each graph as a function of the number of nodes in the coarsened graph. Given a graph G associated with a metric d, a sequence of integers $\mathcal{I}$ representing cardinalities, and a choice of down- or upsampling strategy to construct a sequence of coarsened graphs, $\{G_i\}_{i\in\mathcal{I}} = \mathcal{G}_\mathcal{I}$ such that $|G_i| = i$, we define its *diversity curve*, $\mathrm{DivCurve}(G) = \mathrm{DivCurve}(\mathcal{G}_\mathcal{I}) \in \mathbb{R}^{|\mathcal{I}|}$, as

$$\mathrm{DivCurve}(G)_i := \mathrm{Div}(G_i). \tag{2}$$

This yields a vector representation for each graph, which describes the graph's coarsened geometry across resolutions. Intuitively, graphs with few localised differences will behave similarly when contracted and have more similar diversity curves than graphs with globally different structures. Diversity curves capture exactly these hierarchical trends. In fact, the value of the diversity curve at each scale directly summarises how structurally diverse a given graph is, leading to a clear interpretation as the effective number of distinct nodes in the coarsened graph. This gives a multiscale and size-aware summary of graph structure and diversity that is robust to differences in graph size.

Default parameter choices. As a default, we set the coarsening scales $\mathcal{I}$ at which we compute diversity curves to all integers between one and the maximum number of nodes in the dataset, but other choices are possible, and set $\mathrm{DivCurve}(G)_1 = 1$ as a graph with one node has one distinct point [69, 89]. Further, we cannot use edge contraction pooling to reduce the size of a graph to less than the number of connected components. In these cases, we use linear interpolation of the diversity curves to estimate the value of diversity at lower sizes. We use random edge contractions throughout our experiments as we find the choice of edge scoring has no notable impact on performance (cf. Section C.8). Further, because up- and downsampling procedures introduce some randomness, we do in practice average diversity curves across multiple repeats. This is motivated by the fact that while we do not immediately know which coarsening sequence best describes a given graph, we aim to capture key structural characteristics through repeated coarsening. In fact, we will prove how this pooling strategy improves expressivity [42]. We detail our algorithm in Section A.1.

3.2 Properties of diversity curves

In the following, we describe the theoretical properties that ensure diversity curves are well-behaved and expressive graph representations. Complete proofs and further results are provided in Section B. For some of the theoretical treatment, we consider the diversity curve of a graph G not just with respect to *one* sequence of coarsened graphs, but *all* possible coarsening sequences. Concretely, we consider all possible sequences of edges $s_e = \{e_i\}_{i \leq k}$ in G of length at most $k \leq m$ and the sequence of coarsened graphs $\mathcal{G}_i(s_e)$ obtained from contracting the edges in G according to the sequence s_e. Then we define the *multiset* $\{\{\text{DivCurve}(\mathcal{G}_i(s_e)) \mid s_e \text{ is a valid edge contraction sequence of length at most } k\}\}$. This idealised setting allows us to talk about the invariance and expressivity of diversity curves from a theoretical perspective. However, the evaluation of the multiset quickly becomes infeasible in practice. Hence, we empirically compute the mean diversity curves over a small number of edge contraction sequences. Section C.8 verifies that this empirical approach is nevertheless reasonably stable with respect to the chosen edge contraction sequence and the number of evaluated sequences.

Diversity curves as graph invariants. Our measure of structural diversity is invariant under permutations of vertices and under isometries [51, 89], it follows that the following invariance under isomorphic graphs holds.

Theorem 1. *For two isomorphic graphs, the multisets of the diversity curves over all possible edge contractions are equal.*

Diversity curves collapse to the same diversity. We investigate the limiting behaviour of diversity curves at low coarsening scales by considering the properties of structural diversity. The result below shows that diversity curves can distinguish the number of connected components of a graph and behave reasonably under increased coarsening.

Theorem 2. *Any two graphs G and H with the same number c of connected components will collapse to the same graph of cardinality c and their diversity curves agree for $i \in \{1, \dots c\}$, concretely* $\text{DivCurve}(G)_i = \text{DivCurve}(H)_i = i$. *Furthermore,* $\text{DivCurve}(H)_{c+1} = \text{DivCurve}(G)_{c+1} \neq c + 1$.

Diversity curves increase expressivity. Expressivity, that is the ability to distinguish between graphs, is a key property of useful graph representations [88]. To motivate the use of diversity as a graph invariant together with the coarsening process, we show that expressivity is improved when combining them. We follow Lachi et al. [42] and show that diversity curves improve upon the expressivity of diversity as measured by the spread of a graph through edge contraction pooling.

Definition 1. *For two functions φ and ψ on the set of graphs, we say that φ is* more expressive *than ψ if for any graphs G_A, G_B such that $\psi(G_A) \neq \psi(G_B)$, then also $\varphi(G_A) \neq \varphi(G_B)$, and there exist graphs G_C, G_D such that $\varphi(G_C) \neq \varphi(G_D)$ and $\psi(G_C) = \psi(G_D)$.*

Theorem 1. *The multiset of the diversity curves over all possible edge contractions is more expressive than spread.*

The proof of Theorem 1 consists of providing a pair of graphs with identical spread but distinguishable by their diversity curves. In Section 4.6 we further demonstrate empirically that this holds not just for one example, but improves expressivity across a wide selection of graphs.

Computational complexity. The time complexity of computing the diversity curve is given by $C_{\text{Coarse}} = \mathcal{O}(|E|(\log(|E|) + |X|) + n_{\max}^2)$, constructing one sequence of random edge contractions and the upsampling, and $C_{\text{Div}} = O(|\mathcal{I}| \cdot n_{\max}^3)$, evaluating the structural diversity of all coarsened graphs, where $n_{max} = \max_{i \in \mathcal{I}} i$. Section B contains a more detailed description of the theoretical time complexity. We further evaluate the empirical costs in Section C.7.

4 Experiments

We show the following experimental results: **Section 4.1** verifies that the changes in diversity curves induced by edge perturbations correlates with the strength of the perturbation. This confirms that our methods capture differences between modified graphs. Next, **Section 4.2** demonstrates that diversity curves perform well at distinguishing graph distributions and parameter choices while being robust to variations in graph sizes. Based on this, we investigate applications to bioinformatics tasks. **Section 4.3** shows that diversity curves distinguish between the coarse geometry of trajectory- and cluster-like single-cell graphs surpassing more complex summaries. **Section 4.4** uses diversity curves to characterise similarities between enzyme graphs through statistical comparisons. **Section 4.5** further confirms that diversity curves reach high performance at distinguishing geometric shapes. Finally, **Section 4.6** reports empirical expressivity results.

4.1 Diversity curves track graph perturbations

As a first correctness check, we investigate how diversity curves behave under perturbations of the input graphs [62]. We fix a degree of perturbation $p \in [0, 1]$ and apply edge perturbations by consecutively adding, removing, swapping, or rewiring edges, resulting in edits to all graphs in the `SBM`, `PLANAR`, `EGO`, `LOBSTER` and `COMMUNITIES` datasets [40, 62]. For each dataset and degree of perturbation, we compute the mean norm across diversity curves, which summarises the structural diversity across graphs. We expect that the more graphs are perturbed, the higher the change in diversity curves should be. Figure 2 shows this trend holds true in practice. Across almost all datasets and perturbation types, the higher the perturbation degree, the higher the change in diversity curves. When adding edges, nodes become more similar and structural diversity decreases, hence we observe almost perfect negative correlation. When removing edges, distances between nodes and the value of diversity curves increase, leading to perfect positive correlation. Rewiring and swapping edges similarly tends to destroy graph structure

and increase structural diversity with high correlations for almost all datasets other than PLANAR, which by construction contains more random connectivities. Additional results in Section C.1 confirm that diversity curves detect small perturbations resulting in structural changes to graphs.

Figure 2: Spearman correlation between the degree of perturbation and the change in diversity curve per dataset.

4.2 Diversity curves distinguish graph distributions

To investigate size-robustness, we test whether our method can distinguish graphs from different distributions. In particular, we consider Erdős-Rényi (ER), stochastic block model (SBM), random partition (RP), and random geometric (RG) graphs. For each random graph model and graph sizes between 10 and 29 nodes, we generate three graphs using a fixed parameter setting as detailed in Section C.2. We compute diversity curves using shortest-path distance and use the L^2-norm of the vectorial representations to visualise the distributions with PCA. Figure 3 shows that our embedding separates clearly between graphs from different distributions. Further, we use the diversity curves for 10-fold stratified grouped CV using a kNN-classifier to distinguish between graph models. We compare to relevant unsupervised graph representation methods, namely graph kernels [75], spectral graph representations via SpectralZoo [34] and NetLSD [86], topology-inspired graph embeddings via TopER [82] and vectorial representation of persistence images computed through persistent homology [1], as well as the random walk based approach, FEATHER [71]. The results in Table 1 demonstrate that diversity curves perform similarly to the best baselines in terms of classification accuracies.

Table 1: Performance (mean ± std) of unsupervised methods at distinguishing graphs from four different distributions.

Method	Accuracy (↑)	Silhouette Score (↑)
Diversity Curves (ours)	$\mathbf{0.90 \pm 0.06}$	$\mathbf{0.42 \pm 0.03}$
Kernel WL Opt. Assig.	$\mathbf{0.90 \pm 0.05}$	0.02 ± 0.00
Kernel Shortest Path	0.77 ± 0.07	0.28 ± 0.04
TopER	0.70 ± 0.10	-0.06 ± 0.02
NetLSD	$\mathbf{0.90 \pm 0.03}$	0.06 ± 0.04
FEATHER	0.75 ± 0.09	0.11 ± 0.02
Spectral Zoo	0.72 ± 0.09	0.05 ± 0.03
PH Vietoris–Rips	0.78 ± 0.07	-0.01 ± 0.03

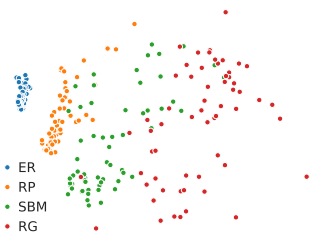

Figure 3: PCA of diversity curves for ER, RP, SBM, and RG models.

To further investigate the quality of each graph representation, Table 1 compares *silhouette scores*, obtained from 50% of the data over 100 resamples,

as a measure of clustering quality, for which diversity curves outperform all other methods. This confirms that diversity curves achieve better clustering than any of the tested baseline methods demonstrating their superior performance at distinguishing graph distributions in an unsupervised manner while being robust to differences in graph sizes. Second, we test the ability of our method to characterise and distinguish between graphs generated from the same underlying random graph model using three different parameter choices. For each graph distribution, parameter setting as described in Appendix C.2, and graph sizes from 10 to 29 nodes we generate three graphs. As before, we use the Euclidean distances of the diversity curves as input to a kNN-classifier with 10-fold stratified grouped CV. We compare silhouette scores in Figure 4 against our baselines, where we achieve comparable silhouette scores to the best kernel baselines for the ER and RP distributions and obtain the highest scores for the RG and SBM graphs. This confirms that diversity curves outperform alternative unsupervised methods at distinguishing between parameters used in a random graph model, while being robust to size differences. Additional clustering quality metrics and extended results in Appendix C.2 confirm our findings.

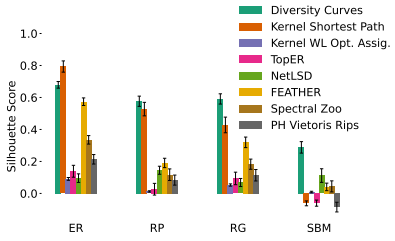

Figure 4: Silhouette scores (↑) for distinguishing different parameter choices.

4.3 Diversity curves distinguish trajectory or cluster structures in single-cell graphs

Table 2: Performance at distinguishing between trajectory- and cluster-like single-cell graphs.

Dataset	Method	Test Accuracy (↑)
Gold (87 Graphs)	Diversity Curves (ours)	**0.860 ± 0.077**
	NetLSD	0.823 ± 0.047
	Scores ([49])	0.804 ± 0.053
	Best Kernel	0.785 ± 0.079
	TopER	0.672 ± 0.086
All (169 Graphs)	Diversity Curves (ours)	**0.759 ± 0.077**
	NetLSD	0.731 ± 0.047
	Scores ([49])	0.716 ± 0.053
	Best Kernel	0.685 ± 0.092
	TopER	0.624 ± 0.086
	Landscape ([49])	0.570 ± 0.025

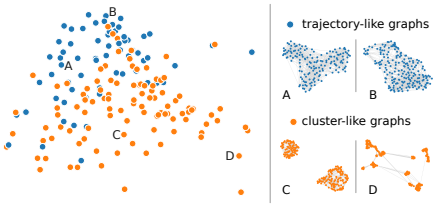

Figure 5: PCA visualisation of the diversity curves for all 169 single-cell graphs (left). Examples of two trajectory-like graphs and two cluster-like single-cell graphs (right).

Comparing between graphs of different sizes is required when analysing single-cell transcriptomics data, where the number of cells often varies widely due to experimental factors. A key question is whether single-cell graphs show cluster- or trajectory-like structures. This inherently geometric question informs the

biological interpretation of data as well as the choice of computational tools and visualisation methods designed for either type. We consider a collection of 169 curated single-cell graphs that have been annotated as cluster- or trajectory-like using biological ground truths [49, 72]. For each graph we compute the diversity curve using shortest-path distances and the diversity curve using Euclidean distances between node features, concatenate both curves, and use these curves as our representations. As described in Section C.3, we note that cluster-like graphs have on average higher structural diversity than trajectory-like graphs. The PCA plot in Figure 5 further visualises that our graph representation separates well between both types.We compare to the methods of Lim and Qiu [49], who compute a set of five geometric scores: the entropy of pairwise diffusion pseudo-time geodesic distances, an entropy-based persistent homology score, a summary of vector norms, a summary of Ripley's K function, and a summary of reachability values. To our knowledge, these are the only methods that have so far been investigated for this task. Due to their complexity, these scores are defined in more detail in Section C.3. We use the scores by Lim and Qiu [49] as inputs for 5-fold stratified CV using a kNN-classifier. Table 2 shows classification accuracies compared to other methods. We achieve notably better accuracies using diversity curves, which also provide more useful and more interpretable representations than existing methods designed for this task. This shows the potential of diversity curves for distinguishing the geometry of single-cell graphs giving highly actionable insights into deciding further analysis steps.

4.4 Diversity curves characterise molecular graph datasets

Characterising the coarse geometry of graphs, diversity curves can be used to compare and find structural similarities between graph datasets. Specifically, we explore common trends across classes of molecular graphs from the `ENZYMES` or `PROTEINS` datasets [6], where each graph represents a protein, with some proteins also being enzymes. We first compare the mean diversity curves per class, observing that non-enzyme graphs are notably distinct and have lower structural diversity on average than enzyme graphs. We further implement a non-parametric two-sample permutation test for the difference in mean diversity curves using the L^2-norm between the curves as a test statistic. Figure 6 reports the p-values, which we computed across 1000 random permutations of the data, adding a pseudo-count for error control [65]. The results confirm that the set of non-enzymes isdistinguishable from all enzyme classes. We find no evidence to indicate that enzymes from the `PROTEINS` dataset differ from

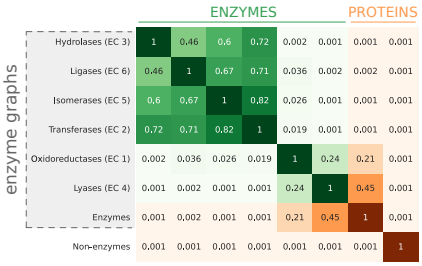

Figure 6: Two sample permutation testing results (p-values) for testing the equality in mean diversity curves per class in the `PROTEINS` and `ENZYMES` datasets using the L^2-norm between curves.

the EC4 or EC1 classes in the ENZYME dataset through their averaged diversity curves. This verifies that the geometry of these proteins is more coarsely similar to enzymes than to non-enzymes, even when comparing across datasets. Hence, we show that enzymes that share similar biological functions have the same structure across different coarsening scales and datasets. which is a crucial question for evaluating graph learning datasets [12], and a step towards quantifying structural similarities for transfer learning tasks [32].

4.5 Diversity curves characterise geometric shapes

We further perform an experiment to support our claim that diversity curves are capable of describing purely geometric graphs. To this end, we consider the MANTRA dataset [2], which contains, among others, triangulations of 2-manifolds, namely a Klein bottle (4657 meshes), a torus (2247 meshes), a sphere (308 meshes), and the real projective plane (1365 meshes). Distinguishing these manifolds has been shown to be challenging for graph-based models [2]. Following our pipeline, we compute diversity curves using shortest-path distances on the 1-skeleton graphs of the triangulation and use these representations as inputs for SVM classification. We denote methods that take graphs as inputs by G in Table 3. Because Klein bottles and tori have the same 1-skeletons, an optimal graph classifier could reach up to 73.8% accuracy. As reported in Table 3, our method almost reaches this theoretical upper bound in practice. Next, we extend our approach and compute diversity curves from heat kernel distances between faces computed from the Hodge Laplacian [39], which encodes higher order structures in mesh datasets. Models that operate on triangulations are denoted by $\mathcal{T}$ in Table 3. Following the same classification procedure as before, the combined diversity curves almost completely solve the task and reach 99% accuracy at distinguishing the homeomorphism types. This is superior to the best baseline, the cellular transformer (CT), reported by Ballester et al. [2], which indicates that there is potential to use and extend our representations for mesh data and shape tasks. We further visualise a PCA plot of the combined diversity curves in Figure 7, which shows that our methods separate the classes in MANTRA.

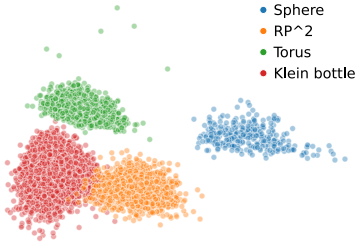

Figure 7: PCA visualisation of diversity curves for MANTRA.

Table 3: Model performance (mean ± std) at distinguishing homeomorphism types of triangulations of 2-manifolds from MANTRA across stratified 5-fold CV.

Method	Version	Accuracy (↑)	AUROC (↑)
Diversity Curve (ours)	$\mathcal{T}$	**0.99 ± 0.00**	**1.00 ± 0.00**
CT (best result from [2])	$\mathcal{T}$	0.91 ± 0.01	0.94 ± 0.00
Diversity Curve (ours)	G	**0.74 ± 0.00**	**0.85 ± 0.00**
Optimal Classifier	G	**0.74 ± 0.00**	0.80 ± 0.00
Diversity Curve (ours)	$\mathcal{T}_{\text{up}}$	**0.83 ± 0.02**	**0.96 ± 0.01**
CT (best result from [2])	$\mathcal{T}_{\text{up}}$	0.74 ± 0.00	0.83 ± 0.01

Lastly, we investigate if our representations encode intrinsic geometric information that is robust to varying graph sizes. To do so, we train our classifier on the diversity curves of the original dataset, but test on a modified dataset created by barycentric subdivision of the original triangulations [2]. This version of the task is denoted by $\mathcal{T}_{\text{up}}$. As reported in Table 3, our methods *retain* high classification performance surpassing alternative models at this arguably difficult experimental setup. We conclude that diversity curves are expressive representations that effectively characterise geometric graphs across varying resolutions and sizes.

4.6 Diversity curves are expressive graph representations

To support our theoretical expressivity analysis, Section C.6 further investigates whether our methods can distinguish pairs of WL-indistinguishable graphs from the `BREC` datasets [42, 88]. The results confirm that diversity curves improve the expressivity of structural diversity measures through graph coarsening. On its own, structural diversity based on shortest-path distances can distinguish 19% of graphs in `BREC` through their absolute difference in spread within an error tolerance of 10^{-5}. Further, by comparing diversity curves evaluated for all possible ways of contracting one edge, we increase the proportion of graph pairs that can be distinguished to 50.8%. This demonstrates the practical utility of using edge pooling for increasing expressivity. Further, these insights highlight that diversity curves successfully distinguish 1-WL indistinguishable graphs.

5 Discussion

By introducing diversity curves as size-aware graph representations, our work contributes a unified framework for graph structural comparisons through coarsening. Diversity curves provably improve expressivity through edge contraction pooling and reach top performance at geometric tasks. In Section C.8 we show that diversity curves clearly outperform simple structural baselines obtained from the graph itself. Nevertheless, our approach has certain *limitations*: We focus on structural comparisons rather than attributed graphs and do not aim to reach state-of-the-art performance in supervised settings. Instead, diversity curves offer interpretable and expressive tools for *unsupervised* structural and distributional comparisons between graphs. Further, our methods implicitly rely on the quality of the employed graph coarsening procedure. For example, some of our theoretical results assume we compute diversity curves exactly for all possible choices of edge contractions, which may not always be feasible. Section C.8 provides an ablation study on the impact the selected pooling strategy and graph metric on the experiment from Figure 4 , justifying the utility of our default choices. Nevertheless, building upon our work, there is potential to generalise the choice of coarsening process, or the graph invariant to track over the coarsening process. Furthermore, there is also the possibility to change the underlying metric; this flexibility was already used to characterise geometric shapes in Section 4.5. Finally, diversity curves could be integrated into supervised graph learning architectures as expressive structural encodings.

