Appendix

A Additional Literature on Graph Representation Learning

After the success of convolutional neural networks (CNN) on image-based tasks, graph neural networks (GNNs) have emerged as a powerful tool for graph classification and representation learning. Based on the spectral graph theory, Bruna et al. (2014) introduced a graph-based convolution in Fourier domain. However, complexity of this model is very high since all Laplacian eigenvectors are needed. To tackle this problem, ChebNet Defferrard et al. (2016) integrated spectral graph convolution with Chebyshev polynomials. Then, Graph Convolutional Networks (GCNs) of Kipf & Welling (2017a) simplified the graph convolution with a localized first-order approximation. More recently, there have been proposed various approaches based on accumulation of the graph information from a wider neighborhood, using diffusion aggregation and random walks. Such higher-order methods include approximate personalized propagation of neural predictions (APPNP) Klicpera et al. (2019), higher-order graph convolutional architectures (MixHop) Abu-El-Haija et al. (2019), multi-scale graph convolution (N-GCN) Abu-El-Haija et al. (2020), and Lévy Flights Graph Convolutional Networks (LFGCN) Chen et al. (2020). In addition to random walks, other recent approaches include GNNs on directed graphs (MotifNet) Monti et al. (2018), graph convolutional networks with attention mechanism (GAT, SPAGAN) Veličković et al. (2018); Yang et al. (2019), and graph Markov neural network (GMNN) Qu et al. (2019). Most recently, Liu et al. Liu et al. (2020) consider utilizing information on the node neighbors' features in GNN, proposing Deep Adaptive Graph Neural Network (DAGNN). However, DAGNN, and other state-of-the-art approaches, does not account for the important information on the shapes of the node neighborhoods.

B Further Background on Single Persistent Homology

Here, we give further details on single parameter persistent homology. To sum up, PH machinery is a 3-step process. The first step is the *filtration* step, where one can integrate the domain information into the process. The second step is the *persistence diagrams*, where the machinery records the evolution of topological features (birth/death times) in the filtration sequence of the simplicial complexes. The final step is the *vectorization* (fingerprinting), where one can convert these records to a function or vector to be used in suitable ML models.

i. Constructing Filtrations: As PH is the machinery to keep track of the evolution of topological features in a sequence, the most important step is inducing this nested sequence of simplicial complexes, $\Delta_0 \subset \Delta_1 \subset \cdots \subset \Delta_m$. This is the key step where one can inject valuable domain information into the PH process by using important domain functions. Two most common methods are *Sublevel/superlevel filtration* and *Vietoris-Rips (VR) filtration*. We already described sublevel filtration in Section 3.1.

VR filtration is another common method especially used for point clouds, where coarse geometry of the data set $\mathcal{X}$ play key role (Chazal & Michel, 2017). Let $\mathcal{X} = \{x_1, x_2, ..., x_N\}$ be the given data set. For a given threshold ϵ_i, one forms a Vietoris-Rips complex Δ_i by adding a k-simplex to X for any subset $\{x_{n_0}, x_{n_1}, ..., x_{n_k}\}$, where the pairwise distances are all $d(x_{n_r}, x_{n_s}) < \epsilon_i$. In particular, if a pair of points x_{n_0}, x_{n_1} has distance $< \epsilon_i$, then in the induced simplicial complex Δ_i, we add an edge $e_{n_0 n_1}$ between the corresponding vertices x_{n_0} and x_{n_1}. If three such points $x_{n_0}, x_{n_1}, x_{n_2}$ have pairwise distances $< \epsilon_i$, then we fill the triangle $e_{n_0 n_1} \cup e_{n_1 n_2} \cup e_{n_2 n_0}$ with a 2-simplex, and so on. For each ϵ_i, one obtains a simplicial complex Δ_i by using this procedure. Changing threshold values $\epsilon_1 < \epsilon_2 < \ldots < \epsilon_m$ results in a hierarchical nested sequence of simplicial complexes $\Delta_1 \subset \Delta_2 \subset \cdots \subset \Delta_m$ that is termed *the Vietoris-Rips filtration* of the data set $\mathcal{X}$. Note that VR filtration can also be considered as a sublevel filtration for the distance function to $\mathcal{X}$ from the ambient space $\mathbb{R}^d$, i.e., $f : \mathbb{R}^d \to \mathbb{R}$ with $f(y) = d(y, \mathcal{X})$ where $\mathcal{X} \subset \mathbb{R}^d$.

ii. Persistence Diagrams: The second step in PH process is to obtain persistence diagrams (PD) for the filtration $\Delta_0 \subset \Delta_1 \subset \cdots \subset \Delta_m$. As explained in Section 3.1, PDs are collection of 2-tuples, marking the birth and death times of the topological features appearing in the filtration, i.e. $\mathrm{PD_k}(\mathcal{X}) = \{(b_\sigma, d_\sigma) \mid \sigma \in H_k(\Delta_i) \text{ for } b_\sigma \leq i < d_\sigma\}$. This step is pretty standard and there are various software libraries for this task (Otter et al., 2017).

iii. Vectorizations: While PH extracts hidden shape patterns from data as persistence diagrams (PD), PDs, a collection of points in $\mathbb{R}^2$ by itself, are not very practical for statistical and ML purposes. Instead, the common techniques are faithfully representing PDs as kernels (Kriege et al., 2020) or vectorizations (Hensel et al., 2021). One can consider this step as converting PDs into a useful format to be used in the ML process as fingerprints of the dataset. This provides a practical way to use the outputs of PH in real-life applications. *Single Persistence Vectorizations* transform obtained PH information (PDs) into a function or a feature vector form which is much more suitable for ML tools than PDs. Common single persistence (SP) vectorization methods are Persistence Images (Adams et al., 2017), Persistence Landscapes (Bubenik, 2015), Silhouettes (Chazal et al., 2014), Betti Curves and various Persistence Curves (Chung & Lawson, 2019). These vectorizations define a single variable or multivariable function out of PDs, which can be used as fixed-size $1D$ or $2D$ vectors in applications, i.e., $1 \times n$ vectors or $m \times n$ vectors. For example, a Betti curve for a PD with n thresholds can also be expressed as $1 \times n$ size vectors. Similarly, Persistence Images is an example of $2D$ vectors with the chosen resolution (grid) size. See the examples given in Section 4.2 for further details.

