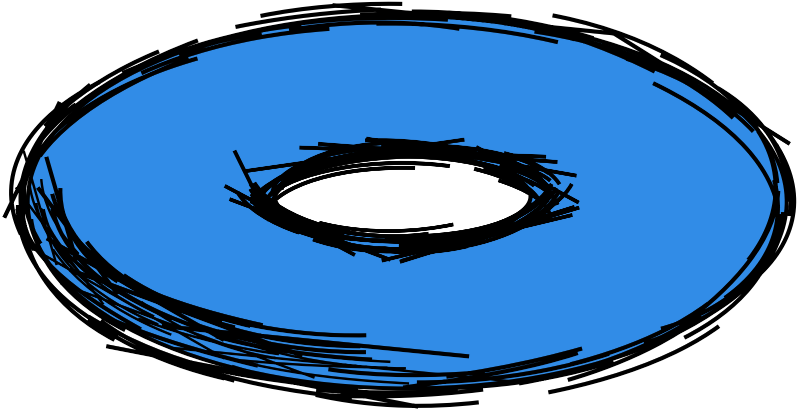
Topology-Based Representation Learning

Bastian Rieck

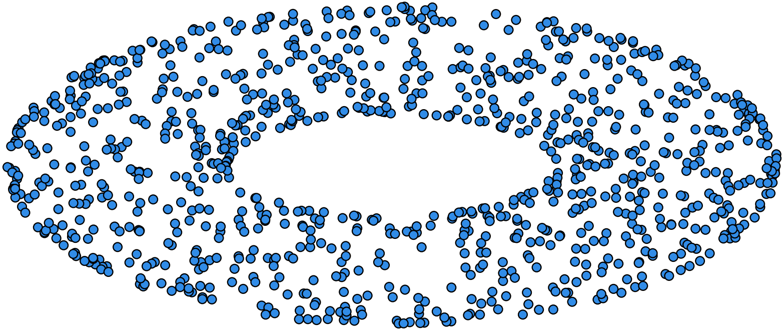
Pseudomanifold

MLCB
D BSSE
ETH zürich

Topological data analysis

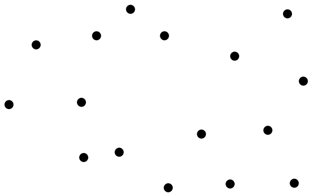


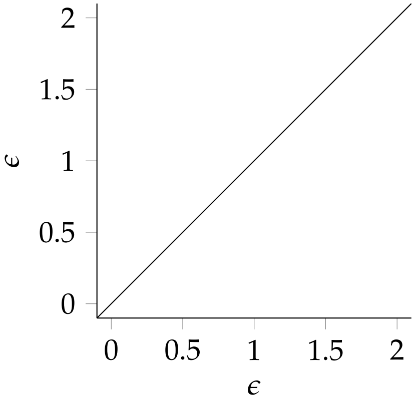
Topological data analysis



Persistent homology

Vietoris–Rips complex calculation

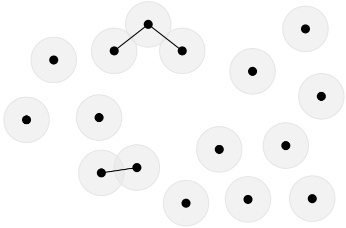


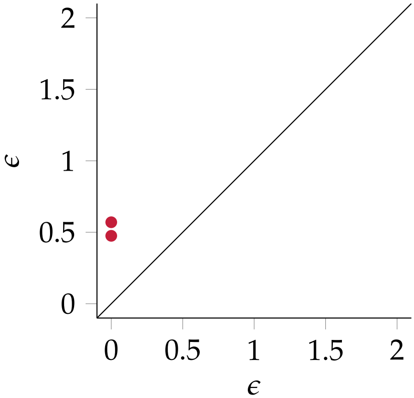


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Persistent homology

Vietoris–Rips complex calculation

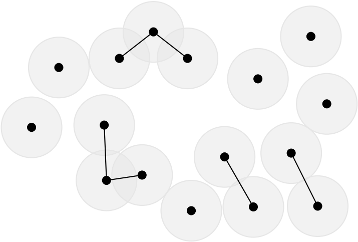


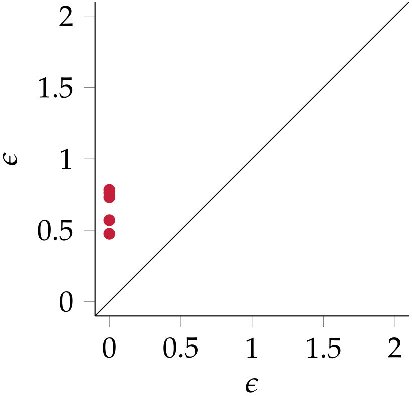


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Persistent homology

Vietoris–Rips complex calculation

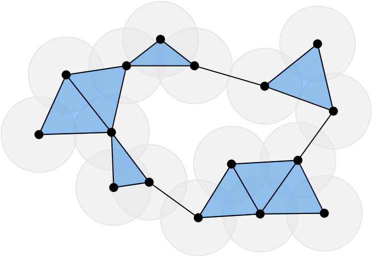


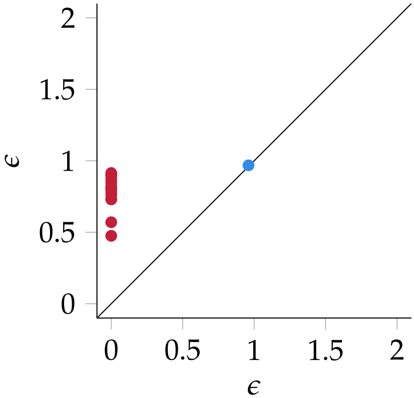


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Persistent homology

Vietoris–Rips complex calculation

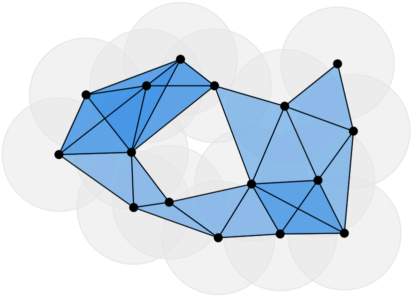


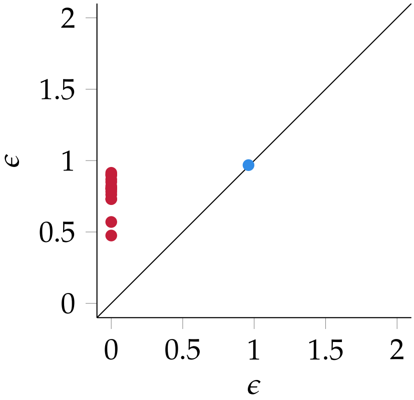


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Persistent homology

Vietoris–Rips complex calculation

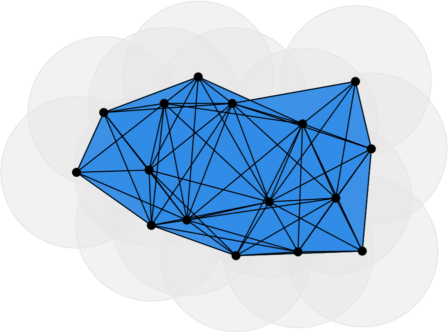


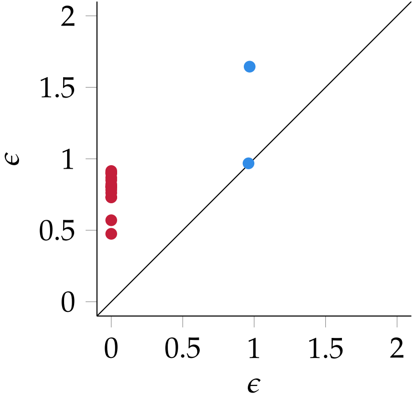


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Persistent homology

Vietoris–Rips complex calculation



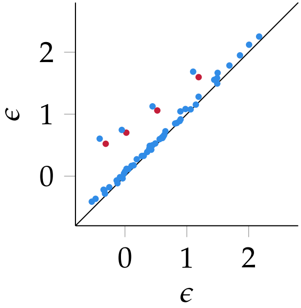


Given $\epsilon \in \mathbb{R}$, the Vietoris–Rips complex contains all simplices whose pairwise distance is less than or equal to ϵ. When using Euclidean balls of radius $r = 0.5\epsilon$, a simplex is created for each pairwise intersection.

Distances between persistence diagrams

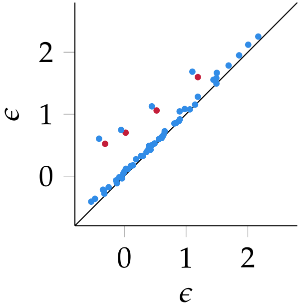
$$W_\infty(\mathcal{D}_1, \mathcal{D}_2) := \inf_{\eta\colon \mathcal{D}_1 \to \mathcal{D}_2} \sup_{x \in \mathcal{D}_1} \|x - \eta(x)\|_\infty$$