Bibliography

[1] Ballester, R., Rieck, B.: On the expressivity of persistent homology in graph learning. In: Proceedings of the Third Learning on Graphs Conference, pp. 42–1, PMLR (2025)

[2] Ballester, R., Röell, E., Schmid, D.B., Alain, M., Escalera, S., Casacuberta, C., Rieck, B.: MANTRA: The manifold triangulations assemblage. In: International Conference on Learning Representations (2025)

[3] Berman, P., Kasiviswanathan, S.P.: Faster approximation of distances in graphs. In: Dehne, F., Sack, J.R., Zeh, N. (eds.) Algorithms and Data Structures, pp. 541–552 (2007)

[4] Biase, F.H., Cao, X., Zhong, S.: Cell fate inclination within 2-cell and 4-cell mouse embryos revealed by single-cell RNA sequencing. Genome research **24**(11), 1787–1796 (2014)

[5] Borgwardt, K., Ghisu, E., Llinares-López, F., O'Bray, L., Rieck, B.: Graph kernels: State-of-the-art and future challenges. Foundations and Trends in Machine Learning **13**(5-6), 531–712 (2020)

[6] Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**(Suppl 1), i47–i56 (2005)

[7] Buettner, F., Natarajan, K.N., Casale, F.P., Proserpio, V., Scialdone, A., Theis, F.J., Teichmann, S.A., Marioni, J.C., Stegle, O.: Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. Nature biotechnology **33**(2), 155–160 (2015)

[8] Burns, J.C., Kelly, M.C., Hoa, M., Morell, R.J., Kelley, M.W.: Single-cell RNA-seq resolves cellular complexity in sensory organs from the neonatal inner ear. Nature communications **6**(1), 8557 (2015)

[9] Camp, J.G., Badsha, F., Florio, M., Kanton, S., Gerber, T., Wilsch-Bräuninger, M., Lewitus, E., Sykes, A., Hevers, W., Lancaster, M., et al.: Human cerebral organoids recapitulate gene expression programs of fetal neocortex development. Proceedings of the National Academy of Sciences **112**(51), 15672–15677 (2015)

[10] Chu, L.F., Leng, N., Zhang, J., Hou, Z., Mamott, D., Vereide, D.T., Choi, J., Kendziorski, C., Stewart, R., Thomson, J.A.: Single-cell RNA-seq reveals novel regulators of human embryonic stem cell differentiation to definitive endoderm. Genome biology **17**(1), 173 (2016)

[11] Close, J.L., Yao, Z., Levi, B.P., Miller, J.A., Bakken, T.E., Menon, V., Ting, J.T., Wall, A., Krostag, A.R., Thomsen, E.R., et al.: Single-cell profiling of an in vitro model of human interneuron development reveals temporal dynamics of cell type production and maturation. Neuron **93**(5), 1035–1048 (2017)

[12] Coupette, C., Wayland, J., Simons, E., Rieck, B.: No metric to rule them all: Toward principled evaluations of graph-learning datasets. In: Proceedings

of the 42nd International Conference on Machine Learning, vol. 267, pp. 11405–11434, PMLR (2025)

[13] Darmanis, S., Sloan, S.A., Zhang, Y., Enge, M., Caneda, C., Shuer, L.M., Hayden Gephart, M.G., Barres, B.A., Quake, S.R.: A survey of human brain transcriptome diversity at the single cell level. Proceedings of the National Academy of Sciences **112**(23), 7285–7290 (2015)

[14] Deng, Q., Ramsköld, D., Reinius, B., Sandberg, R.: Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells. Science **343**(6167), 193–196 (2014)

[15] Dobson, P.D., Doig, A.J.: Distinguishing enzyme structures from non-enzymes without alignments. Journal of Molecular Biology **330**(4), 771–783 (2003)

[16] Dory, M., Forster, S., Kirkpatrick, Y., Nazari, Y., Williams, V.V., Vos, T.d.: Fast 2-approximate all-pairs shortest paths. In: Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 4728–4757, SIAM (2024)

[17] Dunne, K.: Efficiently approximating spread dimension with high confidence. arXiv preprint arXiv:2408.14590 (2024)

[18] Engel, I., Seumois, G., Chavez, L., Samaniego-Castruita, D., White, B., Chawla, A., Mock, D., Vijayanand, P., Kronenberg, M.: Innate-like functions of natural killer T cell subsets result from highly divergent gene programs. Nature immunology **17**(6), 728–739 (2016)

[19] Frazer, S., Prados, J., Niquille, M., Cadilhac, C., Markopoulos, F., Gomez, L., Tomasello, U., Telley, L., Holtmaat, A., Jabaudon, D., et al.: Transcriptomic and anatomic parcellation of 5-HT3AR expressing cortical interneuron subtypes revealed by single-cell RNA sequencing. Nature communications **8**(1), 14219 (2017)

[20] Gokce, O., Stanley, G.M., Treutlein, B., Neff, N.F., Camp, J.G., Malenka, R.C., Rothwell, P.E., Fuccillo, M.V., Südhof, T.C., Quake, S.R.: Cellular taxonomy of the mouse striatum as revealed by single-cell RNA-seq. Cell reports **16**(4), 1126–1137 (2016)

[21] Goolam, M., Scialdone, A., Graham, S.J., Macaulay, I.C., Jedrusik, A., Hupalowska, A., Voet, T., Marioni, J.C., Zernicka-Goetz, M.: Heterogeneity in Oct4 and Sox2 targets biases cell fate in 4-cell mouse embryos. Cell **165**(1), 61–74 (2016)

[22] Granit, R.Z., Masury, H., Condiotti, R., Fixler, Y., Gabai, Y., Glikman, T., Dalin, S., Winter, E., Nevo, Y., Carmon, E., et al.: Regulation of cellular heterogeneity and rates of symmetric and asymmetric divisions in triple-negative breast cancer. Cell reports **24**(12), 3237–3250 (2018)

[23] Grattarola, D., Zambon, D., Bianchi, F.M., Alippi, C.: Understanding pooling in graph neural networks. IEEE Transactions on Neural Networks and Learning Systems **35**(2), 2708–2718 (2022)

[24] Guo, F., Yan, L., Guo, H., Li, L., Hu, B., Zhao, Y., Yong, J., Hu, Y., Wang, X., Wei, Y., et al.: The transcriptome and DNA methylome landscapes of human primordial germ cells. Cell **161**(6), 1437–1452 (2015)

[25] Habib, N., Li, Y., Heidenreich, M., Swiech, L., Avraham-Davidi, I., Trombetta, J.J., Hession, C., Zhang, F., Regev, A.: Div-seq: Single-nucleus RNA-seq reveals dynamics of rare adult newborn neurons. Science **353**(6302), 925–928 (2016)

[26] Hagberg, A.A., Schult, D.A., Swart, P.J.: Exploring network structure, dynamics, and function using networkx. In: Proceedings of the 7th Python in Science Conference, pp. 11 – 15 (2008)

[27] Han, R., Yao, G., Li, M., Zhang, W., Li, Y., Xin, W., Lei, H., Li, M., Zhang, Z., Du, C., et al.: Multi-relation graph-kernel strengthen network for graph-level clustering. In: 2025 International Joint Conference on Neural Networks (IJCNN), pp. 1–8, IEEE (2025)

[28] Han, X., Wang, R., Zhou, Y., Fei, L., Sun, H., Lai, S., Saadatpour, A., Zhou, Z., Chen, H., Ye, F., et al.: Mapping the mouse cell atlas by microwell-seq. Cell **172**(5), 1091–1107 (2018)

[29] Hayashi, T., Ozaki, H., Sasagawa, Y., Umeda, M., Danno, H., Nikaido, I.: Single-cell full-length total RNA sequencing uncovers dynamics of recursive splicing and enhancer RNAs. Nature communications **9**(1), 619 (2018)

[30] Hie, B., Cho, H., DeMeo, B., Bryson, B., Berger, B.: Geometric sketching compactly summarizes the single-cell transcriptomic landscape. Cell systems **8**(6), 483–493 (2019)

[31] Hochgerner, H., Zeisel, A., Lönnerberg, P., Linnarsson, S.: Conserved properties of dentate gyrus neurogenesis across postnatal development revealed by single-cell RNA sequencing. Nature neuroscience **21**(2), 290–299 (2018)

[32] Huang, T., Hu, Z., Ying, R.: Learning to group auxiliary datasets for molecule. Advances in Neural Information Processing Systems **36**, 45339–45359 (2023)

[33] Jang, Y., Lee, S., Ahn, S.: A simple and scalable representation for graph generation. In: International Conference on Learning Representations (2024)

[34] Jin, S., Zafarani, R.: The spectral zoo of networks: Embedding and visualizing networks with spectral moments. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1426–1434 (2020)

[35] Joost, S., Zeisel, A., Jacob, T., Sun, X., La Manno, G., Lönnerberg, P., Linnarsson, S., Kasper, M.: Single-cell transcriptomics reveals that differentiation and spatial signatures shape epidermal and hair follicle heterogeneity. Cell systems **3**(3), 221–237 (2016)

[36] Kim, T., Chen, I.R., Lin, Y., Wang, A.Y.Y., Yang, J.Y.H., Yang, P.: Impact of similarity metrics on single-cell RNA-seq data clustering. Briefings in bioinformatics **20**(6), 2316–2326 (2019)

[37] Klein, A.M., Mazutis, L., Akartuna, I., Tallapragada, N., Veres, A., Li, V., Peshkin, L., Weitz, D.A., Kirschner, M.W.: Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. Cell **161**(5), 1187–1201 (2015)

[38] Kowalczyk, M.S., Tirosh, I., Heckl, D., Rao, T.N., Dixit, A., Haas, B.J., Schneider, R.K., Wagers, A.J., Ebert, B.L., Regev, A.: Single-cell RNA-seq reveals changes in cell cycle and differentiation programs upon aging of hematopoietic stem cells. Genome research **25**(12), 1860–1872 (2015)

[39] Krahn, M., Garg, V.: Heat kernel goes topological. arXiv preprint arXiv:2507.12380 (2025)
[40] Krimmel, M., Hartout, P., Borgwardt, K., Chen, D.: Polygraph discrepancy: a classifier-based metric for graph generation. In: International Conference on Learning Representations (2026)
[41] Krzak, M., Raykov, Y., Boukouvalas, A., Cutillo, L., Angelini, C.: Benchmark and parameter sensitivity analysis of single-cell RNA sequencing clustering methods. Frontiers in genetics **10**, 1253 (2019)
[42] Lachi, V., Moallemy-Oureh, A., Roth, A., Welke, P.: Expressive pooling for graph neural networks. Transactions on Machine Learning Research (2025)
[43] Leinster, T.: The magnitude of metric spaces. Documenta Mathematica **18**, 857–905 (2013)
[44] Leinster, T.: Entropy and Diversity: The Axiomatic Approach. Cambridge University Press (2021)
[45] Leng, N., Chu, L.F., Barry, C., Li, Y., Choi, J., Li, X., Jiang, P., Stewart, R.M., Thomson, J.A., Kendziorski, C.: Oscope identifies oscillatory genes in unsynchronized single-cell RNA-seq experiments. Nature methods **12**(10), 947–950 (2015)
[46] Li, H., Courtois, E.T., Sengupta, D., Tan, Y., Chen, K.H., Goh, J.J.L., Kong, S.L., Chua, C., Hon, L.K., Tan, W.S., et al.: Reference component analysis of single-cell transcriptomes elucidates cellular heterogeneity in human colorectal tumors. Nature genetics **49**(5), 708–718 (2017)
[47] Li, H., Horns, F., Wu, B., Xie, Q., Li, J., Li, T., Luginbuhl, D.J., Quake, S.R., Luo, L.: Classifying drosophila olfactory projection neuron subtypes by single-cell RNA sequencing. Cell **171**(5), 1206–1220 (2017)
[48] Li, L., Dong, J., Yan, L., Yong, J., Liu, X., Hu, Y., Fan, X., Wu, X., Guo, H., Wang, X., et al.: Single-cell RNA-seq analysis maps development of human germline cells and gonadal niche interactions. Cell stem cell **20**(6), 858–873 (2017)
[49] Lim, H.S., Qiu, P.: Quantifying the clusterness and trajectoriness of single-cell RNA-seq data. PLoS Computational Biology **20**(2), e1011866 (2024)
[50] Limbeck, K., Andreeva, R., Sarkar, R., Rieck, B.: Metric space magnitude for evaluating the diversity of latent representations. In: Advances in Neural Information Processing Systems, vol. 37 (2024)
[51] Limbeck, K., Mezrag, L., Wolf, G., Rieck, B.: Geometry-aware edge pooling for graph neural networks. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)
[52] Loh, K.M., Chen, A., Koh, P.W., Deng, T.Z., Sinha, R., Tsai, J.M., Barkal, A.A., Shen, K.Y., Jain, R., Morganti, R.M., et al.: Mapping the pairwise choices leading from pluripotency to human bone, heart, and other mesoderm cell types. Cell **166**(2), 451–467 (2016)
[53] Marques, S., Zeisel, A., Codeluppi, S., Van Bruggen, D., Mendanha Falcão, A., Xiao, L., Li, H., Häring, M., Hochgerner, H., Romanov, R.A., et al.: Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system. Science **352**(6291), 1326–1329 (2016)

[54] Martinkus, K., Loukas, A., Perraudin, N., Wattenhofer, R.: Spectre: Spectral conditioning helps to overcome the expressivity limits of one-shot graph generators. In: Proceedings of the 39th International Conference on Machine Learning, pp. 15159–15179, PMLR (2022)

[55] Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tudataset: A collection of benchmark datasets for learning with graphs. In: ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020) (2020)

[56] Nakamura, T., Okamoto, I., Sasaki, K., Yabuta, Y., Iwatani, C., Tsuchiya, H., Seita, Y., Nakamura, S., Yamamoto, T., Saitou, M.: A developmental coordinate of pluripotency among mice, monkeys and humans. Nature **537**(7618), 57–62 (2016)

[57] Nakamura, T., Yabuta, Y., Okamoto, I., Sasaki, K., Iwatani, C., Tsuchiya, H., Saitou, M.: Single-cell transcriptome of early embryos and cultured embryonic stem cells of cynomolgus monkeys. Scientific data **4**(1), 170067 (2017)

[58] Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., Jaiswal, S.: graph2vec: Learning distributed representations of graphs. arXiv preprint arXiv:1707.05005 (2017)

[59] Nikolentzos, G., Siglidis, G., Vazirgiannis, M.: Graph kernels: A survey. Journal of Artificial Intelligence Research **72**, 943–1027 (2021)

[60] Ollivier-Gooch, C.: Coarsening unstructured meshes by edge contraction. International Journal for Numerical Methods in Engineering **57**(3), 391–414 (2003)

[61] Olsson, A., Venkatasubramanian, M., Chaudhri, V.K., Aronow, B.J., Salomonis, N., Singh, H., Grimes, H.L.: Single-cell analysis of mixed-lineage states leading to a binary cell fate choice. Nature **537**(7622), 698–702 (2016)

[62] O'Bray, L., Horn, M., Rieck, B.A., Borgwardt, K.: Evaluation metrics for graph generative models: Problems, pitfalls, and practical solutions. In: International Conference on Learning Representations (2022)

[63] Park, J., Shrestha, R., Qiu, C., Kondo, A., Huang, S., Werth, M., Li, M., Barasch, J., Suszták, K.: Single-cell transcriptomics of the mouse kidney reveals potential cellular targets of kidney disease. Science **360**(6390), 758–763 (2018)

[64] Petropoulos, S., Edsgärd, D., Reinius, B., Deng, Q., Panula, S.P., Codeluppi, S., Reyes, A.P., Linnarsson, S., Sandberg, R., Lanner, F.: Single-cell RNA-seq reveals lineage and x chromosome dynamics in human preimplantation embryos. Cell **165**(4), 1012–1026 (2016)

[65] Phipson, B., Smyth, G.: Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn. Statistical Applications in Genetics and Molecular Biology **9**, Article39–Article39 (2010)

[66] Plass, M., Solana, J., Wolf, F.A., Ayoub, S., Misios, A., Glažar, P., Obermayer, B., Theis, F.J., Kocks, C., Rajewsky, N.: Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. Science **360**(6391), eaaq1723 (2018)

[67] Poirier, A.: Reconstructing a graph from its edge-contractions. Ph.D. thesis, Université d'Ottawa/University of Ottawa (2018)
[68] Qiu, W.L., Zhang, Y.W., Feng, Y., Li, L.C., Yang, L., Xu, C.R.: Deciphering pancreatic islet β cell and α cell maturation pathways and characteristic features at the single-cell level. Cell metabolism **25**(5), 1194–1205 (2017)
[69] Roff, E., Yoshinaga, M.: The small-scale limit of magnitude and the one-point property. Bulletin of the London Mathematical Society **57**(6), 1841–1855 (2025)
[70] Rozemberczki, B., Kiss, O., Sarkar, R.: Karate club: An API oriented open-source python framework for unsupervised learning on graphs. In: Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), pp. 3125—-3132 (2020)
[71] Rozemberczki, B., Sarkar, R.: Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In: Proceedings of the 29th ACM international conference on information & knowledge management, pp. 1325–1334 (2020)
[72] Saelens, W., Cannoodt, R., Todorov, H., Saeys, Y.: A comparison of single-cell trajectory inference methods. Nature biotechnology **37**(5), 547–554 (2019)
[73] Saliba, A.E., Li, L., Westermann, A.J., Appenzeller, S., Stapels, D.A., Schulte, L.N., Helaine, S., Vogel, J.: Single-cell RNA-seq ties macrophage polarization to growth rate of intracellular salmonella. Nature microbiology **2**(2), 16206 (2016)
[74] Shalek, A.K., Satija, R., Shuga, J., Trombetta, J.J., Gennert, D., Lu, D., Chen, P., Gertner, R.S., Gaublomme, J.T., Yosef, N., et al.: Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. Nature **510**(7505), 363–369 (2014)
[75] Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., Vazirgiannis, M.: GraKeL: A graph kernel library in python. Journal of Machine Learning Research **21**(54), 1–5 (2020)
[76] Southern, J., Wayland, J., Bronstein, M., Rieck, B.: Curvature filtrations for graph generative model evaluation. Advances in Neural Information Processing Systems **36**, 63036–63061 (2023)
[77] Tantardini, M., Ieva, F., Tajoli, L., Piccardi, C.: Comparing methods for comparing networks. Scientific reports **9**(1), 17557 (2019)
[78] Tasic, B., Menon, V., Nguyen, T.N., Kim, T.K., Jarsky, T., Yao, Z., Levi, B., Gray, L.T., Sorensen, S.A., Dolbeare, T., et al.: Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. Nature neuroscience **19**(2), 335–346 (2016)
[79] The Tabula Sapiens Consortium, Jones, R.C., Karkanias, J., Krasnow, M.A., Pisco, A.O., Quake, S.R., Salzman, J., Yosef, N., Bulthaup, B., Brown, P., et al.: The tabula sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. Science **376**(6594), eabl4896 (2022)
[80] Tian, L., Dong, X., Freytag, S., Lê Cao, K.A., Su, S., JalalAbadi, A., Amann-Zalcenstein, D., Weber, T.S., Seidi, A., Jabbari, J.S., et al.: Benchmarking single cell rna-sequencing analysis pipelines using mixture control experiments. Nature methods **16**(6), 479–487 (2019)

[81] Tirosh, I., Izar, B., Prakadan, S.M., Wadsworth, M.H., Treacy, D., Trombetta, J.J., Rotem, A., Rodman, C., Lian, C., Murphy, G., et al.: Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. Science **352**(6282), 189–196 (2016)

[82] Tola, A., Taiwo, F.M., Akcora, C.G., Coskunuzer, B.: TopER: Topological embeddings in graph representation learning. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2025)

[83] Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N.J., Livak, K.J., Mikkelsen, T.S., Rinn, J.L.: The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nature biotechnology **32**(4), 381–386 (2014)

[84] Treutlein, B., Brownfield, D.G., Wu, A.R., Neff, N.F., Mantalas, G.L., Espinoza, F.H., Desai, T.J., Krasnow, M.A., Quake, S.R.: Reconstructing lineage hierarchies of the distal lung epithelium using single-cell RNA-seq. Nature **509**(7500), 371–375 (2014)

[85] Treutlein, B., Lee, Q.Y., Camp, J.G., Mall, M., Koh, W., Shariati, S.A.M., Sim, S., Neff, N.F., Skotheim, J.M., Wernig, M., et al.: Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-seq. Nature **534**(7607), 391–395 (2016)

[86] Tsitsulin, A., Mottin, D., Karras, P., Bronstein, A., Müller, E.: NetLSD: hearing the shape of a graph. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp. 2347–2356 (2018)

[87] Villani, A.C., Satija, R., Reynolds, G., Sarkizova, S., Shekhar, K., Fletcher, J., Griesbeck, M., Butler, A., Zheng, S., Lazo, S., et al.: Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors. Science **356**(6335), eaah4573 (2017)

[88] Wang, Y., Zhang, M.: An empirical study of realized GNN expressiveness. In: Proceedings of the 41st International Conference on Machine Learning, vol. 235, pp. 52134–52155, PMLR (2024)

[89] Willerton, S.: Spread: a measure of the size of metric spaces. International Journal of Computational Geometry & Applications **25**(03), 207–225 (2015)

[90] Xin, Y., Kim, J., Okamoto, H., Ni, M., Wei, Y., Adler, C., Murphy, A.J., Yancopoulos, G.D., Lin, C., Gromada, J.: RNA sequencing of single human islet cells reveals type 2 diabetes genes. Cell metabolism **24**(4), 608–615 (2016)

[91] Yang, L., Wang, W.H., Qiu, W.L., Guo, Z., Bi, E., Xu, C.R.: A single-cell transcriptomic analysis reveals precise pathways and regulatory mechanisms underlying hepatoblast differentiation. Hepatology **66**(5), 1387–1401 (2017)

[92] Yehudai, G., Fetaya, E., Meirom, E., Chechik, G., Maron, H.: From local structures to size generalization in graph neural networks. In: Proceedings of the 38th International Conference on Machine Learning, vol. 139, pp. 11975–11986, PMLR (2021)

[93] Yin, Y., Wang, Q., Huang, S., Xiong, H., Zhang, X.: AutoGCL: Automated graph contrastive learning via learnable view generators. Proceedings of the AAAI Conference on Artificial Intelligence **36**(8), 8892–8900 (Jun 2022)

[94] You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc. (2020)

[95] Zeisel, A., Muñoz-Manchado, A.B., Codeluppi, S., Lönnerberg, P., La Manno, G., Juréus, A., Marques, S., Munguba, H., He, L., Betsholtz, C., et al.: Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq. Science **347**(6226), 1138–1142 (2015)

Acknowledgments. This work has received funding from the Swiss State Secretariat for Education, Research, and Innovation (SERI). G. W. was partially funded by Humboldt Research Fellowship, CIFAR AI Chair, NSERC Discovery grant 03267, FRQNT grant 343567, and NSF grant DMS-2327211. The content provided here is solely the responsibility of the authors and does not necessarily represent the views of the funding agencies.