C STABILITY

C.1 STABILITY OF SINGLE PERSISTENCE SUMMARIES

For a given PD vectorization, the stability is one of the most important properties for statistical purposes. Intuitively, stability question is whether a small perturbation in PD cause a big change in the vectorization or not. To make this question meaningful, one needs to define what "small" and "big" means in this context. Therefore, we need to define distance notion, i.e., metric in the space of persistence diagrams. The most common such metric is called *Wasserstein distance* (or matching distance) which is defined as follows. Let $PD(\mathcal{X}^+)$ and $PD(\mathcal{X}^-)$ be persistence diagrams two datasets $\mathcal{X}^+$ and $\mathcal{X}^-$ (We omit the dimensions in PDs). Let $PD(\mathcal{X}^+) = \{q_j^+\} \cup \Delta^+$ and $PD(\mathcal{X}^-) = \{q_l^-\} \cup \Delta^-$ where $\Delta^\pm$ represents the diagonal (representing trivial cycles) with infinite multiplicity. Here, $q_j^+ = (b_j^+, d_j^+) \in PD(\mathcal{X}^+)$ represents the birth and death times of a hole σ_j in $\mathcal{X}^+$. Let $\phi : PD(\mathcal{X}^+) \to PD(\mathcal{X}^-)$ represent a bijection (matching). With the existence of the diagonal $\Delta^\pm$ in both sides, we make sure the existence of these bijections even if the cardinalities $|\{q_j^+\}|$ and $|\{q_l^-\}|$ are different. Then, the p^{th} Wasserstein distance $\mathcal{W}_p$ defined as

$$\mathcal{W}_p(PD(\mathcal{X}^+), PD(\mathcal{X}^-)) = \min_{\phi} \left(\sum_j \|q_j^+ - \phi(q_j^+)\|_\infty^p \right)^{\frac{1}{p}}, \quad p \in \mathbb{Z}^+.$$

Then, a vectorization (function) $\varphi(PD(\mathcal{X}))$ is called *stable* if $\mathrm{d}(\varphi^+, \varphi^-) \leq C \cdot \mathcal{W}_p(PD(\mathcal{X}^+), PD(\mathcal{X}^-))$ where $\varphi^\pm = \varphi(PD(\mathcal{X}^\pm))$ and $\mathrm{d}(.,.)$ is a suitable metric on the space of vectorizations used. Here, the constant $C > 0$ is independent of $\mathcal{X}^\pm$. This stability inequality interprets as the changes in the vectorizations are bounded by the changes in PDs. Two nearby persistence diagrams are represented by nearby vectorizations. If a given vectorization φ holds such a stability inequality for some d and $\mathcal{W}_p$, we call φ a *stable vectorization* Atienza et al. (2020). Persistence Landscapes Bubenik (2015), Persistence Images Adams et al. (2017), Stabilized Betti Curves Johnson & Jung (2021) and several Persistence curves Chung & Lawson (2019) are among well-known examples of stable vectorizations.

C.2 PROOF OF THEOREM 4.1: STABILITY OF EMP SUMMARIES

Proof. As φ is a stable SP vectorization, for any $1 \leq i \leq m$, we have $\mathrm{d}(\varphi(\mathcal{G}_i^+), \varphi(\mathcal{G}_i^-)) \leq C_\varphi \cdot \mathcal{W}_{p_\varphi}(PD(\mathcal{G}_i^+), PD(\mathcal{G}_i^-))$ for some $C_\varphi > 0$ by Equation (1), where $\mathcal{W}_{p_\varphi}$ is Wasserstein-p

distance. Notice that the constant $C_\varphi > 0$ is independent of i. Hence,

$$\begin{aligned}\mathfrak{D}(\mathbf{M}_\varphi(\mathcal{G}^+), \mathbf{M}_\varphi(\mathcal{G}^-)) &= \sum_{i=1}^{m} \mathrm{d}(\varphi(\mathcal{G}_i^+), \varphi(\mathcal{G}_i^-)) \\ &\leq \sum_{i=1}^{m} C_\varphi \cdot \mathcal{W}_{p_\varphi}(PD(\mathcal{G}_i^+), PD(\mathcal{G}_i^-)) \\ &= C_\varphi \sum_{i=1}^{m} \mathcal{W}_{p_\varphi}(PD(\mathcal{G}_i^+), PD(\mathcal{G}_i^-)) \\ &= C_\varphi \cdot \mathbf{D}_{p_\varphi}(\{PD(\mathcal{G}_i^+)\}, \{PD(\mathcal{G}_i^-)\})\end{aligned}$$

where the first and last equalities are due to Equation (2) and Equation (3), while the inequality follows from Equation (1) which is true for any i. This concludes the proof of the theorem. □

D EMP FRAMEWORK

D.1 EMP FOR OTHER TYPES OF DATA

So far, to keep the exposition simple, we described our construction in the graph setup. However, our framework is suitable for various types of data. Let $\mathcal{X}$ be a an image data or a point cloud. Let $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$ be two filtering functions on $\mathcal{X}$. For example, it can be grayscale function for image data, or density function on point cloud data.