Bottleneck distance



$$W_p(\mathcal{D}_1, \mathcal{D}_2) := \left(\inf_{\eta\colon \mathcal{D}_1 \to \mathcal{D}_2} \sum_{x \in \mathcal{D}_1} \|x - \eta(x)\|_\infty^p \right)^{\frac{1}{p}}$$

Wasserstein distance

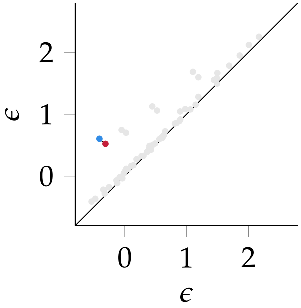


Distances between persistence diagrams

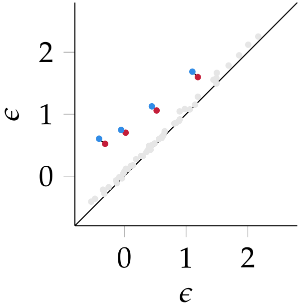
$$W_\infty(\mathcal{D}_1, \mathcal{D}_2) := \inf_{\eta\colon \mathcal{D}_1 \to \mathcal{D}_2} \sup_{x \in \mathcal{D}_1} \|x - \eta(x)\|_\infty$$

$$W_p(\mathcal{D}_1, \mathcal{D}_2) := \left(\inf_{\eta\colon \mathcal{D}_1 \to \mathcal{D}_2} \sum_{x \in \mathcal{D}_1} \|x - \eta(x)\|_\infty^p \right)^{\frac{1}{p}}$$

Bottleneck distance



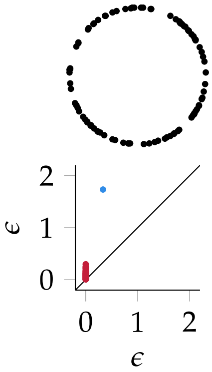
Wasserstein distance



Stability theorem

Robustness to *small-scale* perturbations

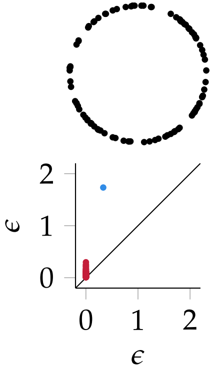
Let $\mathcal{M}$ be a triangulable space with continuous tame functions $f, g\colon \mathcal{M} \to \mathbb{R}$. Then the corresponding persistence diagrams satisfy $\mathrm{W}_\infty(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$.

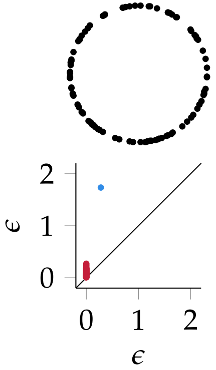


Stability theorem

Robustness to *small-scale* perturbations

Let $\mathcal{M}$ be a triangulable space with continuous tame functions $f, g\colon \mathcal{M} \to \mathbb{R}$. Then the corresponding persistence diagrams satisfy $\mathrm{W}_\infty(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$.

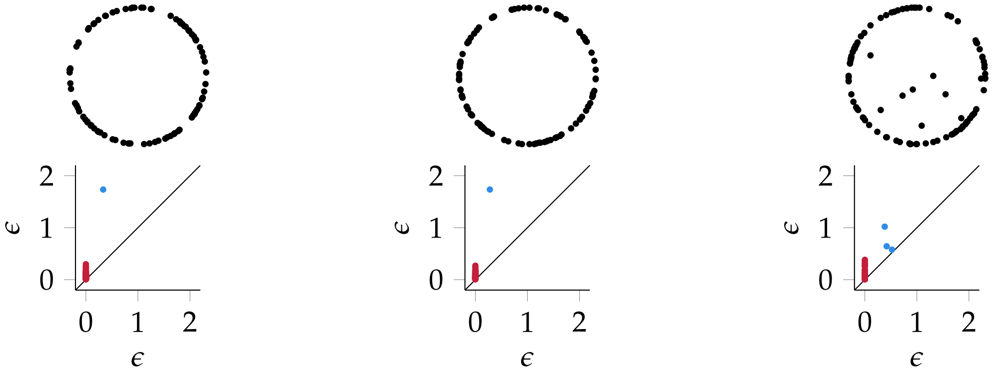




Stability theorem

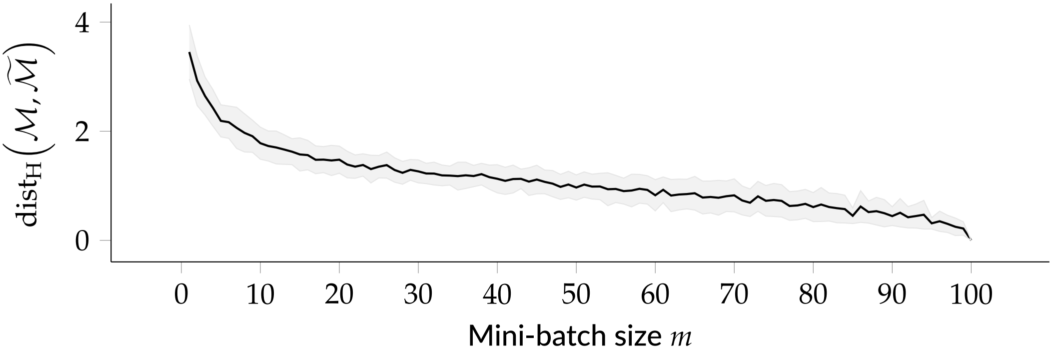
Robustness to *small-scale* perturbations

Let $\mathcal{M}$ be a triangulable space with continuous tame functions $f, g\colon \mathcal{M} \to \mathbb{R}$. Then the corresponding persistence diagrams satisfy $W_\infty(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$.