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

Appendices and Supplementary Materials

A Extended methods

To support the introduction of diversity curves in Section 3.1, we give a more detailed descriptions of our proposed methods, as well as the steps involved in computing diversity curves, and the outline of the final algorithm. Further, we briefly discuss the compute resources used for our experiments, and give a brief introduction to unsupervised baseline methods, with which we compare diversity curves throughout our main experiments.

A.1 Diversity curves

In the following, we define how to compute diversity curves in detail, describe the algorithm used to compute them, and discuss specific parameter choices, such as the choice of pooling operation and the choice of graph metric.

Coarsening operations Throughout our work, we explore edge contraction pooling to downsample the size of graphs. Further, we use upsampling operations through upsampling nodes in a graph or triangles in a mesh. These operations are detailed below. Note that to clarify our choice of coarsening operators, we specifically select transformations that allow us to control the number of nodes in the coarsened graph and induce an iterative process that induces stepwise changes on the graph structure. This allows us to consider specific sequences of sequentially coarsened graphs. Fulfilling these requirements, we find that edge contraction coarsening is a specifically promising operation. In contrast, many other coarsening operations on graphs, might either not directly induce a stepwise sequence of coarsened graphs, or even if they produce a stepwise coarsening hierarchy not enable precise control over the number of nodes in the coarsened graphs [23].

Edge pooling We downsample graphs by performing edge contractions closely following the pooling method proposed by Limbeck et al. [51]. For a graph $G = (X, E)$ and an edge $e = (u, v) \in E$, we define the graph G/e obtained from G by contracting the edge e to have vertex set $X \setminus \{v\}$ and edge set

$E \setminus \{(v,x)|x \in \mathcal{N}_G(v)\} \cup \{(u,x)|x \in \mathcal{N}_G(v)\}$, where $\mathcal{N}_G(v) = \{x \in G|(v,x) \in E\}$ denotes the neighbourhood of the vertex v in G. Note that we do not introduce any multiple edges or a self-loop on the remaining node u. To describe our downsampling process, let $G = (X,E)$ be a graph of size $n = |X|$ with $m = |E|$ edges that we want to downsample to the size $n-k$, where $0 < k \leq m$. To do this, we generate a sequence $\mathcal{E} = \{e_i\}_{1 \leq i \leq k}$ of edges in G to be contracted in that order. We choose this edge sequence based on edge scores that are either drawn uniformly at random from the unit interval for each edge, or we use edge scores based on a diversity measure as proposed by Limbeck et al. [51]. In the latter case, the score for an edge $e \in E$ is the difference in diversity $\mathrm{Div}(G) - \mathrm{Div}(G/e)$. After the scores are computed, we iteratively add the edge with the lowest score to the sequence $\mathcal{E}$ if that edge is not incident to a vertex incident to a previously contracted edge. If there are no more such edges, then we recompute all edge scores and repeat the procedure. With the generated edge sequence, we define the sequence of coarsened graphs $\{G_i\}_{n \geq i \geq n-k}$ where $G_n = G$ and for $n > i \geq n-k$ we set $G_i = G_{i+1}/e_{n-i+1}$. If node features are available, we average the node features of all original nodes that have been merged through these edge contractions. Further, we note that edge contraction pooling extends to mesh data and higher order shapes and is in fact often used to coarsen mesh data in practice [60]. Hence, we also apply edge contraction pooling to the triangulations in Section 4.5 while taking care to track triangles as well as edges through the coarsening process. Edge contraction pooling is thus used throughout all of our experiments.

Node upsampling For upsampling a graph $G = (X,E)$ of size $n = |X|$ to size $N > n$, we generate a sequence $\mathcal{V} = \{v_i\}_{1 \leq i \leq N-n}$ of $N-n$ nodes from G. This sequence determines the nodes that we upsample and the order in which we upsample them. We construct the sequence in such a manner that the difference between the number of times nodes are selected is no more than 1. To do this, let $m, r \in \mathbb{Z}_{\geq 0}$ be such that $N - n = m \cdot n + r$ and choose the first $m \cdot n$ nodes in $\mathcal{V}$ by doing m permutations of all nodes in the graph. For the remaining r nodes we choose them from the set of nodes X without replacement. This ensures each node is picked at least once before a previously upsampled node is chosen again. Going through the sequence $\mathcal{V}$ of nodes to upsample, for every $0 \leq i \leq N-n$ we add a new node u with the same neighbourhood as the sampled node v_i and also connect the sampled node to the new node. That is, we add an edge between the new node u and every node in $\mathcal{N}_G(v_i) \cup \{v_i\}$. If node features are available, we set the node feature of u to be the same as the node feature of v_i. This node-based upsampling procedure is used throughout all of our experiments other than in Section 4.5. Note that the node-based upsampling strategy aims to mimic and reverse the way edge contraction pooling downsamples nodes in a graph.

Triangular upsampling Given a triangular mesh, such as in Section 4.5, we might not want to just upsample nodes in the graphs as this loses information on the inherent shape of the data. Instead, we employ triangular upsampling and upsample each 3-clique in the graph, or each triangle in the triangulation. Specifically, we uniformly select triangles making sure that each triangle is picked

at least once before a previously upsampled triangle could be selected again. We then subdivide the selected triangles in order by adding a central vertex and connecting it to all corners. This way, we uniformly subdivide the surface of the meshes ($\mathcal{T}$), or the 3-cliques in the graphs (G). This upsampling strategy is used to compute diversity curves in Section 4.5 and account for the triangular nature of the data.

Metric choices To compute diversity curves, we consider the structural diversity of a graph to be dependent on a metric between nodes. Here, we will define and detail the distance choices explored in our paper. First, we define shortest-path distances, which are our default choice across this paper, as they are faster to compute than other structure-based metrics, such as heat kernel or diffusion distances, while performing well across our experiments and ablation studies.

Definition 2 (Shortest-path distance). *Given an undirected graph $G = (X, E)$, the* shortest-path distance *d is a metric that counts the minimum number of edges in any path between two nodes, $x_i, x_j \in X$, so that $d(x_i, x_j) = \infty$ if no such path exists and otherwise*

$$d(x_i, x_j) := \min_k \{k \in \mathbb{N} | \exists \text{ path } x_i = v_0, v_1, \ldots, v_k = x_j \text{ where } (v_l, v_{l+1}) \in E,\ v_l \in X\}. \tag{3}$$

When dealing with attributed graphs we can further define distances based on node features:

Definition 3 (Euclidean distance). *Given a graph $G = (X, E)$ associated with features $\mathbf{Y} \in \mathbb{R}^{|X| \times m}$ the* Euclidean distance *d between two nodes $x_i, x_j \in X$ with feature vectors $\mathbf{y_i}, \mathbf{y_j} \in \mathbb{R}^M$ is computed as*

$$d(x_i, x_j) := \|\mathbf{y_i} - \mathbf{y_j}\|_2 = \sqrt{\sum_{k=1}^{M} (\mathbf{Y}_{ik} - \mathbf{Y}_{jk})^2}. \tag{4}$$

Next, we define alternative structure-based metrics, which are computed from a graph's normalised Laplacian and have been used throughout related literature to study the geometry of graphs [12, 51]. Let $G = (X, E)$ be a graph, A its adjacency matrix, and D its degree matrix, whose diagonal entries are given by $D_{ii} = \sum_{j=1}^{|X|} A_{ij}$. The normalised graph Laplacian of G is given by $\hat{L} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$.

Definition 4 (Diffusion distance). *Given a graph $G = (X, E)$, its normalised Laplacian, $\hat{L}$, has positive eigenvalues $2 = \lambda_0 > \lambda_1 > \lambda_2 > \cdots > \lambda_{n-1} \geq 0$ with eigenvectors $\{\psi_l\}_l$. At time t this provides an embedding in Euclidean space given by*

$$\Phi_t(x) = (\lambda_1^t \psi_1(x), \cdots, \lambda_{n-1}^t \psi_{n-1}(x)) \text{ for } x \in X. \tag{5}$$

The diffusion distance *d between two nodes $x_i, x_j \in X$ at time t is defined as*

$$d(x_i, x_j) := \|\Phi_t(x_i) - \Phi_t(x_j)\|_2. \tag{6}$$

Definition 5 (Heat kernel distance). *Given a graph $G = (X, E)$, its normalised Laplacian, $\hat{L}$, has positive eigenvalues $2 = \lambda_0 > \lambda_1 > \lambda_2 > \cdots > \lambda_{n-1} \geq 0$ with eigenvectors $\{\psi_l\}_l$. The* heat kernel distance *d between two nodes $x_i, x_j \in X$ at time t is defined as*

$$d(x_i, x_j) := \sum_{l=0}^{n-1} \exp(-\lambda_l t \psi_l(x_i)\psi_l(x_j)). \tag{7}$$

Unless otherwise stated, we set the time parameter to $t = 1$ throughout or experiments. Further, we define a generalisation of the heat kernel distances to higher-dimensional shapes using the Hodge Laplacian on a simplicial complex [39]. We use existing code for computing the Hodge Laplacian.[1]

Definition 6 (Hodge heat kernel distance). *Given a simplicial complex $\mathcal{C}$, its k-th Hodge Laplacian, $\hat{L}$, has eigenvalues $\lambda_0 > \lambda_1 > \lambda_2 > \cdots > \lambda_{s-1}$ with eigenvectors $\{\psi_l\}_l$. We define the* topological heat kernel distance *d between two k-simplices $c_i, c_j \in \mathcal{C}$ at time t as*

$$d(c_i, c_j) := \sum_{l=0}^{s-1} \exp(-\lambda_l t \psi_l(c_i)\psi_l(c_j)). \tag{8}$$

Choice of coarsening levels Let $\mathcal{D}$ be a set of graphs for which we want to compute the diversity curves and compare them between each other. We select a set $\mathcal{I}$ of integers, which will represent the cardinalities at which we evaluate the diversity curves. That is, for any graph $G = (X, E) \in \mathcal{D}$ in our dataset we construct a sequence $\{G_i\}_{i \in \mathcal{I}}$ of graphs with $|G_i| = i$ based on the edge contraction and node upsampling described above and compute the diversity measure for every graph in the sequence $\{G_i\}_{i \in \mathcal{I}}$ to obtain the diversity curve. By default, we choose the cardinalities $\mathcal{I} = \{1, \ldots, \max_{G \in \mathcal{D}} |G|\}$ in our experiments. Note that for a graph G with precisely c connected components, we can not coarsen the graph to less nodes than c by edge contractions alone. In the case where $c > 1$, to still obtain values for every cardinality of the diversity curve, we linearly interpolate the diversity curve between $\mathrm{DivCurve}(G)_1 = 1$ and the lowest $i \in \mathcal{I}$ to which we could downsample the graph by edge contractions. This setup allows for a lot of flexibility, in particular, for datasets in which there are graphs of large cardinalities for which it is not feasible to compute the diversity measure directly, we can choose the maximum cardinality to be much smaller than the largest cardinality in the dataset. Furthermore, we can also choose to evaluate the diversity curves only at certain cardinalities if we are not interested in the finest possible resolution.

Algorithm We base our computations of structural diversity on `magnipy`[2], a Python library for the computation of metric space diversity measures [50], and

[1] https://github.com/tsitsvero/hodgelaplacians available under an MIT licence.
[2] Available at https://github.com/aidos-lab/magnipy under a BSD 3-Clause License.

`mag_edge_pool`[3], which provides code for computing the spread of a metric space on graphs, and for computing edge contraction pooling operations [51]. For using and manipulating graphs, we use the Python package NetworkX [26] released under a BSD-3-Clause license. Then, we implement the computations of diversity curves as described in Section 3.1 via Algorithm 1. A code implementation of our methods is available on GitHub.[4]

Algorithm 1 Diversity Curve Computation

Require: Graph $G = (X, E)$; evaluation scales $\mathcal{I} = \{n_{\min}, \dots, n_{\max}\}$; upsampling method $\mathcal{U}$; downsampling method $\mathcal{D}$; distance function $d(\cdot,\cdot)$; diversity measure $\mathrm{Div}(\cdot, d)$; number of repetitions $R \in \mathbb{N}$

Ensure: Diversity curve $\mathbf{v} \in \mathbb{R}^{|\mathcal{I}|}$

$\mathbf{v} \leftarrow \mathbf{0}^{|\mathcal{I}|}, \quad n \leftarrow |X|$
for $r = 1, \dots, R$ **do**
  $\mathcal{I}_\uparrow \leftarrow \{i \in \mathcal{I} : i > n\}$ in ascending order
  $\mathcal{I}_\downarrow \leftarrow \{i \in \mathcal{I} : i \leq n\}$ in descending order
  $G_\uparrow \leftarrow G$
  for each $i \in \mathcal{I}_\uparrow$ **do**
    $G_\uparrow \leftarrow \mathcal{U}(G_\uparrow, i)$ {Upsample to i nodes}
    $\mathbf{v}[i] \leftarrow \mathbf{v}[i] + \mathrm{Div}(G_\uparrow, d)$
  end for
  $G_\downarrow \leftarrow G$
  for each $i \in \mathcal{I}_\downarrow$ **do**
    $G_\downarrow \leftarrow \mathcal{D}(G_\downarrow, i)$ {Coarsen to i nodes}
    $\mathbf{v}[i] \leftarrow \mathbf{v}[i] + \mathrm{Div}(G_\downarrow, d)$
  end for
end for
$\mathbf{v} \leftarrow \mathbf{v}/R$
return $\mathbf{v}$

As default settings, we select the following choices: We set the evaluation interval $\mathcal{I} = \{1, 2, \dots, n_{\max}\}$ with $n_{\max}$ being the maximum number of nodes in a given graph dataset. Further, we use random edge contraction pooling to downsample the graph, and random node-based upsampling to upsample small graphs to larger sizes as described in Section A.1. For diversity computations, we choose the shortest-path metric between nodes as a distance function, and the spread of a graph for computing the diversity. Further, we repeat the coarsening across $R = 3$ repetitions and average the diversity curves across these, unless specified otherwise. Finally, whenever the number of connected components in a graph is larger than the minimum evaluation scale, we use linear interpolation to interpolate the value of diversity as described in Section 3.1.

[3] Available at https://github.com/aidos-lab/mag_edge_pool under a BSD 3-Clause License.

[4] The code for computing diversity curves is available at https://github.com/aidos-lab/diversity_curves/.

A.2 Compute resources

We run our experiments in a compute node with ×2 CPU AMD EPYC 7763 64-Core Processor and 1 TiB DRAM. We use these resources to speed up computation, however running experiments on a personal computer is also possible and does not represent a computational bottleneck.

A.3 Baseline methods

For computing the graph embeddings NetLSD [86], FEATHER [71], and Graph2Vec [58] we use the 'Karate Club' Python package [70] published under the MIT license (GPLv3). The Spectral Zoo embeddings we compute via the approximation used by Jin and Zafarani [34][5]. For the computation of the kernel methods we use the Python package GraKel (available under a BSD 3-clause License) [75]. For the computation of the TopER[6] method we use the implementation that has been made available with the publication [82]. We use the same approach for the persistent homology based graph embeddings as described in the classification experiment by Ballester and Rieck [1] and use their provided implementation.[7]

B Proofs and theoretical results

Here, we give an overview of all relevant theorems and proofs that detail and support our theoretical results. Specifically, we detail the properties of diversity curves in terms of their computational complexity and their expressivity at distinguishing graphs.

B.1 Computational complexity

Extending our analysis of the time complexity of diversity curve computations in Section 3.2, we outline the computational complexity of the specific steps involved in computing our methods.

Computing the diversity measure Let C_{Div} be the time complexity for computing the diversity measure of a graph $G = (X, E)$, we conduct our analysis with the spread as it was used in this work. The complexity C_{Div} consists of the time complexity C_{dist} for computing the pairwise distances between all vertices in the graph, and computing the spread. We use the shortest-path distance, which has time complexity $\mathcal{O}(|X|^3)$. In practice, shortest-path distances can be more efficiently computed than alternative graph distances, such as diffusion distances defined in Section A.1 and previously used by Limbeck et al. [51].

[5] Made available at https://github.com/shengminjin/EstimateSpectralMoments with the publication of their paper [34].
[6] Available at https://github.com/AstritTola/TopER under the MIT License
[7] Available at https://github.com/aidos-lab/PH_expressivity under a BSD 3-clause License.

Further, computing the spread has time complexity $\mathcal{O}(|X|^2)$. This is notably faster than computing alternative diversity measures, such as than magnitude of a graph, which has a time complexity $\mathcal{O}(|X|^3)$. Overall, for a general distance we get $C_{Div} = C_{dist} + \mathcal{O}(|X|^2)$, which in the case of shortest-path distance gives $C_{Div} = \mathcal{O}(|X|^3)$ using Floyd-Warshall algorithm. For our work, we choose to use exact computations of both metric space spread and shortest-path distances as described above to ensure computational accuracy. Nevertheless, we note that costs could further be reduced by approximating the shortest-path distance in around $C_{dist} = O(|X|^{2.032})$ time using a 2-approximation of all shortest-paths [3, 16]. Further, the costs of spread could be reduced via iterative normalisation or mini-batching using s subsets of size $|X_s|$ reducing the time complexity of approximating spread to $C_{Div} = C_{dist} + O(s \cdot |X_s| \cdot |X|)$ [17, 51].

Coarsening the graph Given a graph $G = (X, E)$, let C_{Coarse} be the time complexity for computing the sequence of coarsened graphs $\{G_i\}_{i \in \mathcal{I}}$ for a fixed set $\mathcal{I}$ of integers representing the cardinalities, so $|G_i| = i$ for any $i \in \mathcal{I}$. We denote by $n_{max} = \max_{i \in \mathcal{I}} i$ the maximum node size at which we compute the diversity and $n_{min} = \min_{i \in \mathcal{I}} i$ the minimum node size. We obtain the coarsened graphs by doing repeated edge contractions in G and upsampling nodes. The time complexity C_{Coarse} is composed of the time for computing the edge scores, sorting the edges according to their scores, and contracting the edges.