Let $f : \mathcal{X} \to \mathbb{R}$ be the filtering function with threshold set $\{\alpha_i\}_1^m$. Let $\mathcal{X}_i = f^{-1}((-\infty, \alpha_i])$. Then, we get a filtering of $\mathcal{X}$ as nested subspaces $\mathcal{X}_1 \subset \mathcal{X}_2 \subset \cdots \subset \mathcal{X}_m = \mathcal{X}$. By using the second filtering function, we obtain finer filtrations for each subspace $\mathcal{X}_i$ where $1 \leq i \leq m$. In particular, fix $1 \leq i_0 \leq m$ and let $\{\beta_j\}_{j=1}^n$ be the threshold set for the second filtering function g. Then, by restricting g to $\mathcal{X}_{i_0}$, we get a filtering function on X_{i_0}, i.e., $g : \mathcal{X}_{i_0} \to \mathbb{R}$ which produces filtering $\mathcal{X}_{i_0 1} \subset \mathcal{X}_{i_0 2} \subset \cdots \subset \mathcal{X}_{i_0 n} = \mathcal{X}_{i_0}$. By inducing a simplicial complex $\widehat{\mathcal{X}}_{i_0 j}$ for each $\mathcal{X}_{i_0 j}$, we get a filtration $\widehat{\mathcal{X}}_{i_0 1} \subset \widehat{\mathcal{X}}_{i_0 2} \subset \cdots \subset \widehat{\mathcal{X}}_{i_0 n} = \widehat{\mathcal{X}}_{i_0}$. This filtration results in a persistence diagram (PD) $PD(\mathcal{X}_{i_0}, g)$. For each $1 \leq i \leq m$, we obtain $PD(\mathcal{X}_i, g)$. Note that after getting $\{\mathcal{X}_i\}_{i=1}^m$ via f, instead of using second filtering function g, one can apply power filtration or Vietoris-Rips construction based on distance for each X_{i_0} in order to get a different filtration $\widehat{\mathcal{X}}_{i_0 1} \subset \widehat{\mathcal{X}}_{i_0 2} \subset \cdots \subset \widehat{\mathcal{X}}_{i_0 n} = \widehat{\mathcal{X}}_{i_0}$.

By using m PDs, we follow a similar route to define our EMP summaries. Let φ be a single persistence vectorization. By applying the chosen SP vectorization φ to each PD, we obtain a function $\varphi_i = \varphi(PD(\mathcal{X}_i, g))$ on the threshold domain $[\beta_1, \beta_n]$, which can be expresses as a $1D$ (or $2D$) vector in most cases (Section 4.2). Let $\vec{\varphi}_i$ be the corresponding $1 \times k$ vector for the function φ_i. Define the corresponding EMP $\mathbf{M}_\varphi$ as $\mathbf{M}_\varphi^i = \vec{\varphi}_i$ where $\mathbf{M}_\varphi^i$ is the i^{th} row of $\mathbf{M}_\varphi$. In particular, $\mathbf{M}_\varphi$ is a $2D$-vector (a matrix) of size $m \times k$ where m is the number of thresholds for the first filtering function f, and k is the length of the vector $\vec{\varphi}$.

D.2 EMP WITH OTHER FILTRATIONS

Weight filtration For a given weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, it is common to use edge weights $\mathcal{W} = \{\omega_{rs} \in \mathbb{R}^+ \mid \epsilon_{rs} \in \mathcal{E}\}$ to describe filtration. By choosing the threshold set similarly $\mathcal{I} = \{\alpha_i\}_1^m$ with $\alpha_1 = \min\{\omega_{rs} \in \mathcal{W}\} < \alpha_2 < \ldots < \alpha_m = \max\{\omega_{rs} \in \mathcal{W}\}$. For $\alpha_i \in \mathcal{I}$, let $\mathcal{E}_i = \{e_{rs} \in \mathcal{V} \mid \omega_{rs} \leq \alpha_i\}$. Let $\mathcal{G}^i$ be a subgraph of $\mathcal{G}$ induced by $\mathcal{V}_i$. This induces a nested sequence of subgraphs $\mathcal{G}_1 \subset \mathcal{G}_2 \subset \cdots \subset \mathcal{G}_m = \mathcal{G}$ (See top row in Figure 2).

In the case of weighted graphs, one can apply the EMP framework just by replacing the first filtering (via f) with weight filtering. In particular, let $g : \mathcal{V} \to \mathbb{R}$ be a filtering function with threshold set $\{\beta_j\}_{j=1}^n$. Then, one can first apply weight filtering to get $\mathcal{G}_1 \subset \cdots \subset \mathcal{G}_m = \mathcal{G}$ as above, and then apply f to each $\mathcal{G}_i$ to get a bilfiltration $\{\mathcal{G}_{ij}\}$ ($m \times n$ resolution). One gets m PDs as $PD(\mathcal{G}_i, g)$ and induce the corresponding $\mathbf{M}_\varphi$. Alternatively, one can change the order by applying g first, and get a different filtering $\mathcal{G}_1 \subset \mathcal{G}_2 \subset \cdots \subset \mathcal{G}_n = \mathcal{G}$ induced by g. Then, apply to edge weight filtration