Implications for machine learning

Need to be careful when working with mini-batches $\widetilde{\mathcal{M}}$ of a point cloud $\mathcal{M}$. As an example, consider a point cloud with 100 points (normally-distributed in $\mathbb{R}^2$) and 50 subsamples of varying size m.



Bridging the chasm

- Persistent homology is inherently *discrete*
- Deep learning is inherently *continuous*

Challenge

Can we make the calculation of a persistence diagram *differentiable*, in particular if we have some control over the input space $\mathcal{M}$?

First approach

Continuation of Point Clouds via Persistence Diagrams **(M. Gameiro et al.)**

arXiv:1506.03147v1 [math.NA] 10 Jun 2015

Continuation of Point Clouds via Persistence Diagrams

Marcio Gameiro[a], Yasuaki Hiraoka[b], Ippei Obayashi[b]

[a] *Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, Caixa Postal 668, 13560-970, São Carlos, SP, Brazil*

[b] *WPI-Advanced Institute for Materials Research (WPI-AIMR), Tohoku University, 2-1-1 Katahira, Aoba-ku, Sendai, 980-8577 Japan*

Abstract

In this paper, we present a mathematical and algorithmic framework for the continuation of point clouds by persistence diagrams. A key property used in the method is that the persistence map, which assigns a persistence diagram to a point cloud, is differentiable. This allows us to apply the Newton-Raphson continuation method in this setting. Given an original point cloud P, its persistence diagram D, and a target persistence diagram D', we gradually move from D to D', by successively computing intermediate point clouds until we finally find a point cloud P' having D' as its persistence diagram. Our method can be applied to a wide variety of situations in topological data analysis where it is necessary to solve an inverse problem, from persistence diagrams to point cloud data.

Keywords: Point Cloud, Persistent Homology, Persistence Diagram, Continuation

1. Introduction

Let P be a finite set of points in $\mathbb{R}^L$ given by

$$P = \{u_i \in \mathbb{R}^L \mid i = 1 \ldots, M\}. \quad (1.1)$$

We call P a point cloud, following the convention in topological data analysis (TDA) [1, 2]. TDA provides us tools to study the "shape" of P. Among them, persistent homology [3, 4] is one of the most useful tools, and it is now applied into various practical applications, e.g., amorphous solids [5, 6], proteins [7], and sensor networks [8] (see also [1] and references therein).

Persistent homology can be regarded as a collection of maps, called *persistence maps* in this paper, from P to a finite set D_ℓ, for $\ell = 0, 1, \cdots$, in the extended plane $\bar{\mathbb{R}}^2 = \bar{\mathbb{R}} \times \bar{\mathbb{R}}$, where $\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$. The set D_ℓ is called *persistence diagram* and it encodes the ℓ-dimensional topological features of P with metric information (precise definitions are given in Section 2.3).

In many applications, the point cloud P have an intricate "shape" or structure, and, in this situations, persistence diagrams are used to provide the "essential" topological features of P. For example, in the papers [5, 6], the authors study hierarchical geometric structures in several amorphous solids. In such a case, P is given by an atomic configuration of an amorphous solid and consists of thousands of points in $\mathbb{R}^3$ obtained by molecular dynamics simulations. It is a difficult task to directly study the geometry and physical properties of the amorphous solid from P due to its immense size. Hence, a key observation of their work is that the persistence diagrams of the atomic configurations can capture essential geometric information of the amorphous solids. From this significant property, using persistence diagrams they obtain various physical properties of the solid, such as, finding new hierarchical ring structures, decompositions of first sharp diffraction peaks, mechanical responses, etc.

Figure 1 shows a schematic representation of D_1 for silica glass, P, studied in [5] (this corresponds to Figure 1 in that paper). They show that the presence of curves in D_1 precisely distinguishes the amorphous state from liquid and crystalline states. It means that the normal direc-

Email addresses: gameiro@icmc.usp.br (Marcio Gameiro), hiraoka@wpi-aimr.tohoku.ac.jp (Yasuaki Hiraoka), ippei.obayashi.d8@tohoku.ac.jp (Ippei Obayashi)

Preprint submitted to Elsevier *June 11, 2015*

1. Represent persistent homology calculation as a single map of the form

$$\mathbb{R}^n \ni x \mapsto y \in \mathbb{R}^m,$$

where x is a point cloud and y is a vectorised sequence of persistence diagrams.

2. Show that this map decomposes into

$$x \xmapsto{g} r \xmapsto{h} y,$$

where g calculates a filtration, and h calculates its persistence diagrams.

3. Show that g and h are differentiable, thus implying that $f := h \circ g$ is differentiable.

Second approach

***Topological Function Optimization for Continuous Shape Matching* (A. Poulenard et al.)**

Eurographics Symposium on Geometry Processing 2018
T. Ju and A. Vaxman
(Guest Editors)

Volume 37 (2018), Number 5

Topological Function Optimization for Continuous Shape Matching

Adrien Poulenard[1], Primoz Skraba[2], Maks Ovsjanikov[1]

[1]LIX, Ecole Polytechnique, CNRS, [2]Jozef Stefan Institute

Abstract

We present a novel approach for optimizing real-valued functions based on a wide range of topological criteria. In particular, we show how to modify a given function in order to remove topological noise and to exhibit prescribed topological features. Our method is based on using the previously-proposed persistence diagrams *associated with real-valued functions, and on the analysis of the derivatives of these diagrams with respect to changes in the function values. This analysis allows us to use continuous optimization techniques to modify a given function, while optimizing an energy based purely on the values in the persistence diagrams. We also present a procedure for aligning persistence diagrams of functions on different domains, without requiring a mapping between them. Finally, we demonstrate the utility of these constructions in the context of the functional map framework, by first giving a characterization of functional maps that are associated with* continuous *point-to-point correspondences, directly in the functional domain, and then by presenting an optimization scheme that helps to promote the continuity of functional maps, when expressed in the reduced basis, without imposing any restrictions on metric distortion. We demonstrate that our approach is efficient and can lead to improvement in the accuracy of maps computed in practice.*

1. Introduction

A core problem in geometry processing consists in quantifying similarity between shapes and their parts, as well as detecting detailed region or point-based correspondences [VKZHCO11, TCL*13, BCBB16]. A common approach for both shape comparison and correspondence consists in computing real-valued (for example, descriptor) functions defined on the shapes and comparing the shapes and their parts by comparing the values of such functions. This includes both computing correspondences by matching in descriptor space, and also, more recently, by computing linear transformations between spaces of real-valued functions using the so-called functional map framework [OBCS*12, OCB*17].