Computing the edge scores If we choose random edge scores, we need $\mathcal{O}(|E|)$ to get all of the edge scores. When using spread-based scores for determining the order of the edge contractions, it takes $\mathcal{O}(|E| \cdot C_{Div})$ time to compute all scores once [51].
Sorting the edges Sorting the edges according to their scores takes $\mathcal{O}(|E| \log(|E|)$ time.
Contracting the edges Contracting one edge has time complexity $\mathcal{O}(|X|)$ and we do this at most $\min(|E|, |X| - n_{min})$ times.
Repeating the edge scoring We might have to repeat the edge score computation and sorting of the edges since we restrict the edge contractions such that no two adjacent edges get contracted in the same iteration of the process of computing edge scores, sorting the edges, and contracting. Let r denote the number of times we have to recompute the edge scores and sort them. This number depends on n_{min} and how many edges can be contracted per iteration. If we want to coarsen the graph such that every edge is contracted, the number of iterations is $r = \log(n/\min(|E|, |X| - n_{min}))$ in the best case for which we can halve the size of the graph in each iteration, and $r = \min(|E|, |X| - n_{min})$ in the worst case where we can contract only one edge per iteration.
Upsampling a node We might also have to upsample graphs. Upsampling one node in a graph of cardinality n has time complexity $\mathcal{O}(n)$. We have to upsample $u = \max(0, n_{max} - |X|)$ nodes for a graph size of at most n_{max}.

Altogether, considering the steps detailed above we arrive at the following costs: For spread-based edge scores, we get a time complexity of $C_{Coarse} = \mathcal{O}(r(|E| \cdot C_{Div} + |E| \log(|E|)) + |E||X| + u \cdot n_{max}) = \mathcal{O}(r|E|(C_{Div} + \log(|E|)) + |E||X| + n_{max}^2)$.

This is more expensive than random edge selection. In fact, when generating and sorting random edge scores, we obtain the time complexity $C_{Coarse} = \mathcal{O}(r(|E| + |E|\log(|E|)) + |E||X| + u \cdot n_{max}) = \mathcal{O}(|E|(r \cdot \log(|E|) + |X|) + n_{max}^2)$.

Note that we can further speed up the coarsening process for random edge scores by not first generating random edge scores and sorting them, bur rather directly randomly permuting the edges. Then we can continue as above with the ordering of the edges given by the random permutation. This will improve the time complexity for computing the sequence of coarsened graphs using random edge scores to $C_{Coarse} = \mathcal{O}(|E|(r + |X|) + u \cdot n_{max}) = \mathcal{O}(|E|(r + |X|) + n_{max}^2)$. Note that in case $n_{max} < |X|$, we do not need to do any upsampling, and the time complexity for random edge contractions further reduces to $C_{Coarse} = \mathcal{O}(|E|(r + |X|))$.

Computing the diversity curves To compute diversity curves, we calculate the diversity measure at every cardinality $i \in \mathcal{I}$ for graphs with at most n_{max} nodes, which takes $\mathcal{O}(|\mathcal{I}| \cdot n_{max}^3)$ time when using spread. In total, the time complexity of computing the diversity curves is given by

- $C_{Coarse}+C_{Div} = \mathcal{O}(r|E|(C_{Div}+\log(|E|))+|E||X|+|\mathcal{I}|\cdot n_{max}^3)$ using spread-based edge scores,
- $C_{Coarse} + C_{Div} = \mathcal{O}(|E|(r \cdot \log(|E|) + |X|) + |\mathcal{I}| \cdot n_{max}^3)$ when sorting random edge scores, or
- $C_{Coarse} + C_{Div} = \mathcal{O}(|E|(r + |X|) + |\mathcal{I}| \cdot n_{max}^3)$ when permuting edges randomly.

Overall, we find that by using coarsening as one of their core components, diversity curves remain reasonably efficient to compute in practice. Further, our framework allows for investigating even faster coarsening methods or graph invariants in future work.

B.2 Diversity curves collapse to the same diversity

In this subsection, we describe how diversity curves behave if we coarsen graphs as much as possible by performing consecutive edge contractions until there are no more edges to contract. This illustrates that eventually, all diversity curves will reach the same values at low cardinalities.

Theorem 2. *Any two graphs G and H with the same number c of connected components will collapse to the same graph of cardinality c and their diversity curves agree for $i \in \{1, \dots c\}$, concretely* $\mathrm{DivCurve}(G)_i = \mathrm{DivCurve}(H)_i = i$. *Furthermore,* $\mathrm{DivCurve}(H)_{c+1} = \mathrm{DivCurve}(G)_{c+1} \neq c + 1$.

Proof. Consider a graph G with exactly c connected components. Doing edge contractions in a graph does not change its number of connected components. For any possible sequence of edge contractions, we will arrive at the graph K_c, the graph on c vertices with no edges. The spread of this graph is $\mathrm{Div}(K_c) = c$. In particular, a connected graph will collapse to K_1, the graph on 1 vertex, with spread $\mathrm{Div}(K_1) = 1$. For graphs with more than one connected component, we perform the linear interpolation, which gives us values of $\mathrm{DivCurve}(K_c)_i = i$ for $1 \leq i < c$. For any two graphs G and H with the same number c of connected

components we thus have $\text{DivCurve}(G)_i = \text{DivCurve}(H)_i = i$ for $i \in \{1, \dots c\}$. Note also that since the number of connected components is c, for a graph with $c+1$ vertices this is possible only if there exists precisely one edge. The spread of this graph is $c - 1 + \frac{2}{1+\exp(1)}$. Hence, we have shown that $\text{DivCurve}(H)_{c+1} = \text{DivCurve}(G)_{c+1} \neq c+1$.

B.3 Diversity curves as graph invariants

To study the invariance of the diversity curves under isomorphism, we consider the diversity curves under all possible edge contractions. To formally define what we mean by this, let us introduce some notation. For a graph G and a sequence $s_e = \{e_i\}_{1 \leq i \leq k}$ of $m \geq k \geq 0$ of edges in G, we denote by $G/\{e_i\}_{1 \leq i \leq k}$ the graph obtained from G after iteratively contracting the edges in the sequence s_e. We say that the sequence of coarsened graphs $\mathcal{G}_\mathcal{I} = \{G_i\}_{i \in \mathcal{I}}$ is induced by the edge sequence $\{e_i\}_{i \leq k}$ if $\mathcal{I} = \{n-k, \dots, n\}$ and $G_i = G/\{e_j\}_{1 \leq j \leq n-i}$ for any $i \in \mathcal{I}, i < n$ and $G_n = G$. We consider the multiset of all valid edge contraction sequences in the graph G of length $0 \leq k \leq m$ by $\mathcal{E}_k(G) = \{\{ \{e_i\}_{i \leq k} | \forall i \leq k \; e_i \in E(G/\{e_j\}_{j \leq i-1}\}\}$. We can now consider the multiset of diversity curves associated to each sequence of k edge contractions, that is $\{\{\text{DivCurve}(\mathcal{G}_\mathcal{I}) | \exists \{e_i\}_{i \in \mathcal{I}} \in \mathcal{E}_k$ such that $\mathcal{G}_\mathcal{I}$ is induced by $\{e_i\}_{i \in \mathcal{I}}\}\}$, which is what we mean by the multisets of diversity curves under all possible edge contractions.

Theorem 1. *For two isomorphic graphs, the multisets of the diversity curves over all possible edge contractions are equal.*

Proof. The statement follows immediately from the fact that the spread of a graph is invariant under graph isomorphism, and that for two isomorphic graphs $G \cong H$, contracting an edge e in G and the corresponding edge f in H gives two isomorphic graphs. Hence the multisets of sequences of coarsened graphs induced by all possible edge contractions $\{\{\mathcal{G}_\mathcal{I} | \exists \{e_i\}_{i \leq k} \in \mathcal{E}_k(G)$ such that $\mathcal{G}_\mathcal{I}$ is induced by $\{e_i\}_{i \leq k}\}\}$ and $\{\{\mathcal{H}_\mathcal{I} | \exists \{e_i\}_{i \leq k} \in \mathcal{E}_k(H)$ such that $\mathcal{H}_\mathcal{I}$ is induced by $\{e_i\}_{i \leq k}\}\}$ for up to any $m \geq k \geq 0$ edges are equal and therefore the multisets of diversity curves under all possible edge contraction sequences are also equal.

In practice, we cannot compute all possible of edge contraction sequences, but rather have to choose a subset of all possible edge contractions for each graph. Nevertheless, we can say the following:

Proposition 1 *Let G and H be two isomorphic graphs and $\{e_i\}_{1 \leq i \leq k}$ a sequence of edges in G. We consider the coarsened graphs $\mathcal{G}_\mathcal{I}$ induced by the sequence $\{e_i\}_{1 \leq i \leq k}$. There exists a sequence of coarsened graphs $\mathcal{H}_\mathcal{I}$ of H such that* $\text{DivCurve}(\mathcal{G}_\mathcal{I}) = \text{DivCurve}(\mathcal{H}_\mathcal{I})$.

Proof. Let $\phi : G \to H$ be an isomorphism between the two graphs. Let $\mathcal{H}_\mathcal{I}$ be the sequence of coarsened graphs induced by the edge sequence $\{\phi(e_i)\}_{1 \leq i \leq k}$. Since $G_n \cong H_n$, it iteratively also follows for any $n > i \geq n - k$ that $G_i \cong H_i$ and hence $\text{DivCurve}(\mathcal{G}_\mathcal{I}) = \text{DivCurve}(\mathcal{H}_\mathcal{I})$.

Hence, even if we can not compute all possible edge sequences, there always exists an edge sequence such that the diversity curves are equal, and if we choose the edges in a way that is invariant under the isomorphism, we will find the corresponding sequence.

B.4 Diversity curves increase expressivity

In this section, we want to study the expressivity of the diversity curves and spread itself. As detailed in Section 3.2 and Section B.3, we consider the diversity curves over all possible edge contractions for this discussion. Further, we define what it means to increase expressivity in Definition 1 following existing work by Lachi et al. [42], who previously showed that edge contraction pooling layers do not only preserve, but also increase expressivity. Our first result gives an example which shows that tracking the spread over a coarsening of graphs increases the expressivity compared to just considering the spread at the original cardinality of the graphs.

Theorem 1. *The multiset of the diversity curves over all possible edge contractions is more expressive than spread.*

Proof. It is clear that diversity curves are at least as expressive as spread, meaning that for any two graphs G and H with $\mathrm{Div}(G) \neq \mathrm{Div}(H)$, also $\mathrm{DivCurve}(G) \neq \mathrm{DivCurve}(H)$ since $\mathrm{DivCurve}(G)_{|G|} = \mathrm{Div}(G) \neq \mathrm{Div}(H) = \mathrm{DivCurve}(H)_{|H|}$ for any coarsening of G and H respectively, which implies that the multisets of the diversity curves over all possible edge contractions differ, since every diversity curve over a coarsening of G differs in at least one component from a diversity curve over any coarsening of H.
Consider the two graphs G_A, G_B in Figure S.1, where G_A is the graph on the left, and G_B is the so-called house graph on the right. They have equal spread $\mathrm{Div}(G_A) = \mathrm{Div}(G_B) = \frac{2}{1+3\exp(-1)+\exp(-2)} + \frac{3}{1+2\exp(-1)+2\exp(-2)} \approx 2.39$. The one-edge contractions of G_A are all isomorphic to the diamond graph D, the four-cycle with one diagonal edge, which has spread $\mathrm{Div}(D) = \frac{2}{1+2\exp(-1)+\exp(-2)} + \frac{2}{1+3\exp(-1)} \approx 2.02$. But the four-cycle C_4 is in the multiset of one-edge contractions of G_B and has spread $\frac{4}{1+2\exp(-1)+\exp(-2)} \approx 2.14$. This shows that there is an edge contraction sequence for G_B such that the diversity curve of the induced coarsening of the graphs differs from any diversity curve over a coarsening of G_A. Hence, the diversity curves over all possible edge contractions of G_A and G_B differ.
The diagram on the right of Figure S.1 shows the average spread over all possible edge contractions at each cardinality for the two graphs G_A and G_B. This demonstrates that the multisets of diversity curves differ across all possible choices of contracting one edge.

For our next statement, we first introduce another notion of comparing expressivity of two functions.

Definition 7. *For two functions φ and ψ on the set of graphs, we say that φ and ψ are* incomparable *if there exists graphs G_A, G_B, G_C, and G_D such that $\varphi(G_A) \neq \varphi(G_B)$ and $\psi(G_A) = \psi(G_B)$, and $\varphi(G_C) = \varphi(G_D)$ and $\psi(G_C) \neq (G_D)$.*

Proposition 2 *The spread and the Weisfeiler-Leman (1-WL) test are incomparable in terms of expressivity.*

Proof. We give concrete examples of two pairs of graphs. First, consider the disjoint union of two three-cycles $C_3 \sqcup C_3$ and the six-cycle C_6, which are known to be indistinguishable by 1-WL. The spread of the disjoint union of two three-cycles is $\mathrm{Div}(C_3 \sqcup C_3) = \frac{6}{1+2\exp(-1)} \approx 3.46$, whereas the spread of the 6-cycle is $\mathrm{Div}(C_6) = \frac{6}{1+2\exp(-1)+2\exp(-2)+\exp(-3)} \approx 2.92$. Figure S.2 illustrates this example. Second, consider again the two graphs in Figure S.1, which have the same spread as we have seen in the proof of Theorem 1. Those two graphs are distinguishable by 1-WL, since the degrees of the neighbourhoods of the vertices of degree 3 differ in the two graphs. Hence, we show that the spread and 1-WL are incompatible in terms of expressivity.

Proposition 3 *There exists a pair of graphs that is 1-WL-indistinguishable, indistinguishable by their spread alone, but distinguishable by their diversity curves over all possible edge contractions.*

Proof. Consider the two graphs G_C and G_D in Figure S.3, where G_C is the 6-cycle graph with the additional edges $(1,5)$ and $(2,4)$ on the left, and G_D is the 6-cycle graph with the additional edges $(1,4)$ and $(2,5)$ on the right. They are indistinguishable by the 1-WL test as the colouring obtained from the degrees of each node is already a stable colouring. Both graphs have the same spread, concretely $\mathrm{Div}(G_C) = \mathrm{Div}(G_D) = \frac{2}{1+2\exp(-1)+2\exp(-2)+\exp(-3)} + \frac{4}{1+3\exp(-1)+2\exp(-2)} \approx 2.66$. As is shown in Figure S.3 on the right, the average spread over the multiset of all possible edge contractions differ already after one edge contraction in each graph. Concretely, the house graph G_B is in the multiset of one-edge contractions of G_C, its spread is $\mathrm{Div}(G_B) = \frac{2}{1+3\exp(-1)+\exp(-2)} + \frac{3}{1+2\exp(-1)+2\exp(-2)} \approx 2.39$, whereas the multiset of one-edge contractions of G_D consists of two isomorphism classes of graphs, with spread $\frac{1}{1+2\exp(-1)+2\exp(-2)} + \frac{4}{1+3\exp(-1)+\exp(-2)} \approx 2.28$ and $\frac{1}{1+4\exp(-1)} + \frac{2}{1+2\exp(-1)+2\exp(-2)} + \frac{2}{1+3\exp(-1)+\exp(-2)} \approx 2.29$. This shows that there is an edge contraction sequence for G_C such that the diversity curve of the induced coarsening of the graphs differs from any diversity curve over a coarsening of G_D. Hence, the diversity curves over all possible edge contractions of G_A and G_B differ.

In total, these results show that if we have a pipeline that already distinguishes graphs on the level of 1-WL, then adding the diversity curves will increase the expressivity. Empirically, we demonstrate this in Section C.6 and Section 4.6.

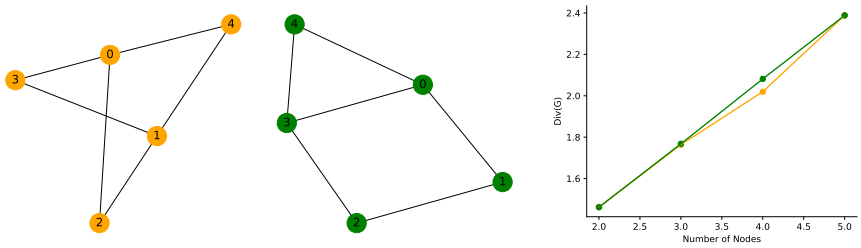

Figure S.1: Graphs with identical diversity, $\mathrm{Div}(G)$ as computed via spread using shortest-path distances, that can be distinguished through edge pooling by their diversity curves, $\mathrm{DivCurve}(G)$. This example is used in Theorem 1 and Proposition 2.

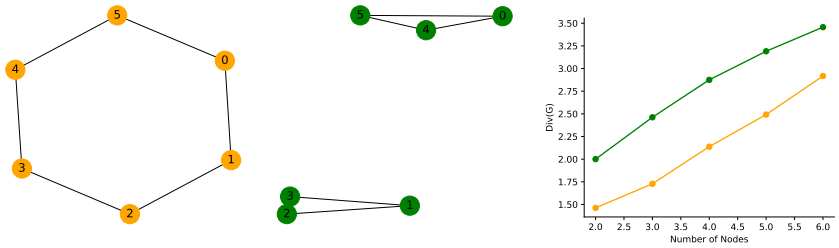

Figure S.2: Graphs that are 1-WL indistinguishable, but can be distinguished through their diversity, $\mathrm{Div}(G)$, or by their diversity curves, $\mathrm{DivCurve}(G)$. This example is used in Proposition 2.

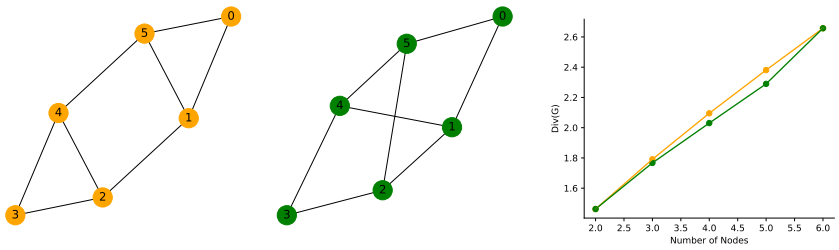

Figure S.3: Graphs that are 1-WL indistinguishable and have identical diversity, $\mathrm{Div}(G)$ as computed via spread using shortest-path distances, that can be distinguished through edge pooling by their diversity curves, $\mathrm{DivCurve}(G)$. This example is used in Proposition 3.

C Extended experiments

In the following section, we describe extended experimental details of all of our experiments, which describe how to reproduce our results. Further, we provide empirical expressivity results and ablation studies that support our main results.

C.1 Graph perturbations

The perturbation experiment in Section 4.1 applies one of the following perturbation scenarios: (i.) ADD_EDGE (▲) addition of an edge, (ii.) REMOVE_EDGE (■) removal of a non-bridge edge (iii.) REWIRE_EDGE (●) connect one of the endpoints of an edge to another vertex and (iv.) SWAP_EDGE (♦) given two edges (v_0, v_1) and (v_2, v_3) construct two new edges as (v_0, v_2) and (v_1, v_3). The perturbation degree refers to the percentage of edges that were perturbed. In each case this represents a different quantity. In (i.) the degree p is the percentage of edges remaining before the graph is fully-connected, i.e. $p = 1$ means enough edges have been added so that the graph is fully-connected. In (ii.) p is the percentage of edges that are not bridges, i.e. edges that do not result in the creation of more connected components when removed. In cases (iii.) and (iv.), p is the percentage of edges in each graph that are altered. We uniformly sample edges from the pool of available ones. Across each perturbation scenario and perturbation degree $p \in \{0.1, 0.2, \ldots, 1\}$, we apply the perturbation and track the norm of the average diversity curve across graphs, computed using shortest-path distances at node numbers between one and the maximum number of nodes in each dataset. Specifically, we consider the following datasets: `PLANAR` consists of 200 graphs with 64 nodes each, `LOBSTER` consists of 128 graphs with parameters $p_1 = p_2 = 0.7$ with a minimum number of 10 nodes and a maximum number of 100 nodes, `SBM` consists on 200 graphs with 20-40 nodes per community with 2-5 communities subset to all graphs with less than 60 nodes, and `COMM20` consists of 200 graphs of a stochastic block model with 2 communities [54][8]. The additional dataset `EGO` consists of 757 graphs extracted from Citeseer [33] [9]. We take a maximum of 80 graphs from each dataset, which is the test split except for the case of `SBM`, for which we consider 40 graphs with less than 60 nodes. Figure S.4 and Figure 2 show the results averaged over 5 random seeds along with the standard deviation.