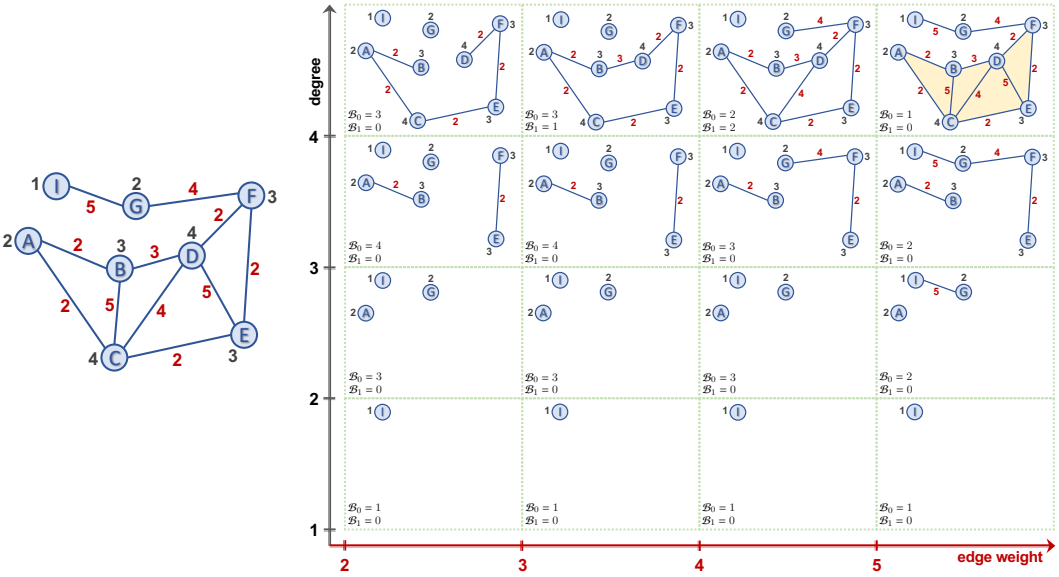

Figure 2: Multidimensional persistence on a graph network (original graph: left). Black numbers denote the degree values of each node whilst red numbers show the edge weights of the network. Hence, shape properties are computed on two filtering functions (i.e., degree and edge weight). While each row filters by degree, each column filters the corresponding subgraph using its edge weights. For each cell, lower left corners represent the corresponding threshold values. For each cell, $\mathcal{B}_0$ and $\mathcal{B}_1$ represent the corresponding Betti numbers.

to any $\mathcal{G}_j$, one gets a bifiltration $\{\widehat{\mathcal{G}}_{ji}\}$ ($n \times m$ resolution) this time. As a result, one gets n PDs as $PD(\mathcal{G}_i, \omega)$ and induce the corresponding $\mathbf{M}_\varphi$. The difference is that in the first case (first apply weights, then g), the filtering function plays more important role as $\mathbf{M}_\varphi$ uses $PD(\mathcal{G}_i, g)$ while in the second case (first apply g, then apply weights) weights have more important role as $\mathbf{M}_\varphi$ is induced by $PD(\mathcal{G}_j, \omega)$. Note also that there is a very different filtration method for weighted graphs by applying the following the following VR-complexes method.

Power (Vietoris-Rips) Filtration There is a highly different filtration technique using distances between the data points in the dataset. The technique is called *power filtration* for unweighted graphs Aktas et al. (2019), while it is called *Vietoris-Rips filtration* for other types of data Edelsbrunner & Harer (2010). The idea is for a point cloud $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, one uses the pairwise distances $d(x_r, x_s)$ to construct the simplicial complexes in the filtration. In particular, for a threshold set $\epsilon_1 < \epsilon_2 < \cdots < \epsilon_n = diam(\mathcal{X})$, one forms a Vietoris-Rips complex Δ_j by adding a k-simplex to $\mathcal{X}$ for any subset $\{x_{r_0}, x_{r_1}, \ldots, x_{r_k}\}$, where the pairwise distances are all $< \epsilon_j$. If a pair of points x_{r_1}, x_{r_2} has distance $< \epsilon_j$, then in the induced simplicial complex Δ_j, we add an edge between the corresponding vertices x_r and x_s. If three such points $x_{r_1}, x_{r_2}, x_{r_3}$ have pairwise distances $< \epsilon_j$, then we fill the triangle $e_{r_1 r_2} \cup e_{r_2 r_3} \cup e_{r_3 r_1}$ with a 2-simplex, and so on. This procedure induces in a hierarchical nested sequence of simplicial complexes $\Delta_1 \subset \Delta_2 \subset \ldots \subset \Delta_m$ that is termed *Vietoris-Rips filtration* induced by the point cloud $\mathcal{X}$. For unweighted graphs, one takes the vertex set $\mathcal{V}$ as the point cloud, and defines the distances $d(v_i, v_j)$ as the shortest distance in the graph where each edge has length 1. For weighted graphs, one can do the same by defining edge lengths induced by the weights.

One can adapt Vietoris-Rips filtrations to our EMP setting as follows. Start with a filtering function $f : \mathcal{X} \to \mathbb{R}$ with threshold set $\{\alpha_i\}_1^m$ and obtain $\mathcal{X}_1 \subset \mathcal{X}_2 \subset \cdots \subset \mathcal{X}_m = \mathcal{X}$ where $\mathcal{X}_i = f^{-1}((-\infty, \alpha_i])$. Then, apply Vietoris-Rips filtration to each $\mathcal{X}_{i_0}$ for threshold set $\{\epsilon_j\}_{j=1}^n$ which produces a filtration $\widehat{\mathcal{X}}_{i_0 1} \subset \widehat{\mathcal{X}}_{i_0 2} \subset \cdots \subset \widehat{\mathcal{X}}_{i_0 n}$ where $\widehat{\mathcal{X}}_{i_0 j}$ is the Vietoris-Rips complex of $\mathcal{X}_{i_0}$ for threshold ϵ_j. Construct $PD(\mathcal{X}_i, VR)$ of these filtrations for each $1 \leq i \leq m$. The following steps are same Section 4.2. For a given SP vectorization φ, let $\vec{\varphi}_i$ be the corresponding $1 \times k$ vector induced by $\varphi(PD(\mathcal{X}_i, VR))$ with domain $[\epsilon_1, \epsilon_n]$. Then, define EMP $\mathbf{M}_\varphi$ as $\mathbf{M}_\varphi^i = \vec{\varphi}_i$ where $\mathbf{M}_\varphi^i$ is the i^{th} row of $\mathbf{M}_\varphi$. Again, $\mathbf{M}_\varphi$ is a $2D$-vector (a matrix) of size $m \times k$ where m is the number of thresholds for the filtering function f, and k is the length of the vector $\vec{\varphi}$.