Many existing techniques for comparison of functions on the shapes directly rely on comparing function values, without analyzing the global structure of the functions involved. For example, a descriptor function computed on one shape can have several prominent maxima, whereas on another shape, it can be uniform or with low variance. Intuitively, pairs of functions with dissimilar structural properties can lead to large errors in the correspondence computation. This problem is especially prominent in the context of *functional maps* which are linear transformations between spaces of real-valued functions defined on different shapes. In this case, one is often interested in formulating an objective which would promote mapping indicator functions of connected regions to other such indicator functions without knowing in advance which regions should match. At a high level, such an objective should promote the preservation of *the topological structure* of the functions before and after the mapping.

In this paper, we show how such problems can be solved by efficiently optimizing the topological structure of real-valued functions defined on the shapes, either independently (to promote certain structural properties), or jointly (to enforce similarity between such properties), without resorting to combinatorial search or point-to-point maps. The key to our approach is the manipulation of *persistence diagrams* [CZCG05, Car09]. These diagrams have been shown to summarize the properties of very general classes of topological spaces, including, most relevant to us, real-valued functions defined on the surfaces, and also enjoy several key properties such as being stable under a broad range of perturbations [CSEH07, CCSG*09]. Existing methods, however, concentrate on either *efficiently constructing* persistence diagrams from a given signal [CZCG05, MMS11, CK13] or using them as a tool for, e.g. shape or image comparison [CZCG05, LOC14] or shape segmentation [SOCG10] among many others.

Our main insight is that it is possible to formulate optimization objectives on the persistence diagrams of real-valued functions, regardless of their underlying spatial domain, and to optimize a given function to improve such objectives, via continuous non-linear optimization. For this, we first show, how the derivative of a persistence diagram of a function can be computed with respect to the change in the function values, and then how this computation can be used to efficiently optimize various energies defined on persis-

- Introduced in the context of analysing a scalar-valued function over a point cloud.
- Applications for shape matching or function simplification.
- Simpler proof of local differentiability!

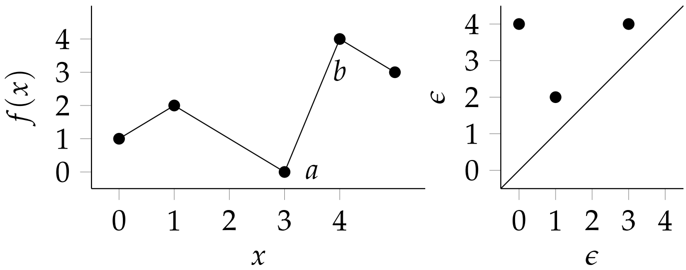
Sketch of the proof

Terminology

- Let $f\colon \mathcal{M} \to \mathbb{R}$ be a function on a point cloud. Persistent homology can be seen as a map from $(\mathcal{M}, f)$ to $\{(c_i, d_i)\}_{i \in \mathcal{I}}$.
- Let $\mathcal{S}$ be a map from points in the persistence diagram to pairs of simplices, i.e. $\mathcal{S}(c_i, d_i) = (\sigma_i, \tau_i)$. We write $\mathcal{S}(\cdot)$ to denote the map for a single point.
- Depending on the filtration, we can also map a simplex to one of its vertices. For the sublevel set filtration, for example, we have a map $\mathcal{V}$ with $\mathcal{V}(\sigma) := \arg\max_{v \in \sigma} f(v)$.
- Finally, let $\mathcal{P} := (\mathcal{P}_c, \mathcal{P}_d)$, with $\mathcal{P}_c := \mathcal{V} \circ \mathcal{S}(c_i)$ and $\mathcal{P}_d := \mathcal{V} \circ \mathcal{S}(d_i)$.

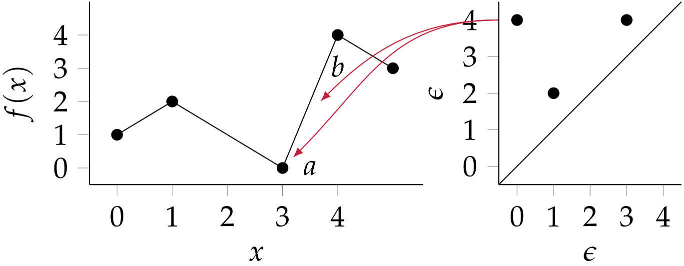
Sketch of the proof

Example



Sketch of the proof

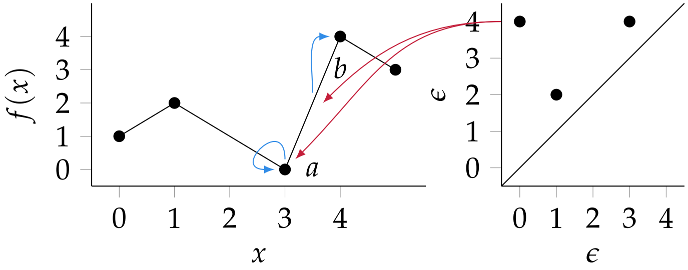
Example



We have $\mathcal{S}(0,4) = (\{a\},\{a,b\})$.

Sketch of the proof

Example

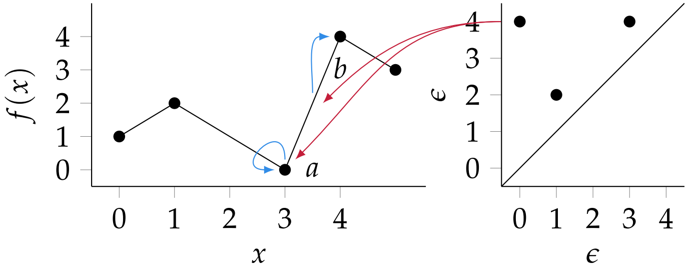


We have $\mathcal{S}(0,4) = (\{a\},\{a,b\})$.

We have $\mathcal{V}(\{a\}) = x_3$ and $\mathcal{V}(\{a,b\}) = x_4$.

Sketch of the proof

Example



We have $\mathcal{S}(0,4) = (\{a\}, \{a,b\})$.

We have $\mathcal{V}(\{a\}) = x_3$ and $\mathcal{V}(\{a,b\}) = x_4$.

We have $\mathcal{P}(0,4) = (\mathcal{V} \circ \mathcal{S})(0,4) = (x_3, x_4)$.

Sketch of the proof, continued

- If the function values are *distinct*, then $\mathcal{P}$ is *unique*.
- If the function values are *distinct*, then $\mathcal{P}$ is *constant* in some neighbourhood.

Assume that f depends on $\theta = (\theta_1, \theta_2, \dots)$. We then have $f(\mathcal{P}_c(c_i)) = c_i$, and, since $\mathcal{P}$ is constant,

$$\frac{\partial c_i}{\partial \theta_j} = \frac{\partial f(\mathcal{P}_c(c_i))}{\partial \theta_j} = \frac{\partial f(v_i)}{\partial \theta_j} = \frac{\partial f}{\partial \theta_j}(v_i),$$

i.e. the partial derivative is equivalent to the derivative of the function evaluated at the image of the map $\mathcal{P}_c$.