C.2 Simulated graph distributions

Following our motivation of characterising graphs drawn from the same underlying distribution but across varying node sizes, we elaborate on our results from Section 4.2.

[8] We take the code for generating the datasets from https://github.com/KarolisMart/SPECTRE available under an MIT licence.

[9] We take the code for generating these dataset from https://github.com/yunhuijang/GEEL

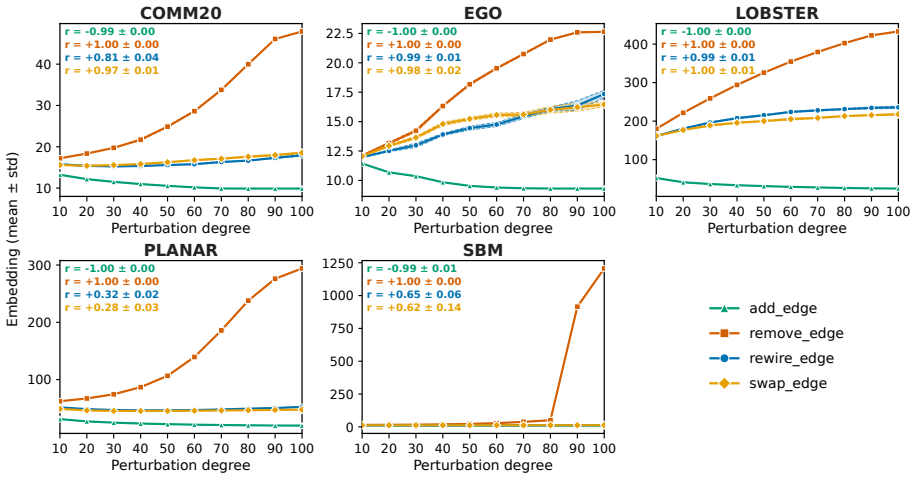

Figure S.4: Diversity curves track the change in graph structure across edge perturbations. We report the norm of the mean diversity curve per dataset by perturbation degree.

Distinguishing graph distributions In Section 4.2 , we investigate how well diversity curves can distinguish between graphs generated from different random graph distributions. Specifically, we generate graphs with different sizes using the Python package NetworkX [26] released under a BSD-3-Clause license. We choose the following four random graph distributions, for which we each generate 3 graphs for every node size $n \in \{10, \ldots, 29\}$:

- Erdős-Rényi (ER) graphs $G_{n,p}$ with edge probability parameter $p = 0.75$.
- Random partition (RP) graphs on n vertices, which have communities of fixed sizes and edge probabilities p_{in} for nodes in the same community and p_{out} for nodes in different communities. We fix the community sizes of the RP graphs such that we have 3 communities, each of about 1/3 the size of the graph and we set $p_{in} = 0.9$, and $p_{out} = 0.1$.
- Random geometric (RG) graphs, which are generated by choosing n nodes uniformly at random in the unit square and adding an edge for two nodes if their Euclidean distance is less than or equal to a parameter r, which we set to $r = 0.25$.
- Stochastic block model (SBM) of n vertices, which partitions the nodes into communities and adds edges according to the community the vertices belong to. As in the RP model, we choose probabilities $p_{in} = 0.8$ and $p_{out} = 0.05$ which specify the edge probabilities between vertices in the same cluster and different clusters respectively. For the communities we partition the node set into 4 communities each of about 1/4 of the graph's size.

Example graphs generated by each distribution are shown in Figure S.5. Based on the graphs generated as detailed above, we aim to assess the ability

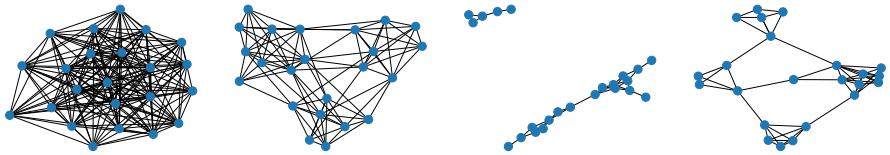

Figure S.5: Graphs drawn from the distributions ER, RP, RG, SBM used in the experiment to distinguish different distributions from each other.

of our method to distinguish between graph distributions. For computing the classification accuracies, we use a kNN-classifier with fixed $k = 5$ and 10-fold stratified grouped cross validation, where we group the graphs of the same node sizes together. To assess the clustering qualities, we report the silhouette score, Calinski–Harabasz index, and Davies–Bouldin index with the true class labels, as well as the adjusted rand indices (ARI) based on k-means clustering and a hierarchical clustering approach. For the k-means clustering we specify the correct number of clusters and use 'k-means++' for the initialisation method. For the hierarchical clustering we use agglomerative clustering where we specify the correct number of clusters and as a linkage criterion we use the average over all distances between the points in two clusters. All parameters which we do not specify here are used with their default setting as implemented in scikit-learn (version 1.5.2). To obtain uncertainties for the clustering metrics, we take the mean and standard deviation over 100 random resamples using 50% of the data points each.

Diversity curve computations. The kNN-classifier as well as the distances with which we compute clustering quality scores use the L^2-norm between the diversity curves viewed as vectorial embedding in $\mathbb{R}^{29}$ across evaluation scales from 1 to 29. Results are reported in Table 1. For visualisation as in Figure 3, we perform PCA on the diversity curves using the L^2-norm.

Spread functions as baselines. The spread function, as introduced by Willerton [89], measures how the spread of a metric space changes with respect to a scale factor t of the distance. Concretely, for a graph G viewed as a metric space with shortest-path distance d_G and a parameter $t > 0$, we define a metric space tG with the same underlying points as G and with the scaled metric $t \cdot d_G$. The spread function is the spread $\mathrm{Div}(tG)$ as a function of $t > 0$. As an ablation, we compare our diversity curves to this approach of the diversity function. For each graph, we compute the spread function at the graph's original size for $1 \leq t \leq 5$ on 20 evenly spaced evaluation scales and report the classification accuracies and clustering metrics using the L^2-norm between the spread functions.

Graph embedding baselines. We compare our results to relevant groups of unsupervised graph representation methods, namely graph kernels, spectral graph representations given by NetLSD [86] and Spectral Zoo [34], the topological

methods TopER [82] and a persistent homology based embedding [1], as well as the graph embeddings FEATHER [71] and Graph2Vec [58]. For all the graph embeddings, NetLSD, FEATHER, and Graph2Vec, we use the vectorial representations with Euclidean distances for computing the clustering accuracies and scores.

Kernel baselines. For the graph kernels, we compute the Kernel matrix K and use $-K + \max_{i,j} K_{i,j}$ as a pairwise distance to obtain clustering accuracies, silhouette scores, and the adjusted rand index with agglomerative clustering. We also use these pairwise distances for dimensionality reduction with MDS and compute all of our clustering metrics and accuracies also on the transformed data.

Topological baselines. For TopER we report for each of the filtrations described by Tola et al. [82] as sublevel filtrations individually and also combine all of them. We use the vectorial representations with Euclidean distances to compute clustering accuracies and scores. We use the same approach for the persistent homology based graph embeddings as described in the classification experiment by Ballester and Rieck [1]. Namely, we use different filtrations to compute persistence homology up to dimension two on the clique complexed obtained from the graphs by filling in every 3-clique, so we obtain a simplicial 2-complex. We transform the resulting persistence diagrams into a vectorial representation using their persistent images. We again use the Euclidean distances of the embeddings to compute clustering accuracies and scores.

The complete results can be found in Table S.1, which is an extended version of Table 1. Diversity curves perform similar in accuracy as the best performing baselines, and outperform the baselines in terms of almost all reported clustering metrics.

Distinguishing parameter choices Next, we evaluate how well our methods can distinguish between different parameters used to generate graphs. We look at one random graph model at a time and vary the parameter choices within the model. We again generate 3 graphs per class for every node size $n \in \{10, \dots, 29\}$ to understand how well the diversity curves can distinguish between these more subtle changes in the distribution while being robust to variations of the sizes of the graphs. We use the same graph models as in Section C.2 and vary the parameters as follows:

- Erdős-Rényi (ER) $G_{n,p}$ where we vary the edge probability $p \in \{0.1, 0.5, 0.9\}$;
- Random partition (RP) graphs where we vary the edge probabilities $(p_{in}, p_{out}) \in \{(0.9, 0.01), (0.9, 0.1), (0.9, 0.5)\}$ and fix the number of communities to be 3;
- Random geometric (RG) graphs where we vary the radius $r \in \{0.25, 0.5, 0.75\}$ determining which pairs of nodes are connected by an edge;
- Stochastic block model (SBM), where we vary the number of communities between 2, 3, and 4 roughly equally sized communities. The edge probabilities $p_{in} = 0.6$ and $p_{out} = 0.01$ are fixed.

Table S.1: Accuracies and clustering scores for distinguishing between the graph distributions ER, RP, SBM, and RG.

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI k-means	ARI agglomerative
Diversity Curves	Shortest Path	**0.904 ± 0.059**	0.423 ± 0.034	0.753 ± 0.060	191.237 ± 33.144	0.495 ± 0.071	0.313 ± 0.111
	Shortest Path PCA	0.892 ± 0.065	**0.432 ± 0.035**	**0.737 ± 0.060**	**194.064 ± 34.006**	**0.500 ± 0.075**	0.323 ± 0.123
Diversity Function	Spread Function Shortest Path	0.829 ± 0.029	0.033 ± 0.019	3.949 ± 0.390	12.517 ± 2.420	0.097 ± 0.050	0.095 ± 0.049
Embeddings	NetLSD	0.900 ± 0.033	0.055 ± 0.038	1.332 ± 0.112	38.799 ± 7.908	0.150 ± 0.054	0.092 ± 0.045
	FEATHER	0.750 ± 0.091	0.108 ± 0.023	1.914 ± 0.287	74.719 ± 8.443	0.293 ± 0.033	0.277 ± 0.038
	Spectral Zoo	0.721 ± 0.088	0.050 ± 0.031	2.875 ± 2.025	33.794 ± 4.979	0.257 ± 0.048	0.140 ± 0.082
	Graph2Vec	0.504 ± 0.094	-0.005 ± 0.001	7.365 ± 0.291	1.405 ± 0.102	0.023 ± 0.024	0.002 ± 0.008
Kernel	Kernel WL Optimal Assignment	0.900 ± 0.046	0.019 ± 0.004	-	-	-	0.259 ± 0.046
	Kernel Core Framework	0.842 ± 0.064	0.249 ± 0.026	-	-	-	0.446 ± 0.077
	Kernel Core Framework MDS	0.821 ± 0.038	0.266 ± 0.026	1.414 ± 0.178	91.392 ± 11.984	0.459 ± 0.049	**0.472 ± 0.060**
	Kernel WL Optimal Assignment MDS	0.796 ± 0.101	0.045 ± 0.020	3.491 ± 0.551	13.624 ± 2.199	0.185 ± 0.045	0.202 ± 0.057
	Kernel WL MDS	0.779 ± 0.072	0.239 ± 0.034	2.531 ± 0.443	21.090 ± 4.602	0.429 ± 0.061	0.273 ± 0.094
	Kernel Shortest Path	0.771 ± 0.073	0.284 ± 0.044	-	-	-	0.426 ± 0.057
	Kernel Shortest Path MDS	0.767 ± 0.046	0.274 ± 0.038	1.597 ± 0.169	57.105 ± 11.254	0.444 ± 0.065	0.423 ± 0.064
	Kernel WL	0.746 ± 0.094	0.213 ± 0.029	-	-	-	0.031 ± 0.090
	Kernel Pyramid Match	0.671 ± 0.039	0.002 ± 0.009	-	-	-	0.017 ± 0.016
	Kernel Pyramid Match MDS	0.588 ± 0.051	-0.009 ± 0.012	7.657 ± 3.929	7.242 ± 1.145	0.065 ± 0.038	0.045 ± 0.032
	Kernel Neighbourhood Hash	0.500 ± 0.062	-0.032 ± 0.006	-	-	-	0.002 ± 0.008
	Kernel Neighbourhood Hash MDS	0.421 ± 0.066	-0.066 ± 0.008	20.290 ± 10.776	0.683 ± 0.411	-0.007 ± 0.008	-0.003 ± 0.007
TopER	TopER Degree	0.758 ± 0.091	-0.064 ± 0.021	5.288 ± 1.902	31.191 ± 6.050	0.089 ± 0.019	0.089 ± 0.024
	TopER Closeness	0.746 ± 0.066	-0.093 ± 0.019	8.445 ± 7.621	25.482 ± 5.115	0.084 ± 0.020	0.076 ± 0.020
	TopER All Filtrations	0.700 ± 0.096	-0.055 ± 0.020	3.255 ± 0.499	27.534 ± 5.504	0.101 ± 0.025	0.073 ± 0.028
	TopER Popularity	0.696 ± 0.093	-0.022 ± 0.037	1.344 ± 0.129	29.667 ± 7.188	0.113 ± 0.076	0.062 ± 0.019
	TopER Forricci	0.646 ± 0.088	-0.067 ± 0.022	2.950 ± 1.701	31.450 ± 5.224	0.112 ± 0.024	0.112 ± 0.027
	TopER Olricci	0.592 ± 0.069	-0.099 ± 0.015	10.200 ± 9.354	13.468 ± 3.562	0.059 ± 0.020	0.043 ± 0.017
Persistent Homology	PH Vietoris Rips	0.783 ± 0.067	-0.010 ± 0.031	2.227 ± 0.175	16.156 ± 2.031	0.085 ± 0.028	0.053 ± 0.029
	PH Laplacian	0.783 ± 0.074	-0.132 ± 0.018	2.668 ± 0.247	11.701 ± 2.184	0.055 ± 0.021	0.034 ± 0.019
	PH Alpha Complex	0.742 ± 0.103	0.014 ± 0.021	1.992 ± 0.182	43.715 ± 6.740	0.202 ± 0.034	0.160 ± 0.054
	PH Fricci	0.742 ± 0.067	-0.099 ± 0.020	2.498 ± 0.220	30.874 ± 4.917	0.090 ± 0.031	0.043 ± 0.029
	PH Degree	0.725 ± 0.062	-0.160 ± 0.019	3.016 ± 0.353	8.580 ± 1.651	0.040 ± 0.020	0.018 ± 0.014
	PH Olricci	0.696 ± 0.085	-0.125 ± 0.023	2.945 ± 0.473	34.161 ± 5.567	0.092 ± 0.031	0.060 ± 0.023

For each random graph model, we compute the classification accuracies on the test set and clustering scores as described in Section C.2 for the same set of baseline methods. The full results can be found in Tables S.2, S.3, S.4, and S.5. Furthermore, we show the PCA plot of the diversity curves for each distribution in Figure S.6, which shows that diversity curves are able to cluster the graphs belonging to the same parameter setting clearly.

Table S.2: Accuracies and clustering scores for distinguishing parameter choices in ER graphs

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
Diversity Curves	Shortest Path	**1.000 ± 0.000**	0.679 ± 0.021	0.346 ± 0.018	272.892 ± 54.881	0.486 ± 0.053	0.444 ± 0.056
	Shortest Path PCA	**1.000 ± 0.000**	0.693 ± 0.021	**0.328 ± 0.019**	277.053 ± 56.295	0.485 ± 0.052	0.445 ± 0.056
Diversity Function	Spread Function Shortest Path	0.956 ± 0.069	0.133 ± 0.019	2.209 ± 0.327	29.456 ± 5.114	0.198 ± 0.097	0.189 ± 0.067
Embeddings	FEATHER	**1.000 ± 0.000**	0.574 ± 0.024	0.698 ± 0.053	186.237 ± 23.639	0.811 ± 0.134	0.800 ± 0.074
	Spectral Zoo	**1.000 ± 0.000**	0.335 ± 0.027	1.039 ± 0.107	161.364 ± 26.514	0.453 ± 0.115	0.439 ± 0.023
	NetLSD	0.972 ± 0.051	0.096 ± 0.027	2.044 ± 0.196	45.743 ± 8.983	0.076 ± 0.062	0.194 ± 0.062
	Graph2Vec	0.589 ± 0.083	-0.001 ± 0.002	7.240 ± 0.335	1.328 ± 0.107	0.001 ± 0.005	0.011 ± 0.024
Kernel	Kernel Shortest Path	**1.000 ± 0.000**	**0.793 ± 0.035**	-	-	0.857 ± 0.066	-
	Kernel Core Framework	**1.000 ± 0.000**	0.644 ± 0.019	-	-	0.778 ± 0.235	-
	Kernel Shortest Path MDS	0.994 ± 0.017	0.696 ± 0.034	0.482 ± 0.072	180.703 ± 47.049	0.758 ± 0.114	0.781 ± 0.064
	Kernel Core Framework MDS	0.994 ± 0.017	0.670 ± 0.018	0.445 ± 0.026	**356.006 ± 44.846**	**0.940 ± 0.093**	**0.942 ± 0.100**
	Kernel WL MDS	0.994 ± 0.017	0.557 ± 0.032	0.678 ± 0.098	97.450 ± 22.580	0.477 ± 0.044	0.765 ± 0.144
	Kernel WL	0.989 ± 0.022	0.513 ± 0.030	-	-	0.459 ± 0.060	-
	Kernel WL Optimal Assignment	0.978 ± 0.037	0.091 ± 0.009	-	-	0.586 ± 0.080	-
	Kernel WL Optimal Assignment MDS	0.917 ± 0.067	0.241 ± 0.030	1.973 ± 0.410	29.540 ± 4.918	0.478 ± 0.122	0.414 ± 0.136
	Kernel Pyramid Match	0.811 ± 0.132	0.035 ± 0.011	-	-	0.017 ± 0.028	-
	Kernel Pyramid Match MDS	0.756 ± 0.106	0.056 ± 0.015	11.737 ± 9.093	6.665 ± 1.152	0.050 ± 0.031	0.036 ± 0.045
	Kernel Neighbourhood Hash	0.611 ± 0.122	-0.007 ± 0.007	-	-	0.006 ± 0.014	-
	Kernel Neighbourhood Hash MDS	0.556 ± 0.082	-0.032 ± 0.010	14.111 ± 14.751	2.214 ± 1.022	0.012 ± 0.020	0.019 ± 0.025
TopER	TopER Forricci	0.906 ± 0.066	0.263 ± 0.044	1.196 ± 0.185	56.032 ± 10.377	0.180 ± 0.097	0.292 ± 0.077
	TopER Closeness	0.894 ± 0.058	0.057 ± 0.037	2.578 ± 0.876	20.609 ± 4.163	0.101 ± 0.053	0.133 ± 0.047
	TopER All Filtrations	0.894 ± 0.084	0.139 ± 0.037	2.062 ± 0.474	30.453 ± 5.249	0.145 ± 0.065	0.213 ± 0.067
	TopER Degree	0.883 ± 0.063	0.061 ± 0.036	2.729 ± 0.930	20.703 ± 4.083	0.110 ± 0.059	0.143 ± 0.044
	TopER Olricci	0.883 ± 0.072	0.162 ± 0.043	1.850 ± 0.481	34.409 ± 6.646	0.160 ± 0.064	0.202 ± 0.055
	TopER Popularity	0.856 ± 0.094	0.178 ± 0.042	2.006 ± 0.608	35.827 ± 5.863	0.211 ± 0.105	0.259 ± 0.070
Persistent Homology	PH Fricci	**1.000 ± 0.000**	0.128 ± 0.045	1.227 ± 0.090	18.835 ± 3.434	0.057 ± 0.029	0.105 ± 0.063
	PH Vietoris Rips	**1.000 ± 0.000**	0.213 ± 0.030	1.264 ± 0.075	29.377 ± 3.326	0.090 ± 0.061	0.139 ± 0.048
	PH Olricci	0.989 ± 0.022	0.176 ± 0.049	0.919 ± 0.047	40.144 ± 6.448	0.081 ± 0.048	0.159 ± 0.046
	PH Laplacian	0.900 ± 0.074	0.034 ± 0.045	2.152 ± 0.176	5.495 ± 1.112	0.021 ± 0.016	0.032 ± 0.020
	PH Degree	0.883 ± 0.091	0.029 ± 0.047	2.237 ± 0.127	5.371 ± 0.788	0.011 ± 0.013	0.037 ± 0.028
	PH Alpha Complex	0.767 ± 0.108	0.067 ± 0.029	2.016 ± 0.321	28.349 ± 6.228	0.124 ± 0.078	0.175 ± 0.048

Table S.3: Accuracies and clustering scores for distinguishing parameter choices in RP graphs.