D.3 MULTIPARAMETER PERSISTENCE THEORY

Multipersistence theory is under intense research because of its promise to significantly improve the performance and robustness properties of single persistence theory. While single persistence theory obtains the topological fingerprint of single filtration, a multidimensional filtration with more than one parameter should deliver a much finer summary of the data to be used with ML models. However, multipersistence virtually has not reached any applications yet and remains largely unexplored by the ML community because of the technical problems. Here, we provide a short summary of these issues. For further details, Botnan & Lesnick (2022) gives a nice outline of current state of the theory and major obstacles.

In single persistence, the threshold space $\{\alpha_i\}$ being a subset of $\mathbb{R}$, is totally ordered, i.e., birth time $<$ death time for any topological feature appearing in the filtration sequence $\{\Delta_i\}$. By using this property, it was shown that "barcode decomposition" is well-defined in single persistence theory in 1950s [Krull-Schmidt-Azumaya Theorem Botnan & Lesnick (2022)–Theorem 4.2]. This decomposition makes the persistence module $M = \{H_k(\Delta_i)\}_{i=1}^{N}$ uniquely decomposable into barcodes. This barcode decomposition is exactly what we call a PD.

However, when one goes to higher dimensions, i.e. $d = 2$, then the threshold set $\{(\alpha_i, \beta_j)\}$ is no longer totally ordered, but becomes partially ordered (Poset). In other words, some indices have ordering relation $(1, 2) < (4, 7)$, while some do not, e.g., (2,3) vs. (1,5). Hence, if one has a multipersistence grid $\{\Delta_{ij}\}$, we no longer can talk about birth time or death time as there is no ordering anymore. Furthermore, Krull-Schmidt-Azumaya Theorem is no longer true for upper dimensions Botnan & Lesnick (2022)–Section 4.2. Hence, for general multipersistence modules barcode decomposition is not possible, and the direct generalization of single persistence to multipersistence fails. On the other hand, even if the multipersistence module has a good barcode decomposition, because of partial ordering, representing these barcodes faithfully is another major problem. Multipersistence modules are an important subject in commutative algebra, where one can find the details of the topic in Eisenbud (2013).

While complete generalization is out of reach for now, several attempts have been tried to utilize MP idea by using one dimensional slices in the MP grid in recent Carrière & Blumberg (2020); Vipond (2020). Slicing techniques use the persistence diagrams of predetermined one-dimensional slices in the multipersistence grid, and then combine (compress) them as one dimensional output Botnan & Lesnick (2022). In that respect, one major issue is that the topological summary highly depends on the predetermined slicing directions in this approach. The other problem is the loss of information when compressing the information in various persistence diagrams.

As explained above, MP approach does not have theoretical foundations yet, and there are several attempts to utilize this idea. In this paper, we do not claim to solve theoretical problems of multi-persistence homology, but offer a novel, highly practical multidimensional topological summary by advancing the existing methods. We use the grid directions in the multipersistence module as natural slicing directions and produce mutidimensional topological summary of the data. As a result, these multidimensional topological fingerprints are capable of capturing very fine topological information hidden in the data. Furthermore, in the case the data provides more than two very important filtering functions, our framework easily accommodates these functions and induces corresponding substructures. Then, our EMP model capture the evolving topological patterns of these substructures and summarize them in matrices and arrays which are highly practical output format to be used with various ML models.

E FURTHER DETAILS ON EXPERIMENTS

E.1 BENCHMARK DATASETS AND EXPERIMENTAL SETUP

In our experiments, we use 11 benchmark datasets for graph classification tasks (see Table 2). We have run our models for graph classification tasks on an 8-core DO droplet machine with Intel Xeon Scalable processors at a base frequency of 2.5 Ghz. Table 2 summarize the statistics of the datasets in our experiments.

The resolution of vectorization is the most significant parameter, which may impact the computational performance and results. As such, we use a fixed resolution to get consistent results in all experiments

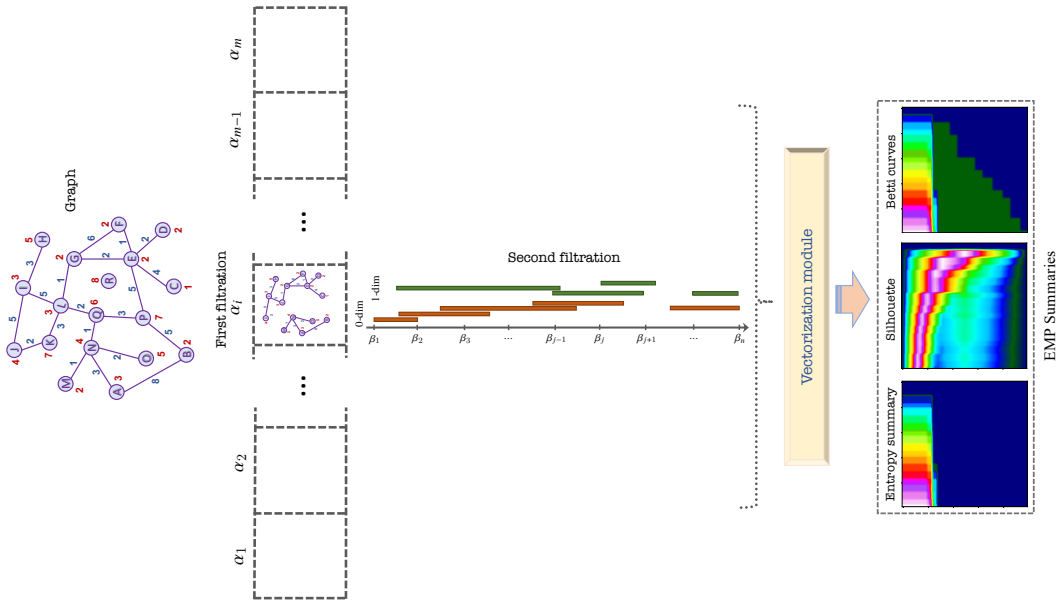

Figure 3: Illustration of the EMP framework for networks. Using the pair of filtering functions f, g we define non-decreasing thresholds $\{\alpha_i\}_1^m$ and $\{\beta_j\}_1^n$, respectively, based on node features, red, and edge features, blue. Both, filtrations and vectorizations run in parallel to better use computational resources and produce EMP representations in a timely manner.