It is a little bit more complicated when using *distances* instead of scalar-valued filtrations, but the principle remains the same.

Topological autoencoders

Topological Autoencoders

Michael Moor [†,1,2] **Max Horn** [†,1,2] **Bastian Rieck** [‡,1,2] **Karsten Borgwardt** [‡,1,2]

Abstract

We propose a novel approach for preserving topological structures of the input space in latent representations of autoencoders. Using *persistent homology*, a technique from topological data analysis, we calculate topological signatures of both the input and latent space to derive a topological loss term. Under weak theoretical assumptions, we construct this loss in a differentiable manner, such that the encoding learns to retain multi-scale connectivity information. We show that our approach is theoretically well-founded and that it exhibits favourable latent representations on a synthetic manifold as well as on real-world image data sets, while preserving low reconstruction errors.

1. Introduction

While topological features, in particular multi-scale features derived from persistent homology, have seen increasing use in the machine learning community (Carrière et al., 2019, Guss & Salakhutdinov, 2018, Hofer et al., 2017, 2019a,b, Ramamurthy et al., 2019, Reininghaus et al., 2015, Rieck et al., 2019a,b), employing topology *directly* as a constraint for modern deep learning methods remains a challenge. This is due to the inherently discrete nature of these computations, making backpropagation through the computation of topological signatures immensely difficult or only possible in certain special circumstances (Chen et al., 2019, Hofer et al., 2019a, Poulenard et al., 2018).

This work presents a novel approach that permits obtaining gradients during the computation of topological signatures. This makes it possible to employ topological constraints while training deep neural networks, as well as building topology-preserving autoencoders. Specifically, we make the following contributions:

1. We develop a new topological loss term for autoencoders that helps harmonise the topology of the data space with the topology of the latent space.
2. We prove that our approach is stable on the level of mini-batches, resulting in suitable approximations of the persistent homology of a data set.
3. We empirically demonstrate that our loss term aids in dimensionality reduction by preserving topological structures in data sets; in particular, the learned latent representations are useful in that the preservation of topological structures can improve interpretability.

2. Background: Persistent Homology

Persistent homology (Barannikov, 1994, Edelsbrunner & Harer, 2008) is a method from the field of computational topology, which develops tools for analysing topological features (connectivity-based features such as connected components) of data sets. We first introduce the underlying concept of simplicial homology. For a simplicial complex $\mathfrak{K}$, i.e. a generalised graph with higher-order connectivity information such as cliques, simplicial homology employs matrix reduction algorithms to assign $\mathfrak{K}$ a family of groups, the *homology groups*. The d^{th} homology group $\mathrm{H}_d(\mathfrak{K})$ of $\mathfrak{K}$ contains d-dimensional topological features, such as connected components ($d = 0$), cycles/tunnels ($d = 1$), and voids ($d = 2$). Homology groups are typically summarised by their ranks, thereby obtaining a simple invariant "signature" of a manifold. For example, a circle in $\mathbb{R}^2$ has one feature with $d = 1$ (a cycle), and one feature with $d = 0$ (a connected component).

In practice, the underlying manifold $\mathbb{M}$ is unknown and we are working with a point cloud $X := \{x_1, \ldots, x_n\} \subseteq \mathbb{R}^d$ and a metric $\mathrm{dist}\colon X \times X \to \mathbb{R}$ such as the Euclidean distance. Persistent homology extends simplicial homology to this setting: instead of approximating $\mathbb{M}$ by means of a *single* simplicial complex, which would be an unstable procedure due to the discrete nature of X, persistent homology tracks changes in the homology groups over *multiple* scales of the metric. This is achieved by constructing a special simplicial complex, the Vietoris–Rips complex (Vietoris, 1927). For $0 \leq \epsilon < \infty$, the Vietoris–Rips complex of X at scale ϵ, denoted by $\mathfrak{R}_\epsilon(X)$, contains all

[†] Equal contribution. [‡] These authors jointly directed this work. [1] Department of Biosystems Science and Engineering, ETH Zurich, 4058 Basel, Switzerland [2] SIB Swiss Institute of Bioinformatics, Switzerland. Correspondence to: Karsten Borgwardt <karsten.borgwardt@bsse.ethz.ch>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

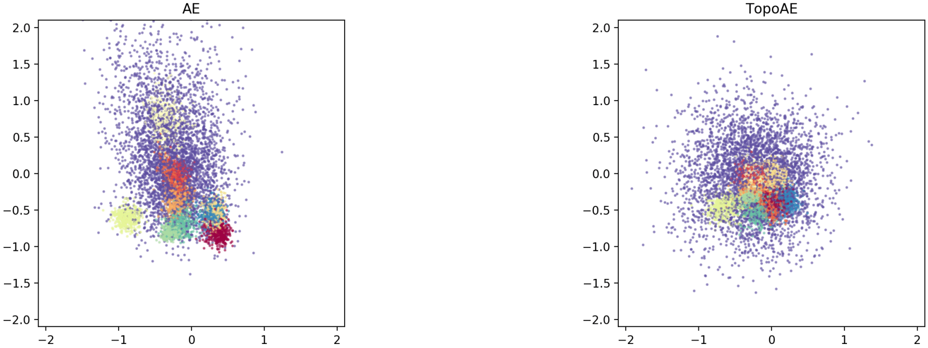
Michael Moor
Michael_D_Moor

Max Horn
ExpectationMax

Karsten Borgwardt
kmborgwardt

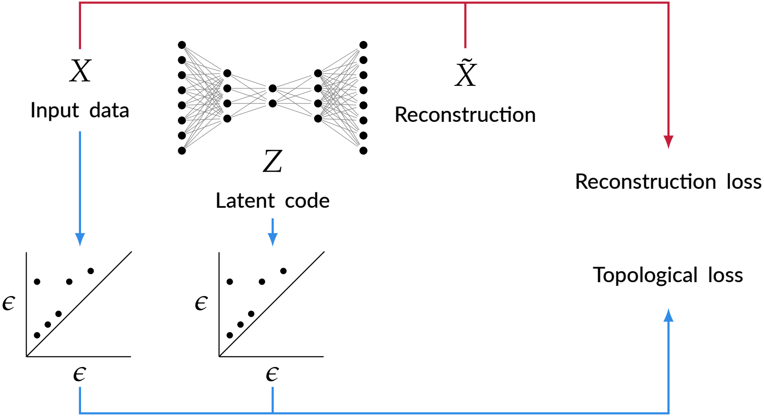
Topological autoencoders

Motivation



Topological autoencoders

Overview



Topological autoencoders

Main intuition

Align persistence diagrams of an *input batch* and of a *latent batch* using a loss function!

Why this works in theory

Let X be a point cloud of cardinality n and $X^{(m)}$ be one subsample of X of cardinality m, i.e. $X^{(m)} \subseteq X$, sampled without replacement. We can bound the probability of the persistence diagrams of $X^{(m)}$ exceeding a threshold in terms of the bottleneck distance as

$$\mathbb{P}\left(\mathrm{W}_\infty\left(\mathcal{D}^X, \mathcal{D}^{X^{(m)}}\right) > \epsilon\right) \leq \mathbb{P}\left(\mathrm{dist}_\mathrm{H}\left(X, X^{(m)}\right) > 2\epsilon\right),$$

where dist_H denotes the Hausdorff distance. In other words: *mini-batches are topologically similar if the subsampling is not too coarse*.