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
Diversity Curves	Shortest Path	**0.978 ± 0.037**	0.576 ± 0.032	0.501 ± 0.038	285.485 ± 47.777	0.448 ± 0.062	**0.674 ± 0.139**
	Shortest Path PCA	**0.978 ± 0.037**	**0.593 ± 0.033**	**0.474 ± 0.041**	**295.427 ± 50.529**	0.458 ± 0.088	0.673 ± 0.141
Diversity Function	Spread Function Shortest Path	0.928 ± 0.070	0.035 ± 0.017	4.557 ± 0.679	6.023 ± 1.646	0.030 ± 0.032	0.021 ± 0.025
Embeddings	NetLSD	0.972 ± 0.037	0.145 ± 0.025	1.385 ± 0.088	32.110 ± 3.923	0.036 ± 0.056	0.382 ± 0.097
	Spectral Zoo	0.906 ± 0.066	0.117 ± 0.036	1.678 ± 0.327	31.343 ± 6.921	0.065 ± 0.064	0.229 ± 0.056
	FEATHER	0.861 ± 0.083	0.191 ± 0.029	1.502 ± 0.134	41.949 ± 7.816	0.175 ± 0.073	0.212 ± 0.051
	Graph2Vec	0.622 ± 0.060	0.003 ± 0.002	6.308 ± 0.298	2.655 ± 0.291	0.024 ± 0.044	0.176 ± 0.067
Kernel	Kernel WL MDS	0.972 ± 0.037	0.340 ± 0.036	1.317 ± 0.149	22.752 ± 4.198	0.109 ± 0.119	0.349 ± 0.128
	Kernel Shortest Path	0.967 ± 0.037	0.527 ± 0.042	-	-	**0.510 ± 0.088**	-
	Kernel Shortest Path MDS	0.967 ± 0.027	0.434 ± 0.036	0.990 ± 0.075	40.063 ± 5.645	0.133 ± 0.106	0.469 ± 0.118
	Kernel Core Framework	0.961 ± 0.050	0.345 ± 0.027	-	-	0.244 ± 0.104	-
	Kernel Core Framework MDS	0.950 ± 0.072	0.268 ± 0.035	1.484 ± 0.233	36.632 ± 6.913	0.289 ± 0.129	0.424 ± 0.077
	Kernel WL	0.939 ± 0.063	0.236 ± 0.033	-	-	0.016 ± 0.021	-
	Kernel WL Optimal Assignment	0.889 ± 0.082	0.012 ± 0.004	-	-	0.171 ± 0.065	-
	Kernel Pyramid Match	0.783 ± 0.098	0.038 ± 0.012	-	-	0.010 ± 0.022	-
	Kernel WL Optimal Assignment MDS	0.783 ± 0.098	0.061 ± 0.022	8.316 ± 6.006	8.633 ± 2.437	0.201 ± 0.101	0.189 ± 0.081
	Kernel Pyramid Match MDS	0.767 ± 0.121	0.026 ± 0.018	5.064 ± 0.996	5.336 ± 1.275	0.040 ± 0.027	0.040 ± 0.055
	Kernel Neighbourhood Hash	0.667 ± 0.090	-0.025 ± 0.007	-	-	0.007 ± 0.013	-
	Kernel Neighbourhood Hash MDS	0.611 ± 0.086	-0.050 ± 0.011	11.336 ± 6.653	1.569 ± 0.900	0.006 ± 0.018	0.020 ± 0.029
TopER	TopER All Filtrations	0.872 ± 0.093	0.026 ± 0.037	1.914 ± 0.364	26.117 ± 5.753	0.147 ± 0.042	0.139 ± 0.037
	TopER Closeness	0.783 ± 0.135	0.023 ± 0.038	1.954 ± 0.446	23.042 ± 5.265	0.112 ± 0.050	0.141 ± 0.043
	TopER Popularity	0.706 ± 0.138	-0.025 ± 0.047	2.128 ± 0.522	35.234 ± 8.414	0.158 ± 0.037	0.151 ± 0.034
	TopER Degree	0.678 ± 0.099	-0.028 ± 0.031	2.310 ± 0.930	23.952 ± 5.574	0.085 ± 0.041	0.111 ± 0.037
	TopER Forricci	0.672 ± 0.110	-0.043 ± 0.022	3.328 ± 1.586	11.954 ± 4.146	0.058 ± 0.029	0.060 ± 0.028
	TopER Olricci	0.572 ± 0.075	-0.050 ± 0.022	3.725 ± 1.580	9.520 ± 3.548	0.035 ± 0.029	0.052 ± 0.028
Persistent Homology	PH Alpha Complex	0.894 ± 0.072	0.183 ± 0.031	1.313 ± 0.142	49.927 ± 8.714	0.200 ± 0.090	0.319 ± 0.079
	PH Vietoris Rips	0.889 ± 0.079	0.083 ± 0.031	2.219 ± 0.219	15.271 ± 2.565	0.049 ± 0.040	0.077 ± 0.044
	PH Laplacian	0.889 ± 0.102	-0.092 ± 0.019	3.261 ± 0.337	11.802 ± 2.092	0.015 ± 0.015	0.073 ± 0.037
	PH Fricci	0.828 ± 0.088	-0.045 ± 0.022	2.239 ± 0.203	16.281 ± 3.202	0.035 ± 0.038	0.094 ± 0.035
	PH Degree	0.822 ± 0.096	-0.095 ± 0.019	3.074 ± 0.351	7.049 ± 1.181	0.008 ± 0.012	0.047 ± 0.036
	PH Olricci	0.817 ± 0.096	-0.014 ± 0.026	1.862 ± 0.140	23.389 ± 4.529	0.081 ± 0.039	0.105 ± 0.033

Table S.4: Accuracies and clustering scores for distinguishing parameter choices of RG graphs.

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
Diversity Curves	Shortest Path	**0.967 ± 0.044**	0.591 ± 0.032	0.483 ± 0.046	427.861 ± 97.530	0.486 ± 0.056	0.478 ± 0.129
	Shortest Path PCA	**0.967 ± 0.044**	**0.600 ± 0.033**	**0.468 ± 0.046**	**434.844 ± 100.391**	0.486 ± 0.057	0.480 ± 0.134
Diversity Function	Spread Function Shortest Path	0.889 ± 0.066	0.093 ± 0.017	2.660 ± 0.426	20.325 ± 3.851	0.110 ± 0.059	0.126 ± 0.059
Embeddings	FEATHER	0.939 ± 0.072	0.319 ± 0.033	1.173 ± 0.129	87.254 ± 13.845	0.456 ± 0.073	0.489 ± 0.063
	NetLSD	0.922 ± 0.067	0.068 ± 0.025	2.388 ± 0.317	61.887 ± 12.320	0.138 ± 0.057	0.232 ± 0.071
	Spectral Zoo	0.906 ± 0.111	0.183 ± 0.032	1.305 ± 0.187	77.332 ± 13.485	0.223 ± 0.145	0.365 ± 0.037
	Graph2Vec	0.578 ± 0.062	0.000 ± 0.002	7.192 ± 0.355	1.483 ± 0.132	0.012 ± 0.029	0.041 ± 0.038
Kernel	Kernel Core Framework MDS	0.939 ± 0.076	0.373 ± 0.030	0.891 ± 0.063	99.746 ± 12.842	0.460 ± 0.151	**0.574 ± 0.146**
	Kernel Core Framework	0.939 ± 0.107	0.386 ± 0.032	-	-	0.486 ± 0.139	-
	Kernel WL Optimal Assignment	0.928 ± 0.056	0.053 ± 0.007	-	-	0.398 ± 0.115	-
	Kernel WL MDS	0.911 ± 0.103	0.226 ± 0.037	3.921 ± 2.201	14.689 ± 4.101	0.025 ± 0.058	0.362 ± 0.130
	Kernel WL Optimal Assignment MDS	0.906 ± 0.075	0.175 ± 0.026	2.739 ± 0.885	22.119 ± 3.766	0.358 ± 0.123	0.350 ± 0.119
	Kernel WL	0.889 ± 0.129	0.185 ± 0.030	-	-	0.001 ± 0.005	-
	Kernel Shortest Path	0.889 ± 0.127	0.428 ± 0.049	-	-	**0.489 ± 0.080**	-
	Kernel Shortest Path MDS	0.861 ± 0.115	0.320 ± 0.043	1.768 ± 0.438	38.574 ± 7.686	0.321 ± 0.198	0.486 ± 0.055
	Kernel Pyramid Match	0.811 ± 0.075	0.040 ± 0.010	-	-	0.028 ± 0.038	-
	Kernel Pyramid Match MDS	0.672 ± 0.088	0.004 ± 0.014	9.754 ± 6.360	3.794 ± 1.051	0.041 ± 0.030	0.025 ± 0.037
	Kernel Neighbourhood Hash	0.578 ± 0.083	-0.014 ± 0.007	-	-	0.002 ± 0.008	-
	Kernel Neighbourhood Hash MDS	0.500 ± 0.119	-0.044 ± 0.007	16.075 ± 8.087	0.807 ± 0.521	-0.004 ± 0.011	-0.005 ± 0.010
TopER	TopER Popularity	0.833 ± 0.129	0.153 ± 0.046	2.013 ± 0.627	33.288 ± 6.509	0.189 ± 0.104	0.270 ± 0.055
	TopER Degree	0.806 ± 0.083	0.048 ± 0.039	2.340 ± 0.874	21.922 ± 4.646	0.099 ± 0.039	0.151 ± 0.039
	TopER All Filtrations	0.800 ± 0.100	0.094 ± 0.039	2.353 ± 0.745	25.906 ± 4.954	0.136 ± 0.075	0.212 ± 0.065
	TopER Closeness	0.800 ± 0.097	0.058 ± 0.039	2.473 ± 1.004	22.667 ± 4.479	0.105 ± 0.048	0.166 ± 0.053
	TopER Forricci	0.800 ± 0.051	0.119 ± 0.041	2.321 ± 0.838	32.456 ± 7.086	0.189 ± 0.078	0.235 ± 0.067
	TopER Olricci	0.650 ± 0.119	0.029 ± 0.043	7.906 ± 5.057	15.313 ± 3.525	0.073 ± 0.045	0.118 ± 0.043
Persistent Homology	PH Vietoris Rips	0.939 ± 0.084	0.114 ± 0.036	1.849 ± 0.119	19.354 ± 2.626	0.044 ± 0.030	0.115 ± 0.040
	PH Olricci	0.900 ± 0.074	0.138 ± 0.050	1.124 ± 0.075	28.997 ± 5.001	0.043 ± 0.036	0.146 ± 0.052
	PH Fricci	0.872 ± 0.079	0.101 ± 0.047	1.568 ± 0.118	13.261 ± 3.103	0.008 ± 0.014	0.097 ± 0.057
	PH Laplacian	0.811 ± 0.106	0.034 ± 0.047	2.030 ± 0.170	6.883 ± 1.267	0.010 ± 0.014	0.043 ± 0.029
	PH Alpha Complex	0.778 ± 0.090	0.105 ± 0.033	1.723 ± 0.272	32.251 ± 6.547	0.130 ± 0.065	0.233 ± 0.078
	PH Degree	0.739 ± 0.108	0.030 ± 0.046	2.535 ± 0.272	4.823 ± 0.962	0.012 ± 0.016	0.029 ± 0.024

Table S.5: Accuracies and clustering scores for distinguishing parameter choices in SBM graphs.

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
Diversity Curves	Shortest Path	**0.856 ± 0.079**	0.288 ± 0.036	1.062 ± 0.115	98.923 ± 15.813	0.397 ± 0.064	**0.427 ± 0.068**
	Shortest Path PCA	0.844 ± 0.092	**0.296 ± 0.037**	**1.038 ± 0.113**	**101.116 ± 16.296**	**0.406 ± 0.067**	0.426 ± 0.070
Diversity Function	Spread Function Shortest Path	0.794 ± 0.079	-0.000 ± 0.014	5.739 ± 1.063	3.491 ± 1.383	0.007 ± 0.021	0.002 ± 0.015
Embeddings	NetLSD	0.828 ± 0.080	0.112 ± 0.043	1.899 ± 0.432	20.365 ± 3.643	0.044 ± 0.027	0.095 ± 0.067
	FEATHER	0.656 ± 0.116	0.041 ± 0.024	2.383 ± 0.462	26.831 ± 5.108	0.137 ± 0.059	0.156 ± 0.040
	Spectral Zoo	0.633 ± 0.083	0.044 ± 0.035	2.676 ± 0.855	24.065 ± 4.737	0.104 ± 0.053	0.157 ± 0.043
	Graph2Vec	0.622 ± 0.085	0.000 ± 0.002	6.338 ± 0.297	1.882 ± 0.215	0.017 ± 0.043	0.090 ± 0.042
Kernel	Kernel WL Optimal Assignment	0.800 ± 0.075	0.009 ± 0.005	-	-	0.145 ± 0.056	-
	Kernel Core Framework MDS	0.700 ± 0.090	0.029 ± 0.018	3.843 ± 0.664	9.330 ± 2.056	0.037 ± 0.040	0.065 ± 0.035
	Kernel Shortest Path MDS	0.694 ± 0.076	-0.030 ± 0.013	25.418 ± 24.425	0.710 ± 0.570	0.038 ± 0.026	0.034 ± 0.026
	Kernel Core Framework	0.694 ± 0.103	0.029 ± 0.021	-	-	0.033 ± 0.038	-
	Kernel Pyramid Match	0.672 ± 0.091	-0.009 ± 0.007	-	-	-0.001 ± 0.014	-
	Kernel WL MDS	0.672 ± 0.115	-0.036 ± 0.015	25.410 ± 68.230	0.991 ± 0.586	0.002 ± 0.014	0.015 ± 0.019
	Kernel WL	0.661 ± 0.088	-0.017 ± 0.015	-	-	0.009 ± 0.016	-
	Kernel Shortest Path	0.639 ± 0.080	-0.062 ± 0.015	-	-	0.017 ± 0.024	-
	Kernel WL Optimal Assignment MDS	0.639 ± 0.083	-0.001 ± 0.017	4.799 ± 0.956	6.008 ± 1.697	0.101 ± 0.054	0.102 ± 0.046
	Kernel Pyramid Match MDS	0.622 ± 0.092	-0.009 ± 0.011	6.867 ± 2.211	3.509 ± 0.942	0.029 ± 0.030	0.023 ± 0.034
	Kernel Neighbourhood Hash	0.539 ± 0.117	-0.020 ± 0.005	-	-	-0.002 ± 0.010	-
	Kernel Neighbourhood Hash MDS	0.489 ± 0.065	-0.043 ± 0.006	27.497 ± 47.287	0.601 ± 0.419	-0.005 ± 0.009	-0.006 ± 0.010
TopER	TopER Degree	0.700 ± 0.079	-0.072 ± 0.016	5.790 ± 3.233	8.171 ± 3.373	0.027 ± 0.019	0.031 ± 0.022
	TopER Popularity	0.661 ± 0.091	-0.066 ± 0.023	4.008 ± 2.100	12.780 ± 3.919	0.050 ± 0.027	0.060 ± 0.030
	TopER Forricci	0.622 ± 0.078	-0.049 ± 0.022	3.615 ± 1.488	12.333 ± 4.063	0.042 ± 0.026	0.050 ± 0.027
	TopER All Filtrations	0.600 ± 0.065	-0.061 ± 0.019	4.282 ± 1.701	11.334 ± 3.697	0.050 ± 0.028	0.046 ± 0.027
	TopER Olricci	0.594 ± 0.108	-0.061 ± 0.019	4.774 ± 3.093	8.447 ± 3.225	0.035 ± 0.025	0.036 ± 0.025
	TopER Closeness	0.589 ± 0.120	-0.058 ± 0.028	4.114 ± 2.327	13.275 ± 4.404	0.053 ± 0.032	0.071 ± 0.031
Persistent Homology	PH Degree	0.717 ± 0.107	-0.096 ± 0.020	3.451 ± 0.386	6.487 ± 1.265	0.005 ± 0.010	0.056 ± 0.039
	PH Olricci	0.639 ± 0.115	-0.094 ± 0.026	2.729 ± 0.484	12.478 ± 2.966	0.014 ± 0.016	0.081 ± 0.035
	PH Laplacian	0.639 ± 0.143	-0.096 ± 0.018	3.627 ± 0.355	6.482 ± 1.446	0.009 ± 0.013	0.051 ± 0.043
	PH Vietoris Rips	0.611 ± 0.099	-0.085 ± 0.030	4.222 ± 1.024	4.581 ± 1.386	0.010 ± 0.013	0.016 ± 0.019
	PH Alpha Complex	0.606 ± 0.128	-0.053 ± 0.032	4.534 ± 2.876	8.124 ± 3.283	0.025 ± 0.029	0.065 ± 0.032
	PH Fricci	0.600 ± 0.124	-0.097 ± 0.027	3.291 ± 0.536	8.157 ± 1.917	0.005 ± 0.011	0.053 ± 0.033

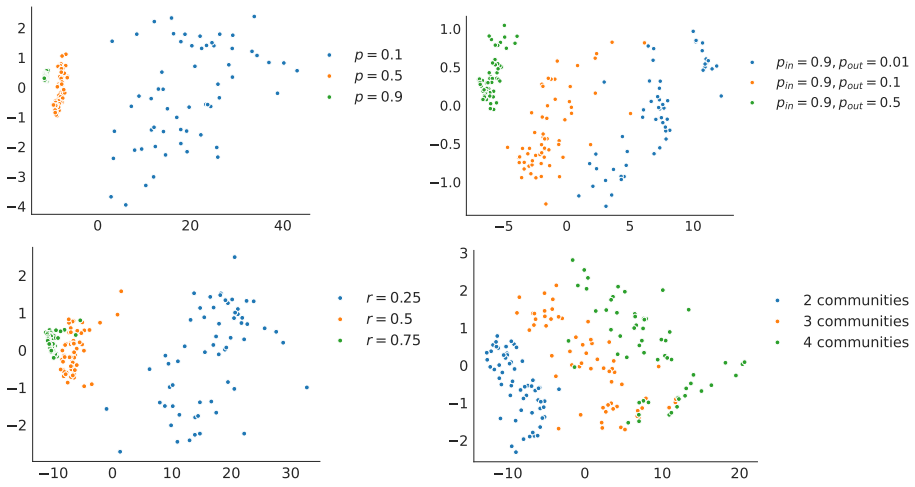

Figure S.6: PCA plots of the diversity curves based on shortest-path distance for ER, RP, RG, and SBM graphs.

C.3 Distinguishing between trajectory or cluster structures in single-cell graphs

Single-cell RNA sequencing datasets are often characterised as neighbourhood graphs that connect individual cells based on their molecular similarity. These single-cell graphs can further be characterised as displaying clustering-like or trajectory-like geometries [49]. Clustering-like graphs are dominated by clearly separated clusters of cells. Trajectory-like patterns are characterised by continuous transitions between cell states that form continuously connected trajectories on the single-cell graphs [72]. Knowing which type of geometry a graph displays directly gives actionably insights into which type of downstream analysis and visualisation methods are most appropriate for a given dataset. However, despite its practical relevance, the task of distinguishing between trajectory and clustering structures has not yet been solved. The most recent and to our knowledge only study on this task by Lim and Qiu [49] considers a collection of 169 single-cell datasets, out of which 85 show cluster-like structures and 84 show trajectory-like patterns. These datasets have been collected from a wide range of existing studies on evaluating trajectory inference methods [72], evaluating clustering methods [36, 41], and across a list of further datasets [4, 7–11, 13, 14, 18–22, 24, 25, 28, 29, 31, 35, 37, 38, 45–49, 52, 53, 56, 57, 61, 63, 64, 66, 68, 73, 74, 78–81, 83–85, 87, 90, 91, 95]. Out of the final collection of datasets, we consider two versions: First, we consider all 169 datasets altogether. Second, we consider a subset of 27 trajectory-like datasets whose trajectory structure has been verified gold-standard through biological validation [72] as well as 60 cluster-like datasets, which have been annotated as "organs" or "clusters" by Lim and Qiu [49]. Overall, this leaves a subset of 87 graphs, whose annotations are presumed to be less ambiguous.

Previously, Lim and Qiu [49] reach a classification accuracy of around 57% on the collection all 169 single-cell datasets as reported in their publication. Specifically, Lim and Qiu [49] used a set of 5 geometry-inspired scores to construct a landscape i.e. a shared representation of real single-cell datasets as well as simulated trajectory- or cluster-like datasets. They then use kNN-classification to predict the type of the real datasets based on the simulated examples, leading to a substantial proportion of graphs being misclassified. Given that Lim and Qiu [49] provide exact numbers on how many graphs per category were classified correctly or incorrectly in Figure 6.c of their manuscript, we use these numbers to simulate the performance of their Landscape approach across 5-fold stratified cross validation and report the estimated accuracies in Table S.6. Overall, we note that this reaches only about 57% accuracy. We aim to improve upon these results in our own study by investigating this classification task across a broader range of baseline methods.