Table 2: Summary statistics of the datasets.

Dataset	**# Graphs**	**Avg. $\lvert\mathcal{V}\rvert$**	**Avg. $\lvert\mathcal{E}\rvert$**	**# Class**	**Node Attr. (Dim.)**	**Edge Attr. (Dim.)**
BZR_MD	306	21.30	225.06	2	3	-
COX2_MD	303	26.28	335.12	2	3	-
DHFR_MD	393	23.87	283.02	2	-	1
MUTAG	188	17.93	19.79	2	-	-
PROTEINS	1113	39.06	72.82	2	1	-
IMDB-B	1000	19.77	96.53	2	-	-
IMDB-M	1500	13.00	65.94	3	-	-
REDDIT-B	2000	429.63	497.75	2	-	-
REDDIT-M-5K	4999	508.82	594.87	2	-	-

and consider time constraints on server usage. We use resolution size of 50×50 for each summary function, and the standard parameters set by the Gudhi library in Python [2]. The order of landscape summary function is set to 1 (max), whilst the power of weights is set to 1 for silhouette summaries.

F Computational Complexity

Computational complexity (CC) of persistence diagram $\mathrm{PD}_k(\Delta)$ is $\mathcal{O}(\mathcal{N}^3)$, where $\mathcal{N}$ is the number of k-simplices in Δ (Otter et al., 2017). CC of EMP summary $\mathbf{M}_\varphi^d$ depends on the vectorization φ used and the number d of the filtering functions one uses. If r is the resolution size of the multipersistence grid, then one needs $r^{(d-1)}$ single persistence diagrams to obtain $\mathbf{M}_\varphi^d$. Therefore, $\mathrm{CC}(\mathbf{M}_\varphi^d) = \mathcal{O}(r^{(d-1)} \cdot \mathcal{N}^3 \cdot C_\varphi(m))$ where $C_\varphi(m)$ is CC for φ and m is the number of barcodes in PD_k, e.g., if φ is persistence landscape, then $C_\varphi(m) = m^2$ and hence CC for EMP Landscape with $d = 2$ is $\mathcal{O}(r \cdot \mathcal{N}^3 \cdot m^2)$. In practice, r is a constant and m is small compared to $\mathcal{N}$, hence the complexity is again reduced to $\mathcal{O}(\mathcal{N}^3)$. On the other hand, as Betti numbers do not need PD_k to be

[2] https://gudhi.inria.fr/python/latest/

computed, it is possible to obtain much faster algorithms for EMP Betti Summary (Edelsbrunner & Parsa, 2014). Recently, Lesnick & Wright (2022) introduced a quite fast algorithm for EMP Betti summaries with $\mathcal{O}(\mathcal{M}^3)$ time where $\mathcal{M}$ is the rank of the multipersistence module with minimal representation.

G Ablation Study

Furthermore, to evaluate the performance of different types of EMP summaries (i.e., EMP Silhouette (EMP-S), EMP Entropy (EMP-E), and EMP Betti (EMP-B) with different dimensions (i.e., H_0: 0-dimensional topology, and H_1: 1-dimensional topology) and graph-based features (i.e., $f_{\mathcal{V}}$: node features, and $f_{\mathcal{E}}$: edge features; see more details in Section 5.2 and Appendix E.1). We include comparative statistics using all combinations of input variables, i.e. ablation study, for 3 datasets in our analysis: BZR-MD (Table 3), DHFR-MD (Table 4), and REDDIT-BINARY (Table 5).

Each cell shows the classification accuracy (in % $\pm$ standard deviation) when using different combinations of variables. The best accuracy result is highlighted using bold font. A row/column named as 'none' means that either graph-based features, or EMP summaries where not used to compute the experiments. As such, the top row shows all the accuracy results without using any EMP topological features. The first/left column shows the results only using EMP topological features. The cell in the bottom right corner, of each EMP group, contains the accuracy results using all graph-extracted/EMP features available.

We can observe that: (i) best results contains 1-dimensional EMP summaries, demonstrating the necessity of capturing higher-order structures (e.g., triangles/cycles), (ii) the choice of the EMP summary can significantly affect the performance, and (iii) the addition of EMP topological features generally improves accuracy of versions without EMP summaries, thus, demonstrating the importance of modeling global and topological graph structures.