Topological autoencoders

Gradient calculation intuition

Distance matrix $\mathbf{A}$

$$\begin{bmatrix} 0 & 1 & 9 & 10 \\ 1 & 0 & 7 & 8 \\ 9 & 7 & 0 & 3 \\ 10 & 8 & 3 & 0 \end{bmatrix}$$

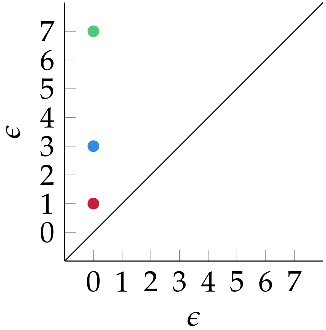
Every point in the persistence diagram can be mapped to *one* entry in the distance matrix! Each entry *is* a distance, so it can be changed during training (at least in the latent space).

Topological autoencoders

Gradient calculation intuition

Distance matrix $\mathbf{A}$

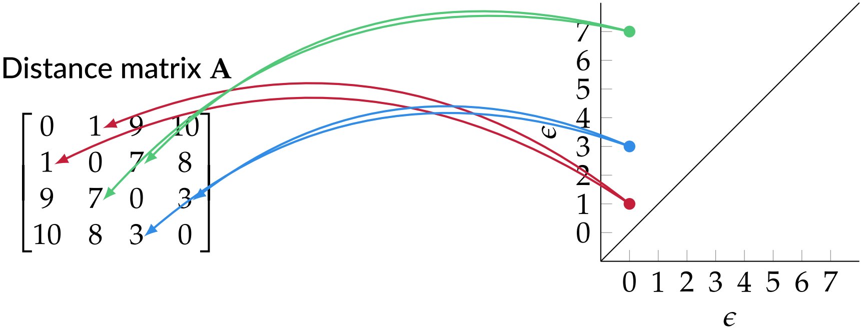
$$\begin{bmatrix} 0 & 1 & 9 & 10 \\ 1 & 0 & 7 & 8 \\ 9 & 7 & 0 & 3 \\ 10 & 8 & 3 & 0 \end{bmatrix}$$



Every point in the persistence diagram can be mapped to *one* entry in the distance matrix! Each entry *is* a distance, so it can be changed during training (at least in the latent space).

Topological autoencoders

Gradient calculation intuition



Every point in the persistence diagram can be mapped to *one* entry in the distance matrix! Each entry *is* a distance, so it can be changed during training (at least in the latent space).

Topological autoencoders

Loss term

$$\mathcal{L}_t := \mathcal{L}_{\mathcal{X}\to\mathcal{Z}} + \mathcal{L}_{\mathcal{Z}\to\mathcal{X}}$$

$$\mathcal{L}_{\mathcal{X}\to\mathcal{Z}} := \frac{1}{2}\left\|\mathbf{A}^X\left[\pi^X\right] - \mathbf{A}^Z\left[\pi^X\right]\right\|^2 \qquad \mathcal{L}_{\mathcal{Z}\to\mathcal{X}} := \frac{1}{2}\left\|\mathbf{A}^Z\left[\pi^Z\right] - \mathbf{A}^X\left[\pi^Z\right]\right\|^2$$

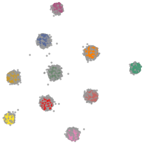
- $\mathcal{X}$: input space
- $\mathcal{Z}$: latent space
- $\mathbf{A}^X$: distances in input mini-batch
- $\mathbf{A}^Z$: distances in latent mini-batch
- π^X: persistence pairing of input mini-batch
- π^Z: persistence pairing of latent mini-batch

The loss is *bi-directional*!

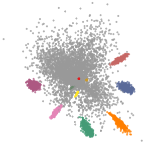
Qualitative evaluation

‘Spheres’ data set

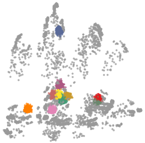
PCA



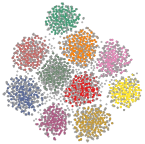
UMAP



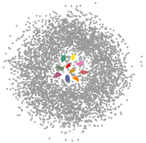
Autoencoder



Isomap



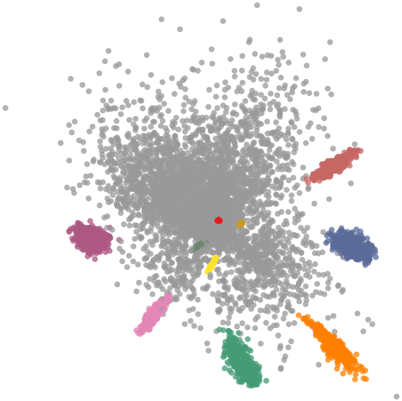
t-SNE



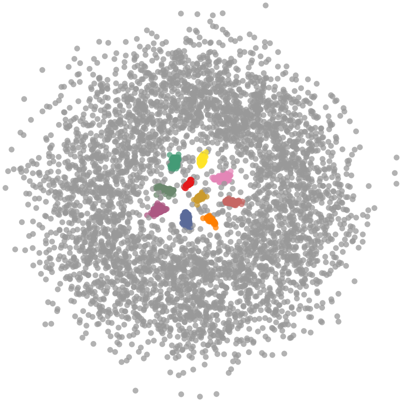
Topological autoencoder

Qualitative evaluation

‘Spheres’ data set; zooming in...



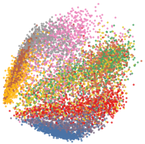
Autoencoder



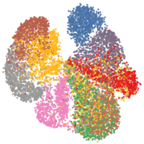
Topological autoencoder

Qualitative evaluation

‘FashionMNIST’ data set

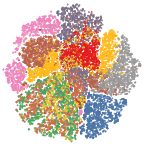


PCA

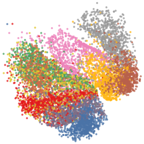


UMAP

Autoencoder



t-SNE



Topological autoencoder

A new evaluation metric

Use *distance to a measure* density estimator, i.e.

$$f_\sigma^{\mathcal{X}}(x) := \sum_{y \in \mathcal{X}} \exp\left(-\sigma^{-1} \operatorname{dist}(x, y)^2\right),$$

where dist denotes a metric such as the Euclidean distance. This is well-defined on mini-batches and on the full input data set.

Given σ, we evaluate $\mathrm{KL}_\sigma := \mathrm{KL}\left(f_\sigma^X \parallel f_\sigma^Z\right)$, which measures the similarity between the two density distributions.