Following Lim and Qiu [49], we download their dataset of processed single-cell datasets, and the set of scores they computed for each dataset, which have been made available with their codebase[10] and their publication. Preprocessing and

[10] Made available at https://github.com/pqiu/Quantifying_clusterness_trajectoriness with the publication of their paper [49].

quality control for the RNA sequencing datasets has been applied as described by Saelens et al. [72]. Lim and Qiu [49] further used PCA to reduce the dimensionality of each dataset to 20 principal components. Given the substantial size differences between datasets, ranging between around 19953 and 56 observations, we apply geometric sketching to subsample each dataset to at most 200 cells [30]. Further, we convert each dataset into a graph by connecting each cell to its 10 closest neighbours. Figure 5 then shows representative examples of four of these processed single-cell graphs, two trajectory-like graphs (A: cellbench-SC2_luyitian, GSE118767, simulated trajectory-like human cells [72, 80]; B: aging-hsc-young_kowalczyk, GSE59114, hematopoietic stem cell of mice [38, 72]), as well as two cluster-like graphs (C: mouse-cell-atlas-combination-5, cells from a mouse tissue atlas [28, 72]; D: GSE82187, mouse brain cells [20, 36]). Across these examples, we see that cluster-like graphs have distinct communities, while trajectory-like graphs display continuous transitions between cells. Our aim is now to distinguish these two types of graph through their structure.

We input these graphs into graph embedding models, or kernel methods. Further, we compute diversity curves with shortest-path distances, diffusion distances, or Euclidean distances between the scaled node features. The resulting mean diversity curves per group and their standard deviation are shown in Figure S.7. These diversity curves demonstrate that cluster-like single-cell graphs have on average higher structural diversity as measured by diversity curves, which use adjacency-based distances, such as shortest-path or diffusion distances. This is motivated by the fact that clusters of cells in these graphs are more distinct and more well separated structurally. These trends carry over to the PCA plot in Figure 5, which is computed from the standard scaled diversity curves using shortest-path as well as feature-based distances. We further match each of the four example graphs A to D to their corresponding representation in the PCA visualisation. Overall, this demonstrates visually that diversity curves distinguish the geometry of single-cell graphs.

Beyond the results reported in our main experiments, we further provide comparisons with feature-based baselines. Specifically, we apply mean pooling across the node features of the downsampled single-cell datasets as well as on the standard scaled version of these features. Comparing graph-based embeddings with these feature baselines then helps to assess to what extent the construction of neighbourhood graphs aids with distinguishing the geometries of single-cell datasets.

Table S.6 reports kNN-classification results for each method across 5-times repeated 5-fold stratified cross-validation, where 64% of the data is used for training, 16% for validation, and 20% for testing. The number of k nearest neighbours is tuned by grid search on the validation dataset in a range of $\{1, 3, 5, 7, 9\}$. We use a standard scaling transform on the diversity curves as this transformation on the training set helps highlight differences across graph sizes. Finally, Table S.6 shows the performance of each method, including the geometry scores by Lim and Qiu [49], in terms of both classification accuracies and AUROC scores.

Table S.6: Model performance metrics (mean ± std) at distinguishing between trajectory- and cluster-like single-cell graphs using kNN-classification across 5-fold stratified cross validation.

	Method	Accuracy - All Graphs	AUROC - All Graphs	Accuracy - Gold	AUROC - Gold
Diversity Curves	Diversity Curves All Metrics	**0.766 ± 0.062**	**0.810 ± 0.068**	**0.860 ± 0.062**	0.824 ± 0.068
	Shortest Path + Feature Dist.	0.759 ± 0.077	0.808 ± 0.081	**0.860 ± 0.077**	0.831 ± 0.081
	Diffusion Dist.	0.743 ± 0.061	0.786 ± 0.065	0.807 ± 0.061	0.795 ± 0.065
	Feature Dist.	0.721 ± 0.075	0.759 ± 0.089	0.843 ± 0.075	0.809 ± 0.089
	Shortest Path	0.682 ± 0.095	0.723 ± 0.115	0.773 ± 0.095	0.757 ± 0.115
Embeddings	NetLSD	0.731 ± 0.047	0.802 ± 0.057	0.823 ± 0.047	**0.848 ± 0.057**
	FEATHER	0.715 ± 0.071	0.771 ± 0.085	0.830 ± 0.071	0.824 ± 0.085
	Spectral Zoo	0.675 ± 0.074	0.734 ± 0.095	0.821 ± 0.074	0.799 ± 0.095
	Graph2Vec	0.622 ± 0.074	0.648 ± 0.088	0.741 ± 0.074	0.790 ± 0.088
Scores [49]	All Scores	0.716 ± 0.053	0.760 ± 0.061	0.804 ± 0.053	0.797 ± 0.061
	PH Score	0.697 ± 0.058	0.747 ± 0.066	0.829 ± 0.058	0.830 ± 0.066
	Reachability Score	0.632 ± 0.070	0.667 ± 0.087	0.727 ± 0.070	0.709 ± 0.087
	Vector Norm Score	0.626 ± 0.074	0.671 ± 0.102	0.694 ± 0.074	0.668 ± 0.102
	Diffusion Distance Score	0.520 ± 0.089	0.516 ± 0.100	0.611 ± 0.089	0.600 ± 0.100
	Landscape	0.570 ± 0.025	0.586 ± 0.037	-	-
	Ripley's K Score	0.532 ± 0.079	0.559 ± 0.088	0.598 ± 0.079	0.458 ± 0.088
Kernel	Kernel Pyramid Match	0.685 ± 0.092	0.721 ± 0.081	0.784 ± 0.092	0.757 ± 0.081
	Kernel WL	0.652 ± 0.079	0.699 ± 0.099	0.785 ± 0.079	0.773 ± 0.099
	Kernel Shortest Path	0.649 ± 0.085	0.690 ± 0.104	0.782 ± 0.085	0.770 ± 0.104
	Kernel Core Framework	0.649 ± 0.085	0.690 ± 0.104	0.782 ± 0.085	0.770 ± 0.104
	Kernel WL Optimal Assignment	0.600 ± 0.062	0.626 ± 0.083	0.679 ± 0.062	0.653 ± 0.083
	Kernel Neighborhood Hash	0.568 ± 0.082	0.588 ± 0.083	0.704 ± 0.082	0.667 ± 0.083
TopER	TopER All Filtrations	0.624 ± 0.086	0.669 ± 0.121	0.672 ± 0.086	0.649 ± 0.121
	TopER Olricci	0.586 ± 0.058	0.607 ± 0.073	0.649 ± 0.058	0.659 ± 0.073
	TopER Forricci	0.581 ± 0.053	0.600 ± 0.064	0.670 ± 0.053	0.610 ± 0.064
	TopER Popularity	0.516 ± 0.079	0.543 ± 0.090	0.662 ± 0.079	0.563 ± 0.090
	TopER Closeness	0.505 ± 0.069	0.520 ± 0.076	0.624 ± 0.069	0.476 ± 0.076
	TopER Degree	0.503 ± 0.090	0.521 ± 0.089	0.630 ± 0.090	0.480 ± 0.089
Node Features	Mean Pooling	0.619 ± 0.091	0.652 ± 0.103	0.661 ± 0.091	0.614 ± 0.103
	Standard Scaling + Mean Pooling	0.559 ± 0.068	0.573 ± 0.006	0.553 ± 0.068	0.585 ± 0.060

C.4 Characterising molecular graph datasets

To describe structural similarities between enzyme classes, we download the `PROTEINS` and `ENZYMES` datasets [6, 15], which are available under the TUDataset [55] collection through PyTorch Geometric (available under an MIT License). We map the classes from the `PROTEINS` graphs to the labels "enzymes" or "non-enzymes" by checking with the original publications [6, 15] and verifying the number of graphs in each class matches what is described by Borgwardt et al. [6]. Classes in `ENZYMES` are characterised by their Enzyme Commission (EC) number. We then compare the class of proteins that are not enzymes, to the classes of proteins that are enzymes across both datasets. To do so, we compute diversity curve using shortest-path distances on the graphs across node ranges form 1 to 34 nodes. The mean curves per class are visualised in Figure S.8 and highlight that non-enzymes have notably lower structural diversity on average than enzyme graphs. To verify this, we implement the statistical test for equality in the mean diversity curves as describe in Section 4.4. This case study demonstrates that diversity curves can be used for statistical testing, which highlights their utility for data-driven and uncertainty-aware analysis. Further, diversity curves directly allow for comparisons between differently sized graphs and graph distributions across varying cardinalities. Based on this, we believe there is promising potential

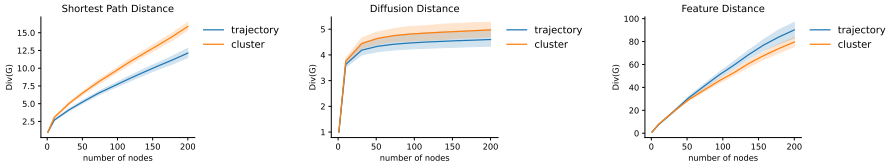

Figure S.7: Mean diversity curves per trajectory type computed using shortest-path distances, diffusion distances, or Euclidean distances based on standard scaled node features.

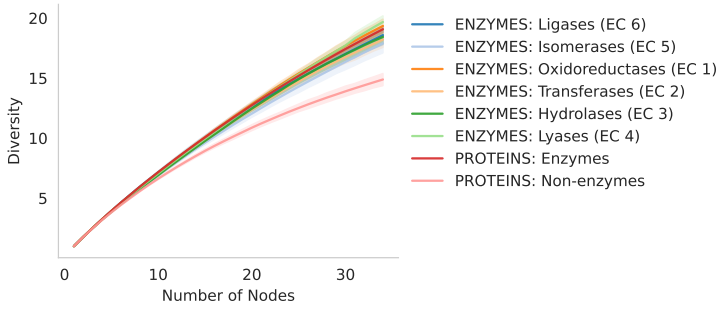

Figure S.8: Mean diversity curves per class in the `PROTEINS` and `ENZYMES` datasets.

for further applications to describe, explore, and compare the structural properties of graph datasets, both to describe the quality of graph datasets [12], as well as towards evaluating structural similarities for transfer learning tasks [32].

C.5 Distinguishing geometric shapes

To test whether our methods encode the inherent structure of data, we use the `MANTRA`[11] dataset [2], which has been proposed to benchmark the capability of geometric and topological deep learning models to distinguish between combinatorial triangulations of manifolds. Table S.7 specifically shows the number of triangulations of 2-manifolds in the `MANTRA` dataset, out of which triangulations of a Klein bottle (K), a torus (T^2), a sphere (S^2), and the real projective plane (RP^2) are associated with explicit homeomorphism types that characterise their shape. For our main experiments, we address the task of distinguishing between these four types.

We follow the experimental setup by Ballester et al. [2] to evaluate each classification model across 5-fold stratified cross validation, taking 20% of the data for testing, 16% for validation, and 64% for training. In this manner, we train Support Vector Machines with radial basis function kernels using default

[11] https://github.com/aidos-lab/MANTRA available under a BSD-3-Clause license.

Table S.7: The number of triangulations of 2-manifolds per shape for all classes with more than 100 objects in the `MANTRA` dataset.

Surface	$\#^4\mathrm{RP}^2$	$\#^3\mathrm{RP}^2$	$\#^5\mathrm{RP}^2$	Klein bottle	T^2	RP^2	$\#^6\mathrm{RP}^2$	$\#^2T^2$	S^2
Number	13695	11917	7053	4657	2247	1365	1022	868	308

parameters as implemented in scikit-learn (version 1.8.0) unless mentioned otherwise. For hyperparameter tuning, we use randomised grid search to select the cost parameter C in $[0.01, 100]$ and gamma in $[0.0001, 100]$ with log-uniform probabilities across 20 random iterations and 3-fold cross validation on the validation set. We let our models predict probabilities to report the one-vs-rest multi-class AUROC score on the test set, and further report the test accuracies to compare with the results by Ballester et al. [2]. All performance scores for the Cellular Transformer (CT) are cited directly from Table 11 in Ballester et al. [2], where this model is the best performing architecture at the intended task.

To compute diversity curves, we use edge contraction for coarsening as described throughout our paper. But we also note that some triangulations in `MANTRA` only have very few nodes making direct comparisons with larger graphs difficult. Therefore, we also employ upsampling operations via upsampling each 3-clique in the graph, or each triangle in the triangulation as described in Section A.1. Specifically, we uniformly select triangles making sure that each triangle is picked at least once before a previously upsampled triangle could be selected again, and subdivide the selected triangle by adding a central vertex and connecting it to all corners. This way, we uniformly subdivide the surface of the meshes ($\mathcal{T}$), or the 3-cliques in the graphs (G).

We report three versions of our results:

G: First, to train on graphs, denoted by G, we convert each triangulation into their one-skeleton. Then we compute diversity curves based on shortest-path distances on these graphs, for cardinalities between 1 and 15, the maximum number of nodes in the graphs. Note that for this version of the task, tori and Klein bottles will have identical 1-skeleton graphs, and thus be indistinguishable by their graph structure alone [2]. Based on this observation, we choose to report the performance of an optimal or dummy classifier in Table 3, whose performance we simulate by assuming that all classes other than tori get classified correctly with probability 1, while the triangulations of a torus get confidently misclassified as Klein bottles again with probability 1, which we use to simulate both the accuracy and the multi-class AUROC of this classifier baseline.

$\mathcal{T}$: Second, we input the triangulations themselves into our model, denoted by $\mathcal{T}$, which include information on nodes, edges, and faces allowing us to integrate higher order features into the predictions. Specifically, we compute heat kernel distances between faces to compute diversity curves on these higher order structures for cardinalities between 3 and 15. We then concatenate these curves with the previously computed graph-based diversity curves, as inputs for our prediction models.

Table S.8: Model performance (mean ± std) at distinguishing triangulations of 2-manifolds across stratified 5-fold CV.

Method	Version	Accuracy	AUROC
Diversity Curve	$\mathcal{T}$	1.000 ± 0.001	1.000 ± 0.000
Diversity Curve	G	0.928 ± 0.001	0.975 ± 0.002
Optimal Classifier	G	0.948 ± 0.000	0.941 ± 0.000
Diversity Curve	$\mathcal{T}_{\text{up}}$	0.3882 ± 0.006	0.841 ± 0.014

$\mathcal{T}_{\text{up}}$: Further, we apply barycentric subdivision to subdivide each mesh. As described by Ballester et al. [2], we then aim to address the task of training our models on the original dataset, but testing it on these subdivided meshes. Effectively, models that can solve this challenge have learned the intrinsic shape of each object while being robust to differences in scale and cardinality. This version of the task is denoted by $\mathcal{T}_{\text{up}}$. We compute diversity curves using both shortest-path distances on the graphs, and higher order heat kernel distances on the faces. For computing these higher-order distances we set $t = 20$ and $k = 2$ to compute a metric between triangles. Further, we use triangular upsampling to compute diversity curves for node sizes from 1 to 15 with a step size of one, and 20 to 60 with a step size of 5, extending the evaluation scales to capture trends at higher cardinalities in a size-aware manner. The final results for each version of the task are reported in Table 3. Further, in this scenario we apply standard scaling to the test and the training set independently to specifically account for domain-shift between the original shapes and upsampled shapes as we expect their diversity curves to follow the same overall trends but across varying ranges.

Finally, we extend the three tasks above to a larger set of triangulations in `MANTRA`. Instead of just taking the four shapes with known homeomorphism types, we also consider all other classes of 2-manifolds in `MANTRA` that consist of more than 100 elements. Results for this extended version are reported in Table S.8.

C.6 Expressivity

The `BREC`[12] dataset [88] consists of 400 pairs of graphs, which are used to test the expressivity of graph learning methods. All pairs of graphs are 1-WL indistinguishable, and 270 remain 3-WL indistinguishable [88]. We first collect results that have been reported in existing literature and summarise how many pairs of graphs each method can distinguish.

Wang and Zhang [88] report a range of graph learning models, for which we report the three best performing GNNs, I2-GNN and GSN, as well as Graphormer as a notably worse-performing transformer baseline. Relevant to our use of edge contraction pooling for comparing graphs, further work by Lachi et al. [42] has investigated under which conditions graph pooling operators do not only maintain

[12] https://github.com/GraphPKU/BREC available under an MIT licence.

but also improve expressivity. Their results prove that edge-based pooling in particular increases expressivity. In contrast, node-based pooling operators that do not incorporate any information beyond WL into their pooling choices, such as DiffPool which clusters nodes by utilising MPNNs, cannot improve expressivity. In fact, results by Lachi et al. [42] show DiffPool distinguishes no pair of graphs in `BREC` correctly, whereas EdgePool as well as their own edge-based pooling operator, EXpressive Pooling (XP), both improve expressivity on `BREC`. These results are summarised in Table S.9 as cited from the original publications [42, 88].

Further, we report results by Ballester and Rieck [1] on the performance of alternative geometric methods, namely persistent homology (PH) at distinguishing graphs from `BREC`. Specifically, we report the performance of different filtrations when filtering through all 2-simplices in the graphs, which represent 3-cliques, and computing persistent homology up to dimension 2, which tracks the evolution of connected components, circles, and two-dimensional voids across the filtration. See Ballester and Rieck [1] for a more detailed description.

Motivated by our theoretic results on the expressivity of diversity curves, we investigate the performance of our methods on `BREC` in practice. First, we check if spread itself can distinguish each pair of graphs by testing whether their difference in spread is above a given error tolerance [76], which we set to 10^{-5}. We try this for shortest-path, diffusion, and heat kernel distances on the graph, and note that the latter two metric choices can distinguish a larger set of graphs based on their structural diversity alone. We reason this is likely because both are computed from the normalised graph Laplacian, which can be more sensitive to subtle changes in graph structure, than shortest-path distances. Nevertheless, even structural diversity based on shortest-path distances can already distinguish 1-WL indistinguishable graphs.

Next, we extend our experiment to investigate whether edge contraction pooling can improve expressivity. To do so, we check each pair of graphs that cannot be distinguished through their spread alone, and iterate through all possible choices for contracting one edge, and compute the average diversity curve at $n-1$ for each graph $|G| = n$. Based on this, we again check if the difference in diversity curves at $n-1$ is larger than the error tolerance 10^{-5}. Across different choices of graph metric, we observe that edge contraction pooling enables diversity curves to distinguish more pairs of graphs than structural diversity alone. In particular, diversity curves based on shortest-path distances on the graph benefit from this coarsening approach.

For each comparison, we further check a reliability set to ensure that within the chosen error tolerance, pairs of isomorphic graphs are not distinguished. This empirical check holds across all comparisons verifying the validity of our results. From a theoretical perspective, we know that isomorphic graphs have the same deck of edge contractions [67], and thus have the same average diversity across all possible edge contraction choices as further described in Section B.3. Note that the expressivity results for diversity curves presented in Table S.9 represent a best case scenario where the sequences of coarsened graphs for which diversity is computed contain all possible edge contractions. This allows us to compare

Table S.9: Number of pairs from the BREC dataset distinguished via structural diversity or diversity curves. Compared to representative results from [1, 42, 76, 88].