Table 3: Ablation Study on BZR_MD dataset.

	none	$f_{\mathcal{V}}$	$f_{\mathcal{E}}$	$f_{\mathcal{V}} + f_{\mathcal{E}}$
none	-	67.124 $\pm$ 1.959	58.714 $\pm$ 0.520	67.642 $\pm$ 1.724
EMP-B H_0	67.870 $\pm$ 1.677	68.499 $\pm$ 1.962	67.514 $\pm$ 1.868	68.037 $\pm$ 2.035
EMP-B H_1	69.205 $\pm$ 1.239	69.157 $\pm$ 1.220	68.935 $\pm$ 0.818	69.223 $\pm$ 1.058
EMP-B H_0 + H_1	68.100 $\pm$ 1.171	68.492 $\pm$ 1.245	68.335 $\pm$ 1.445	69.077 $\pm$ 1.470
EMP-E H_0	65.190 $\pm$ 1.484	64.930 $\pm$ 2.045	64.962 $\pm$ 1.682	65.288 $\pm$ 1.754
EMP-E H_1	77.469 $\pm$ 1.144	77.438 $\pm$ 0.727	77.441 $\pm$ 0.830	**77.766** $\pm$ **0.952**
EMP-E H_0 + H_1	73.023 $\pm$ 0.818	72.963 $\pm$ 1.014	73.057 $\pm$ 1.284	73.324 $\pm$ 1.086
EMP-S H_0	68.135 $\pm$ 1.294	68.525 $\pm$ 0.865	68.426 $\pm$ 1.235	68.232 $\pm$ 0.969
EMP-S H_1	71.068 $\pm$ 1.417	71.071 $\pm$ 1.167	70.906 $\pm$ 1.416	70.743 $\pm$ 1.275
EMP-S H_0 + H_1	72.282 $\pm$ 1.012	72.246 $\pm$ 0.883	72.682 $\pm$ 0.913	72.904 $\pm$ 0.901

Table 4: Ablation Study on DHFR_MD dataset.

	none	$f_{\mathcal{V}}$	$f_{\mathcal{E}}$	$f_{\mathcal{V}} + f_{\mathcal{E}}$
none	-	68.333 ± 0.874	60.707 ± 1.455	68.058 ± 1.159
EMP-B H_0	71.872 ± 0.804	72.097 ± 0.876	71.794 ± 0.541	72.121 ± 0.943
EMP-B H_1	66.638 ± 0.690	68.190 ± 0.933	67.072 ± 0.705	68.774 ± 1.028
EMP-B H_0 + H_1	73.959 ± 0.746	74.087 ± 0.820	74.085 ± 0.671	74.213 ± 0.905
EMP-E H_0	74.906 ± 0.968	74.550 ± 0.867	74.521 ± 1.066	74.523 ± 1.027
EMP-E H_1	66.218 ± 1.499	66.424 ± 1.304	66.067 ± 1.152	66.397 ± 0.899
EMP-E H_0 + H_1	74.947 ± 0.663	74.849 ± 0.819	74.667 ± 0.922	74.796 ± 0.960
EMP-S H_0	78.434 ± 0.344	78.028 ± 0.597	78.537 ± 0.677	78.309 ± 0.681
EMP-S H_1	75.053 ± 0.909	74.953 ± 0.930	75.056 ± 0.785	75.259 ± 1.051
EMP-S H_0 + H_1	80.174 ± 1.081	$\mathbf{80.503 \pm 1.066}$	80.174 ± 0.864	80.126 ± 0.939

Table 5: Ablation Study on REDDIT-B dataset.

	none	$f_{\mathcal{V}}$	$f_{\mathcal{E}}$	$f_{\mathcal{V}} + f_{\mathcal{E}}$
none	-	89.680 ± 0.300	79.590 ± 0.394	90.125 ± 0.277
EMP-B H_0	88.925 ± 0.168	90.400 ± 0.145	89.145 ± 0.193	90.520 ± 0.204
EMP-B H_1	80.715 ± 0.235	87.090 ± 0.217	84.620 ± 0.160	88.225 ± 0.280
EMP-B H_0 + H_1	89.970 ± 0.309	90.945 ± 0.184	90.025 ± 0.234	$\mathbf{91.025 \pm 0.221}$
EMP-E H_0	88.140 ± 0.265	89.875 ± 0.185	88.400 ± 0.210	89.945 ± 0.177
EMP-E H_1	79.850 ± 0.365	87.670 ± 0.303	84.600 ± 0.214	88.640 ± 0.156
EMP-E H_0 + H_1	89.285 ± 0.234	90.010 ± 0.234	89.240 ± 0.202	90.045 ± 0.221
EMP-S H_0	86.150 ± 0.261	87.925 ± 0.162	86.940 ± 0.206	88.415 ± 0.241
EMP-S H_1	77.580 ± 0.186	85.435 ± 0.332	81.580 ± 0.222	86.960 ± 0.346
EMP-S H_0 + H_1	86.790 ± 0.385	88.250 ± 0.266	87.410 ± 0.315	88.590 ± 0.377

REFERENCES

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International Conference on Machine Learning*, 2019.

Sami Abu-El-Haija, Amol Kapoor, Bryan Perozzi, and Joonseok Lee. N-gcn: Multi-scale graph convolution for semi-supervised node classification. In *Uncertainty in Artificial Intelligence*, pp. 841–851, 2020.

Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, and Lori Ziegelmeier. Persistence images: A stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18, 2017.

Mehmet E Aktas, Esra Akbas, and Ahmed El Fatmaoui. Persistence homology of networks: methods and applications. *Applied Network Science*, 4(1):1–28, 2019.

Nieves Atienza, Rocío González-Díaz, and Manuel Soriano-Trigueros. On the stability of persistent entropy and new summary functions for topological data analysis. *Pattern Recognition*, 107: 107509, 2020.

Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21(suppl_1): i47–i56, 2005.

Magnus Bakke Botnan and Michael Lesnick. An introduction to multiparameter persistence. *arXiv preprint arXiv:2203.14289*, 2022.

Magnus Bakke Botnan, Steffen Oppermann, and Steve Oudot. Signed barcodes for multi-parameter persistence via rank decompositions and rank-exact resolutions. *arXiv preprint arXiv:2107.06800*, 2021.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.

P. Bubenik. Statistical topological data analysis using persistence landscapes. *JMLR*, 16(1):77–102, 2015.

Mathieu Carrière and Andrew Blumberg. Multiparameter persistence image for topological machine learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. Perslay: A neural network layer for persistence diagrams and new graph topological signatures. In *AISTATS*, pp. 2786–2796, 2020.

Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *arXiv preprint arXiv:1710.04019*, 2017.

Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in Artificial Intelligence*, 4, 2021.

Frédéric Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, and Larry Wasserman. Stochastic convergence of persistence landscapes and silhouettes. In *Proceedings of the thirtieth annual symposium on Computational geometry*, pp. 474–483, 2014.

Y. Chen, Y.R. Gel, and K. Avrachenkov. Lfgcn: Levitating over graphs with levy flights. In *IEEE ICDM*, 2020.

Yu-Min Chung and Austin Lawson. Persistence curves: A canonical framework for summarizing persistence diagrams. *arXiv preprint arXiv:1904.07768*, 2019.

M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pp. 3844–3852, 2016.

Tamal Krishna Dey and Yusu Wang. *Computational Topology for Data Analysis*. Cambridge University Press, 2022.

Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

Herbert Edelsbrunner and Salman Parsa. On the computational complexity of betti numbers: reductions from matrix rank. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on discrete algorithms*, pp. 152–160. SIAM, 2014.

David Eisenbud. *Commutative algebra: with a view toward algebraic geometry*, volume 150. Springer Science & Business Media, 2013.

Barbara Giunti. Tda applications library, 2022. https://www.zotero.org/groups/2425412/tda-applications/library.

Felix Hensel, Michael Moor, and Bastian Rieck. A survey of topological machine learning methods. *Frontiers in Artificial Intelligence*, 4:52, 2021.

Christoph Hofer, Florian Graf, Bastian Rieck, Marc Niethammer, and Roland Kwitt. Graph filtration learning. In *International Conference on Machine Learning*, pp. 4314–4323. PMLR, 2020.

Christoph D Hofer, Roland Kwitt, and Marc Niethammer. Learning representations of persistence barcodes. *Journal of Machine Learning Research*, 20(126):1–45, 2019.

Megan Johnson and Jae-Hun Jung. Instability of the betti sequence for persistent homology and a stabilized version of the betti sequence. *arXiv preprint arXiv:2109.09218*, 2021.

T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017a.

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

Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2019.

Risi Kondor and Horace Pan. The multiscale laplacian graph kernel. *Advances in neural information processing systems*, 29, 2016.

Nils Kriege and Petra Mutzel. Subgraph matching kernels for attributed graphs. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 291–298, 2012.

Nils M Kriege, Pierre-Louis Giscard, and Richard Wilson. On valid optimal assignment kernels and applications to graph classification. *Advances in neural information processing systems*, 29, 2016.

Nils M Kriege, Fredrik D Johansson, and Christopher Morris. A survey on graph kernels. *Applied Network Science*, 5(1):1–42, 2020.

Panagiotis Kyriakis, Iordanis Fostiropoulos, and Paul Bogdan. Learning hyperbolic representations of topological features. In *International Conference on Learning Representations*, 2021.

P. Langley. Crafting papers on machine learning. In Pat Langley (ed.), *International Conference on Machine Learning*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Tam Le and Makoto Yamada. Persistence fisher kernel: A riemannian manifold kernel for persistence diagrams. In *Advances in Neural Information Processing Systems*, pp. 10007–10018, 2018.

Michael Lesnick and Matthew Wright. Computing minimal presentations and bigraded betti numbers of 2-parameter persistent homology. *arXiv preprint arXiv:1902.05708*, 2019.

Michael Lesnick and Matthew Wright. Computing minimal presentations and bigraded betti numbers of 2-parameter persistent homology. *SIAM Journal on Applied Algebra and Geometry*, 6(2): 267–298, 2022.

Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 338–348, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403076. URL https://doi.org/10.1145/3394486.3403076.

Federico Monti, Karl Otness, and Michael M Bronstein. Motifnet: a motif-based graph convolutional network for directed graphs. In *IEEE DSW*, pp. 225–228, 2018.

Christopher Morris, Nils M Kriege, Kristian Kersting, and Petra Mutzel. Faster kernels for graphs with continuous attributes via hashing. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1095–1100. IEEE, 2016.

Leslie O'Bray, Bastian Rieck, and Karsten Borgwardt. Filtration curves for graph representation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1267–1275, 2021.

Nina Otter, Mason A Porter, Ulrike Tillmann, Peter Grindrod, and Heather A Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6:1–38, 2017.

Meng Qu, Yoshua Bengio, and Jian Tang. Gmnn: Graph markov neural networks. *arXiv:1905.06214*, 2019.

Bastian Rieck, Christian Bock, and Karsten Borgwardt. A persistent weisfeiler-lehman procedure for graph classification. In *International Conference on Machine Learning*, pp. 5448–5458, 2019.

Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(9), 2011.

Jeffrey J Sutherland, Lee A O'brien, and Donald F Weaver. Spline-fitting with a genetic algorithm: A method for developing classification structure- activity relationships. *Journal of chemical information and computer sciences*, 43(6):1906–1915, 2003.

Matteo Togninalli, Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten Borgwardt. Wasserstein weisfeiler-lehman graph kernels. In *Advances in Neural Information Processing Systems*, pp. 6439–6449, 2019.

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

Oliver Vipond. Multiparameter persistence landscapes. *J. Mach. Learn. Res.*, 21:61–1, 2020.

Larry Wasserman. Topological data analysis. *Annual Review of Statistics and Its Application*, 5: 501–532, 2018.

Yiding Yang, Xinchao Wang, Mingli Song, Junsong Yuan, and Dacheng Tao. Spagan: shortest path graph attention network. In *International Joint Conference on Artificial Intelligence*, pp. 4099–4105, 2019.

Qi Zhao and Yusu Wang. Learning metrics for persistence-based summaries and applications for graph classification. In *NeurIPS*, pp. 9855–9866, 2019.