Quantitative evaluation

Data set	Method	$\mathrm{KL}_{0.01}$	$\mathrm{KL}_{0.1}$	KL_{1}	ℓ-MRRE	ℓ-Cont	ℓ-Trust	ℓ-RMSE	MSE (data)
'Spheres'	Isomap	0.181	**0.420**	**0.00881**	**0.246**	**0.790**	**0.676**	10.4	
	PCA	0.332	0.651	0.01530	0.294	0.747	0.626	11.8	0.9610
	t-SNE	**0.152**	0.527	0.01271	**0.217**	0.773	**0.679**	**8.1**	
	UMAP	0.157	0.613	0.01658	0.250	0.752	0.635	**9.3**	
	AE	0.566	0.746	0.01664	0.349	0.607	0.588	13.3	**0.8155**
	TopoAE	**0.085**	**0.326**	**0.00694**	0.272	**0.822**	0.658	13.5	**0.8681**
'Fashion-MNIST'	PCA	**0.356**	**0.052**	**0.00069**	0.057	0.968	0.917	**9.1**	0.1844
	t-SNE	0.405	0.071	0.00198	**0.020**	0.967	**0.974**	41.3	
	UMAP	0.424	0.065	0.00163	0.029	**0.981**	0.959	**13.7**	
	AE	0.478	0.068	0.00125	**0.026**	0.968	**0.974**	20.7	**0.1020**
	TopoAE	**0.392**	**0.054**	**0.00100**	0.032	**0.980**	0.956	20.5	**0.1207**
'MNIST'	PCA	0.389	0.163	0.00160	0.166	0.901	0.745	**13.2**	0.2227
	t-SNE	**0.277**	**0.133**	0.00214	**0.040**	0.921	**0.946**	22.9	
	UMAP	**0.321**	0.146	0.00234	**0.051**	**0.940**	**0.938**	**14.6**	
	AE	0.620	0.155	**0.00156**	0.058	0.913	0.937	18.2	**0.1373**
	TopoAE	0.341	**0.110**	**0.00114**	0.056	**0.932**	0.928	19.6	**0.1388**

Topological autoencoders

Summary

- A simple way to preserve topological information of the input space for dimensionality reduction tasks
- Our loss term is differentiable under mild theoretical assumptions
- We only need *distances* to train (simple extension to other structured data sets?)

Learning graph filtrations

Graph Filtration Learning

Christoph D. Hofer[1] **Florian Graf**[1] **Bastian Rieck**[2] **Marc Niethammer**[3] **Roland Kwitt**[1]

Abstract

We propose an approach to learning with graph-structured data in the problem domain of graph classification. In particular, we present a novel type of *readout* operation to aggregate node features into a graph-level representation. To this end, we leverage persistent homology computed via a real-valued, learnable, filter function. We establish the theoretical foundation for differentiating through the persistent homology computation. Empirically, we show that this type of readout operation compares favorably to previous techniques, especially when the graph connectivity structure is informative for the learning problem.

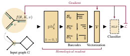


Figure 1. Overview of the proposed *homological readout*. Given a graph/simplicial complex, we use a vertex-based graph functional f to assign a real-valued score to each node. A practical choice is to implement f as a GNN, with one level of message passing. We then compute persistence barcodes, $\mathcal{B}_k$, using the filtration induced by f. Finally, barcodes are fed through a vectorization scheme $\mathcal{V}$ and passed to a classifier (e.g., an MLP). Our approach allows passing a learning signal *through* the persistent homology computation, allowing to optimize f for the classification task.

1. Introduction

We consider the task of learning a function from the space of (finite) undirected graphs, $\mathbb{G}$, to a (discrete/continuous) target domain $\mathbb{Y}$. Additionally, graphs might have discrete, or continuous attributes attached to each node. Prominent examples for this class of *learning* problem appear in the context of classifying molecule structures, chemical compounds or social networks.

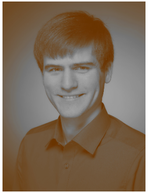
A substantial amount of research has been devoted to developing techniques for supervised learning with graph-structured data, ranging from kernel-based methods (Shervashidze et al., 2009; 2011; Feragen et al., 2013; Kriege et al., 2016), to more recent approaches based on graph neural networks (GNN) (Scarselli et al., 2009; Hamilton et al., 2017; Zhang et al., 2018b; Morris et al., 2019; Xu et al., 2019; Ying et al., 2018). Most of the latter works use an iterative message passing scheme (Gilmer et al., 2017) to learn node representations, followed by a graph-level pooling operation that aggregates node-level features. This aggregation step is typically referred to as a *readout* operation. While research has mostly focused on variants of the message passing function, the readout step may have a significant impact, as it aims to capture properties of the entire graph. Importantly, both simple and more refined readout operations, such as summation, differentiable pooling (Ying et al., 2018), or sort pooling (Zhang et al., 2018a), are inherently coupled to the amount of information carried over via multiple rounds of message passing. Hence, architectural GNN choices are typically guided by dataset characteristics, e.g., requiring to tune the number of message passing rounds to the expected size of graphs.

Contribution. We propose a *homological* readout operation that captures the full global structure of a graph, while relying only on node representations learned from immediate neighbors. This not only alleviates the aforementioned design challenge, but potentially offers additional discriminative information. Similar to previous works, we consider a graph, G, as a simplicial complex, K, and use persistent homology (Edelsbrunner & Harer, 2010) to capture homological changes that occur when constructing the graph one part at a time (i.e., revealing changes in the number of connected components or loops). As this hinges on an ordering of the parts, prior works rely on a suitable *filter function*

[1]Department of Computer Science, Univ. of Salzburg, Austria [2]Department of Biosystems Science and Engineering, ETH Zurich, Switzerland [3]Univ. of North Carolina, Chapel Hill, USA. Correspondence to: Christoph D. Hofer <chr.dav.hofer@gmail.com>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