Method	Details	Basic (60)	Regular (140)	Extension (100)	CFI (100)	All (400)
3 WL	(from [88])	60 (100.0%)	50 (35.7%)	100 (100.0%)	60 (60.0%)	270 (67.5%)
1 WL	(from [76])	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
I2-GNN	(from [88])	60 (100.0%)	100 (71.4%)	100 (100.0%)	0 (0.0%)	281 (70.2%)
KP-GNN	(from [88])	60 (100.0%)	106 (75.7%)	98 (98.0%)	11 (11.0%)	275 (68.8%)
GSN	(from [88])	60 (100.0%)	99 (70.7%)	95 (95.0%)	0 (0.0%)	254 (63.5%)
Graphormer	(from [88])	16 (26.7%)	12 (8.6%)	41 (41.0%)	10 (10.0%)	79 (19.8%)
CurvPool	(from [42])	35 (58.3%)	65 (46.4.0%)	53 (53.0%)	4 (4.0%)	157 (39.3%)
XP	(from [42])	48 (80.0%)	3 (0.02%)	72 (72.0%)	1 (1.0%)	124 (31.0%)
EdgePool	(from [42])	13 (21.7%)	0 (0.0%)	10 (0.1%)	0 (0.0%)	23 (5.8%)
DiffPool	(from [42])	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
PH Laplacian	(from [1])	60 (100%)	74 (53%)	100 (100%)	6 (6%)	216 (54%)
PH Ollivier–Ricci Curvature	(from [1])	60 (100%)	76 (54%)	92 (92%)	3 (3%)	208 (52%)
PH Forman–Ricci Curvature	(from [1])	59 (98%)	70 (50%)	59 (59%)	3 (3%)	172 (43%)
PH Degree Filtration	(from [1])	47 (78%)	55 (39%)	26 (26%)	3 (3%)	116 (29%)
PH Vietoris-Rips	(from [1])	31 (52%)	55 (39%)	11 (11%)	3 (3%)	84 (21%)
NetLSD	[70, 86]	60 (100.0%)	50 (35.7%)	100 (100.0%)	3 (3.0%)	213 (53.2%)
FEATHER	[70, 71]	52 (86.6%)	46 (32.9%)	5 (5.0%)	0 (0.0%)	103 (31.0%)
Div(G)	heat kernel distance	60 (100.0%)	50 (35.7%)	92 (92.0%)	3 (3.0%)	205 (51.3%)
Div(G)	diffusion distance	60 (100.0%)	49 (35.0%)	84 (84.0%)	3 (3.0%)	196 (49.0%)
Div(G)	shortest path metric	16 (26.7%)	13 (9.3%)	41 (41.0%)	6 (6.0%)	76 (19.0%)
DivCurve($\mathcal{G}_{\{n-1,n\}}$)	heat kernel distance	60 (100.0%)	65 (46.4%)	96 (96.0%)	3 (3.0%)	224 (51.3%)
DivCurve($\mathcal{G}_{\{n-1,n\}}$)	diffusion distance	60 (100.0%)	58 (41.4%)	94 (94.0%)	3 (3.0%)	215 (53.8%)
DivCurve($\mathcal{G}_{\{n-1,n\}}$)	shortest path metric	60 (100.0%)	48 (34.3%)	87 (87.0%)	8 (8.0%)	203 (50.8%)

the difference in diversity curves exactly rather than requiring a more complex statistical approach to account for randomness in the graph embeddings, which necessitates more stringent error control and can be unreliable [88].

For further comparison, we repeat the same procedure above on deterministic graph embedding methods, NetLSD and FEATHER, by computing the embedding vector for each graph and testing if the L^2-norm between embedding vectors is above the error threshold 10^{-5}.

Finally, Table S.9 reports how well each of the baseline methods or our diversity curves can distinguish pairs of graphs from BREC. Overall, we find that diversity curves perform better or competitively to existing geometry-aware baseline methods, such as persistent homology, or graph embedding methods, such as NetLSD and FEATHER. In fact, results improve upon any previously reported performance of edge pooling layers, and are better than certain baselines reported for notably more parametrised models, such as Graphormer. Overall, we thus see diversity curves improve the expressivity of diversity through edge contraction pooling in practice, while ensuring competitive performance on BREC.

C.7 Computational efficiency

To investigate computational costs empirically , we compare the runtime of our method to relevant baseline methods. We generate 30 Erdős-Rényi graphs

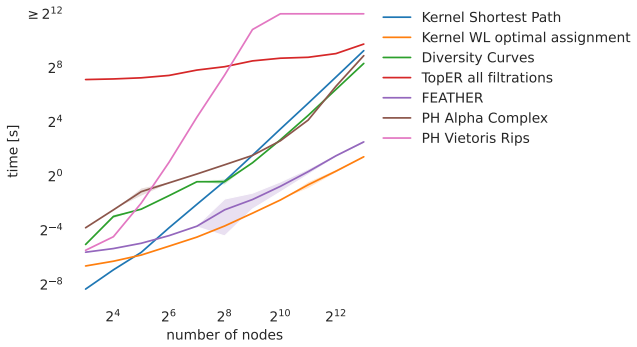

Figure S.9: Empirical runtimes of computing the diversity curves and baselines on ER graphs with increasing number of node sizes in a log-log-plot.

$G_{n,p}$ for $n \in \{2^i | i \in \{3, \dots, 13\}\}$ and $p = 1/n$. For each method, we average the runtime over 10 runs. We choose to evaluate diversity curves at the cardinalities $\mathcal{I} = \{20 \cdot i + 10 | 0 \leq i \leq 4\}$, hence the maximal cardinality at which we compute the diversity is $n_{max} = 90$. The results can be seen in Figure S.9. While certain variations of kernel methods and persistent homology are faster to compute than the diversity curves for this experimental setup, we note that we expect kernels to scale worse with a larger dataset size. Further, since our configuration of the diversity curves relies on the shortest-path distance in the graph, when comparing with other shortest-path based methods, such as persistent homology using the Vietoris-Rips filtration, we note that our method outperforms this baseline especially at larger graph sizes. Hence, we conclude that diversity curves can still be reasonably efficiently computed for larger graphs, and are slightly faster than for example the shortest-path kernel or persistent homology using the alpha complex. Because coarsening is an integral component of our framework, computational costs can be efficiently reduced by preprocessing graphs to smaller sizes, or evaluating diversity curves at fewer evaluation scales or lower node numbers. To compute persistent homology for this experiment, we build a clique complex of the graph by adding all 2-simplices, which correspond to 3-cliques in the graph, and compute the persistent homology on the filtrations of the clique complex up to dimension 2. This is the same setting that we used in the persistent homology baselines in the experiment from Section 4.2. The analysis of the empirical runtimes was performed on a compute node with ×2 CPU AMD EPYC 7763 64-Core Processor and 1 TiB DRAM

C.8 Ablation studies

Choice of distance and edge score We perform an ablation study to investigate the choice of our parameters, specifically the choice of graph distance to compute the spread and the choice of the edge scoring method. With the same experimental setup for distinguishing parameter choices in the ER, RP, RG, and

SBM graphs as described in Section C.2, we compute the diversity curves for the spread with shortest-path distance, diffusion distance, and resistance distance. Furthermore, for each choice of distance we vary the edge scoring method between random edge scores, edge scores based on the difference in spread, and an approximation of the edge scores based on the spread which approximates the distance between nodes after edge contraction by updating the distances on the original graph by taking the minimum distance to the original nodes [51]. We report all the accuracies and clustering metrics in Table S.10, Table S.13, Table S.12, and Table S.11. We further visualize the different silhouette scores in Figure S.10. We find that over our chosen random graph models, the shortest-path distance performs the best and the choice of edge scoring does not have any significant impact, which is why we decided to opt for random edge scoring in order to reduce the computational time complexity.

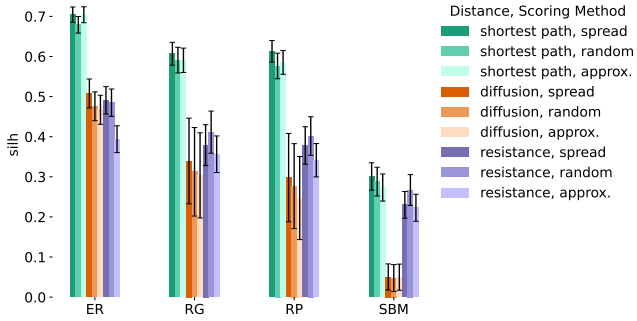

Figure S.10: Silhouette scores at distinguishing different parameter choices, varying the distance used to compute the spread and the edge scoring method.

Table S.10: Accuracies and clustering scores for distinguishing parameter choices for ER graphs.

Distance	Score	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
shortest path	spread	**1.000 ± 0.000**	**0.704 ± 0.019**	0.313 ± 0.017	**288.211 ± 57.504**	**0.491 ± 0.048**	0.442 ± 0.014
	random	**1.000 ± 0.000**	0.679 ± 0.021	0.346 ± 0.018	272.892 ± 54.881	0.486 ± 0.053	**0.444 ± 0.056**
	approx.	**1.000 ± 0.000**	0.704 ± 0.020	**0.312 ± 0.019**	282.266 ± 57.066	0.491 ± 0.047	0.441 ± 0.015
diffusion	spread	**1.000 ± 0.000**	0.508 ± 0.036	0.552 ± 0.056	69.180 ± 15.427	0.190 ± 0.091	0.271 ± 0.061
	random	**1.000 ± 0.000**	0.476 ± 0.036	0.560 ± 0.052	67.561 ± 15.040	0.194 ± 0.102	0.270 ± 0.063
	approx.	**1.000 ± 0.000**	0.467 ± 0.036	0.557 ± 0.049	69.157 ± 15.465	0.191 ± 0.094	0.271 ± 0.061
resistance	spread	0.994 ± 0.017	0.491 ± 0.034	0.579 ± 0.036	163.684 ± 34.156	0.390 ± 0.068	0.375 ± 0.033
	random	**1.000 ± 0.000**	0.485 ± 0.034	0.569 ± 0.034	146.921 ± 30.415	0.376 ± 0.074	0.365 ± 0.034
	approx.	**1.000 ± 0.000**	0.394 ± 0.033	0.791 ± 0.053	158.064 ± 33.425	0.368 ± 0.069	0.361 ± 0.033

Table S.11: Accuracies and clustering scores for distinguishing parameter choices for RP graphs.

Distance	Score	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
shortest path	spread	**0.978 ± 0.027**	**0.613 ± 0.027**	0.505 ± 0.037	**285.518 ± 36.276**	**0.480 ± 0.153**	**0.789 ± 0.188**
	random	**0.978 ± 0.037**	0.576 ± 0.032	**0.501 ± 0.038**	285.485 ± 47.777	0.448 ± 0.062	0.674 ± 0.139
	approx.	**0.978 ± 0.037**	0.585 ± 0.030	0.532 ± 0.043	246.763 ± 34.457	0.433 ± 0.135	0.740 ± 0.187
diffusion	spread	0.961 ± 0.056	0.298 ± 0.110	0.829 ± 0.155	55.944 ± 13.154	0.218 ± 0.041	0.218 ± 0.041
	random	0.967 ± 0.037	0.277 ± 0.106	0.846 ± 0.174	62.807 ± 15.787	0.218 ± 0.041	0.218 ± 0.041
	approx.	0.972 ± 0.028	0.247 ± 0.104	0.884 ± 0.168	55.656 ± 13.187	0.218 ± 0.041	0.218 ± 0.041
resistance	spread	0.972 ± 0.037	0.378 ± 0.047	0.777 ± 0.060	146.244 ± 27.256	0.346 ± 0.051	0.371 ± 0.045
	random	0.967 ± 0.044	0.402 ± 0.048	0.728 ± 0.058	181.881 ± 39.252	0.364 ± 0.062	0.384 ± 0.047
	approx.	0.956 ± 0.069	0.341 ± 0.042	0.847 ± 0.049	141.658 ± 28.112	0.346 ± 0.042	0.359 ± 0.053

Table S.12: Accuracies and clustering scores for distinguishing parameter choices for RG graphs.

Distance	Score	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
shortest path	spread	**0.983 ± 0.036**	**0.607 ± 0.028**	**0.445 ± 0.037**	393.501 ± 80.612	0.485 ± 0.051	**0.486 ± 0.120**
	random	0.967 ± 0.044	0.591 ± 0.032	0.483 ± 0.046	**427.861 ± 97.530**	**0.486 ± 0.056**	0.478 ± 0.129
	approx.	0.972 ± 0.037	0.591 ± 0.030	0.472 ± 0.042	381.289 ± 83.355	0.478 ± 0.053	0.462 ± 0.114
diffusion	spread	0.956 ± 0.042	0.339 ± 0.107	0.784 ± 0.184	124.151 ± 26.770	0.292 ± 0.107	0.357 ± 0.036
	random	0.950 ± 0.046	0.313 ± 0.110	0.845 ± 0.194	124.600 ± 27.846	0.287 ± 0.110	0.352 ± 0.037
	approx.	0.950 ± 0.046	0.304 ± 0.106	0.863 ± 0.188	124.430 ± 26.946	0.292 ± 0.107	0.357 ± 0.035
resistance	spread	0.950 ± 0.058	0.379 ± 0.051	0.798 ± 0.085	219.894 ± 57.742	0.439 ± 0.080	0.407 ± 0.029
	random	0.967 ± 0.044	0.411 ± 0.053	0.757 ± 0.087	254.085 ± 69.380	0.460 ± 0.068	0.412 ± 0.027
	approx.	0.944 ± 0.056	0.356 ± 0.046	0.840 ± 0.071	218.943 ± 57.164	0.438 ± 0.081	0.407 ± 0.028

Table S.13: Accuracies and clustering scores for distinguishing parameter choices of SBM graphs

Distance	Score	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI agglomerative	ARI k-means
shortest path	spread	**0.861 ± 0.062**	**0.301 ± 0.034**	1.122 ± 0.130	94.305 ± 13.963	0.371 ± 0.117	**0.432 ± 0.076**
	random	0.856 ± 0.079	0.288 ± 0.036	1.062 ± 0.115	**98.923 ± 15.813**	**0.397 ± 0.064**	0.427 ± 0.068
	approx.	0.861 ± 0.062	0.273 ± 0.034	1.154 ± 0.130	87.747 ± 13.267	0.376 ± 0.108	0.421 ± 0.074
diffusion	spread	0.783 ± 0.072	0.050 ± 0.033	2.158 ± 0.446	25.955 ± 5.913	0.153 ± 0.058	0.136 ± 0.035
	random	0.822 ± 0.054	0.048 ± 0.034	2.156 ± 0.457	25.918 ± 6.075	0.162 ± 0.047	0.160 ± 0.045
	approx.	0.783 ± 0.072	0.050 ± 0.033	2.161 ± 0.444	25.843 ± 5.908	0.152 ± 0.056	0.136 ± 0.035
resistance	spread	0.822 ± 0.060	0.230 ± 0.033	1.213 ± 0.128	70.056 ± 10.781	0.329 ± 0.095	0.374 ± 0.067
	random	0.817 ± 0.061	0.267 ± 0.038	**1.059 ± 0.106**	87.486 ± 15.035	0.361 ± 0.084	0.399 ± 0.078
	approx.	0.817 ± 0.050	0.223 ± 0.034	1.237 ± 0.128	66.452 ± 10.478	0.328 ± 0.094	0.364 ± 0.069

Table S.14: Accuracies and clustering scores of baseline structural summaries for distinguishing between the graph distributions ER, RP, SBM, and RG.

	Method	Accuracy	Silhouette Score	Davies-Bouldin	Calinski-Harabasz	ARI k-means	ARI agglomerative
Diversity Curves	Shortest Path	**0.904 ± 0.042**	0.427 ± 0.034	0.748 ± 0.060	194.340 ± 34.081	**0.511 ± 0.081**	0.311 ± 0.119
	Shortest Path PCA	0.888 ± 0.053	**0.436 ± 0.034**	**0.732 ± 0.060**	**197.355 ± 35.008**	**0.511 ± 0.080**	**0.317 ± 0.123**
Baseline	Summary Statistic	0.838 ± 0.035	-0.055 ± 0.017	3.250 ± 0.711	31.350 ± 5.313	0.111 ± 0.024	0.090 ± 0.033
	Degree Sequence	0.833 ± 0.046	-0.028 ± 0.012	3.459 ± 0.308	23.817 ± 3.310	0.135 ± 0.044	0.091 ± 0.034
	Average Shortest Path	0.762 ± 0.059	0.187 ± 0.040	23.435 ± 90.534	30.409 ± 7.871	0.378 ± 0.044	0.305 ± 0.050
	Clustering Coefficient	0.617 ± 0.095	0.041 ± 0.031	4.252 ± 4.872	34.702 ± 5.473	0.235 ± 0.055	0.129 ± 0.091
	Max Diameter	0.579 ± 0.101	0.243 ± 0.043	15.235 ± 31.362	33.867 ± 6.923	0.378 ± 0.091	0.169 ± 0.072
	Number of Edges	0.483 ± 0.121	-0.094 ± 0.021	3.666 ± 2.338	33.508 ± 5.866	0.113 ± 0.024	0.089 ± 0.029
	Number of Connected Components	0.454 ± 0.060	-0.134 ± 0.055	1.713 ± 0.409	68.197 ± 14.380	0.168 ± 0.032	0.186 ± 0.032
	Number of Nodes	0.250 ± 0.000	-0.083 ± 0.013	99.298 ± 196.613	0.532 ± 0.474	-0.012 ± 0.006	-0.012 ± 0.007

Comparison against simple structural baselines As another sanity check, we compare the performance of diversity curves in the simulation experiment from Section 4.2 as reported in Table S.14 against simple structural baselines. Concretely, we consider the number of nodes, number of edges, number of connected components, average clustering coefficient, the maximum diameter of each connected component, the average shortest path distance (averaged over all connected components), or the degree statistic (degree sequence padded with zeros to obtain the same length for all graphs) and use each of these invariants to perform the kNN-classification and compute the clustering scores. Additionally, we also report the accuracy and clustering performance when collecting all the structural invariants in a summary statistic. As expected, we see that the classification based on the number of nodes is based on chance, confirming that by construction the number of nodes is uninformative for the tasks at hand.

We further observe that all tested structural baselines perform worse in terms of the clustering metrics compared to diversity curves. Also in terms of accuracies, diversity curves outperform all structural baselines, with the degree statistic achieving the highest accuracy amongst the structural baselines. Hence, we verify that diversity curves provide information beyond standard graph summary statistics for these tasks.

Stability with respect to contraction sequence Because computing multiset of diversity curves over all possible edge contractions is infeasible in practice, particularly on large graphs, we have to limit ourselves to evaluating a smaller selection of possible edge contraction sequences. This will introduce some randomness in the selection of the edge contraction sequence. Even when choosing the edge contraction sequence by using a process that is based only on the graph structure, edges in symmetric graphs that cannot structurally be distinguished will obtain the same importance score. Hence, it is desirable to investigate the stability of diversity curves with respect to this randomness. In this ablation we study how stable the diversity curves are with respect to different edge contraction sequences and random seeds.

We follow the experiment from Section 4.2 to distinguish between the different random graph distributions and choose the same parameter settings as described in Section C.2. For each graph we compute its diversity curve based only on

one sequence of random edge contractions and repeat the computation over 10 different random seeds. As a first measure of the stability, we consider the distribution of the mean accuracies from the kNN-classification over the 10 different seeds. We find that the average and standard deviation of the mean accuracies obtained directly from the embedding through the diversity curves is 0.88 ± 0.009.

Secondly, we compare the differences between the diversity curves themselves. To do this, we compute the mean diversity curve of each graph over all 10 seeds and then for each graph in each seed we once get the L^2-distance to the mean curve of the same graph and to a randomly selected graph from a different graph distribution. The mean and standard deviation for the distance distributions of the diversity curve between the same graphs is 0.48 ± 0.33, while for the distance distribution of the diversity curve with a randomly chosen graph from a different distribution is 14.68 ± 8.99. Figure S.11 shows a histogram of the two distributions. To conclude, we find that empirically the diversity curves are stable with respect to different random edge contractions, validating our approach of considering a subset of the multiset of the diversity curves for our empirical evaluation.

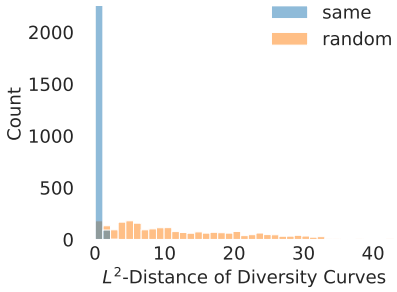

Figure S.11: Distribution of differences between the diversity curve of each graph in each seed and its mean diversity curve over all seeds ('same') or when comparing to a randomly chosen graph from another graph model ('random').

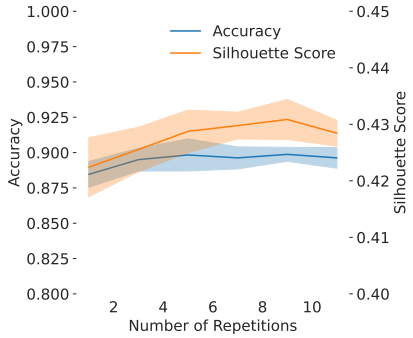

Figure S.12: Mean accuracy and silhouette score (avg±std) at distinguishing graph models over 10 different seeds for varying number of repetitions over which we compute and average the diversity curves.

Stability with respect to number of repeats As we saw, the choice of random edge contraction sequence introduces some variance in the diversity curves, thus in the experiments we choose to repeat the diversity curves over 3 different random edge contraction sequences and take the mean. This ablation addresses the influence on the number of repeats in computing the diversity curves. Concretely, we repeat the experiment on the simulated graph dataset for distinguishing between different distributions with the number of repetitions for which we compute diversity curves ranging over $\{1, 3, 5, 7, 9, 11\}$. For each fixed number of repetitions, we repeat over 10 different seeds and report the average

and standard deviation over the different seeds of the mean accuracy and mean silhouette score in Figure S.12. We find that that for only one repetition, the scores are lowest but that overall the diversity curves remain stable with respect to varying the number of repetitions.