Christoph Hofer

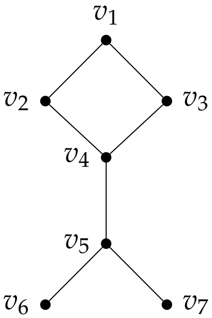


Florian Graf

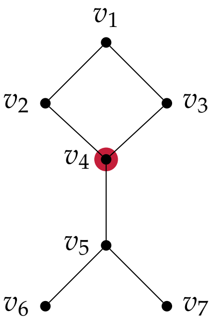
Marc Niethammer
MarcNiethammer

Roland Kwitt
rkwitt1982

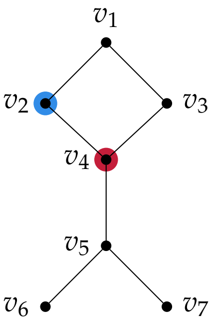
Message passing in graphs



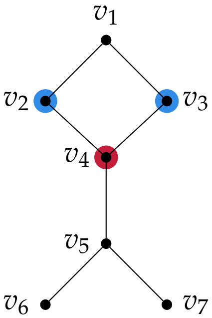
Message passing in graphs



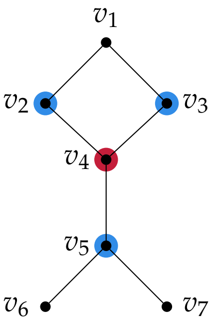
Message passing in graphs



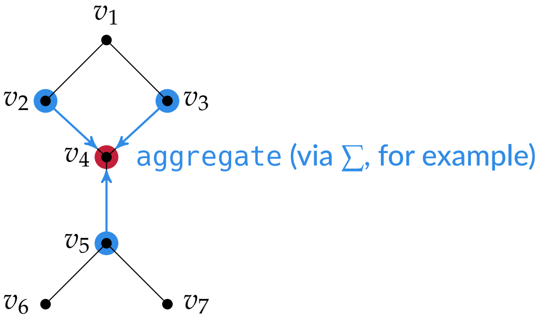
Message passing in graphs



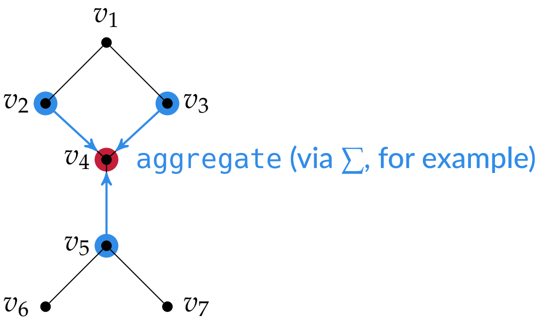
Message passing in graphs



Message passing in graphs



Message passing in graphs



Repeat this process multiple times and update the vertex representations accordingly.
Use a `readout` function to obtain a graph-level representation.

Learning graph filtrations

Motivation

- When classifying graphs with TDA, we often employ a *filter function* $f\colon \mathfrak{V} \to \mathbb{R}$. For example, $f(v) := \deg(v)$ is commonly employed.
- We typically extend f to a full graph G by setting $f(\{u, v\} := \max\{f(u), f(v)\}$.
- Can we *learn* f end-to-end?

Learning graph filtrations

Details

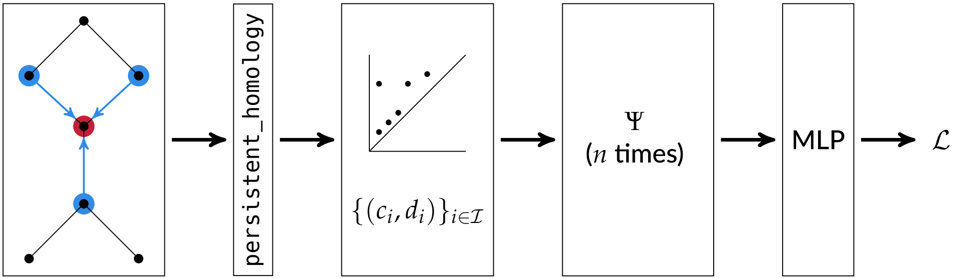
Use a differentiable *coordinatisation* scheme of the form $\Psi\colon \mathcal{D} \to \mathbb{R}$. Letting $p := (c, d)$ for a tuple in a diagram, we have

$$\Psi(p) := \frac{1}{1 + \|p - c\|_1} - \frac{1}{1 + \left||r| - \|p - c\|_1\right|},$$

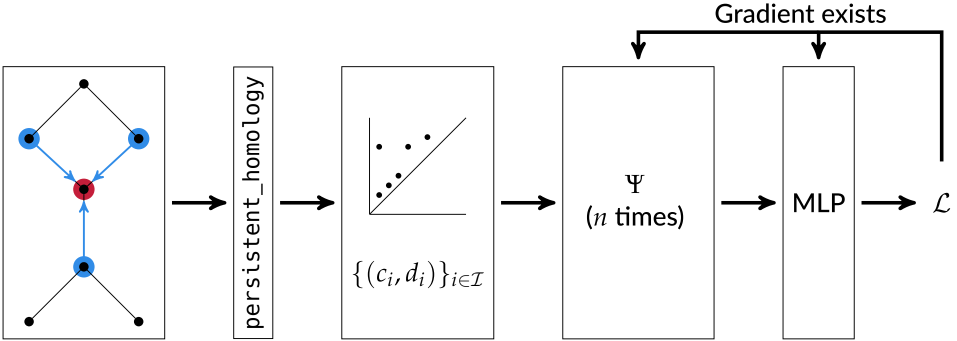
with $c \in \mathbb{R}^2$ and $r \in \mathbb{R}$ being *trainable* parameters. The whole diagram is represented as a sum over each individual projections.

Using n different coordinatisations, we obtain a differentiable embedding of a persistence diagram into $\mathbb{R}^n$.

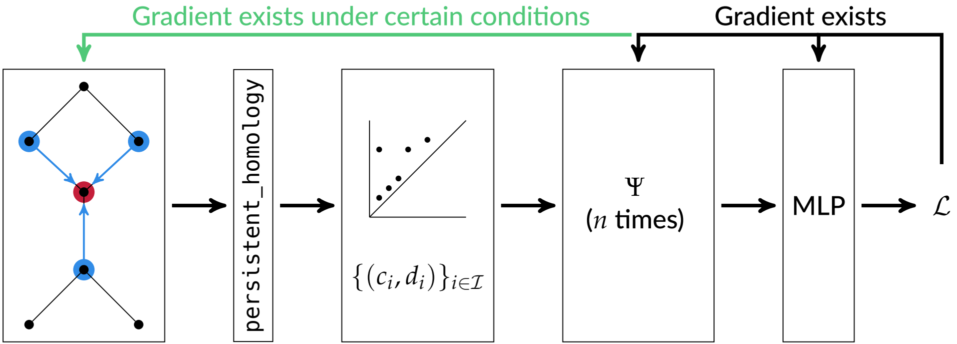
A `readout` function based on persistent homology



A `readout` function based on persistent homology



A `readout` function based on persistent homology



Obtaining a filter function f

Use a single GIN-ϵ layer with one level of message passing (`1-GIN`) with hidden dimensionality 64, followed by a two-layer MLP.

`GIN-1`, $h = 64$ $\longrightarrow$ `MLP`$(64, 64, 1)$ with sigmoid activation

Hence, $f\colon \mathfrak{V} \to [0, 1]$.

Using this in practice

- If f is *injective* on the graph vertices, the gradient exists.
- We can initialise f using the vertex degree or uniform weights (plus a symbolic perturbation to ensure gradient existence).
- Simple integration into existing architectures.

Using this in practice

- If f is *injective* on the graph vertices, the gradient exists.
- We can initialise f using the vertex degree or uniform weights (plus a symbolic perturbation to ensure gradient existence).
- Simple integration into existing architectures.

Method	IMDB-BINARY	IMDB-MULTI
1-GIN (GFL)	74.5 ± 4.6	49.7 ± 2.9
1-GIN (SUM)	73.5 ± 3.8	50.3 ± 2.6
1-GIN (SP)	73.0 ± 4.0	50.5 ± 2.1
Baseline	72.7 ± 4.6	49.9 ± 4.0
PH	68.9 ± 3.5	46.1 ± 4.2

Graph filtration learning

- We are able to *learn* a scalar-valued filtration function in an end-to-end fashion.
- The `readout` function integrates nicely into existing architectures.
- Predictive performance is better than ‘raw’ persistent homology (with only a single level of message passing).

Summary

- Persistent homology *can* be made differentiable!
- Topological features improve representation learning tasks.
- This is only just the beginning; need to handle higher-dimensional features, different filtrations, and much more...

❤ My co-authors, in particular Max, Michael, and Roland for providing figures, illustrations, and animations.