Multidimensional Persistence Module Classification via Lattice-Theoretic Convolutions

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

To write...

1 Introduction

Persistent homology has the ability to discern both the global topology [] and local geometry [] of finite metric spaces (e.g. embedded weighted graphs, point clouds in $\mathbb{R}^d$) making it a befitting feature for the purposes of training a neural network. Single-dimensional homological persistence has drawn recent attention in deep learning [7, 11, 2]. This is, in part, due to a wide range of efficient software libraries [9, 6, 1] for computing barcodes (in the "west-coast" lingo) of filtrations (e.g. Rips, alpha, Cech, sub-levelset). Barcodes provide a compact shape descriptor for metric space data that is stable with respect to Gromov-Hausdorff distance []. In a seminal paper [4], Carlsson and Zomorodian show there is no such compact description for multi-parameter persistence.

2 Backgound

Due to space constraints, we can offer only a brief overview of multiparameter persistent homology. For a primer on persistent homology, see [5, 3]; for multiparameter persistent homology, see []. An introduction to lattices may be found in [].

2.1 Rips complexes and persistent homology

Let $(\mathcal{M}, d)$ be a finite metric space. The *Vietoris-Rips complex* of $\mathcal{M}$ at scale r is the abstract simplicial complex $\mathrm{Rips}_r(\mathcal{M})$ whose simplices are subsets of $\mathcal{M}$ of diameter at most r. There is a natural inclusion $\mathrm{Rips}_r(\mathcal{M}) \to \mathrm{Rips}_{r'}(\mathcal{M})$ for $r \leq r'$.

Applying the simplicial homology functor (with coefficients in a field k) H_i to $\mathrm{Rips}_r(\mathcal{M})$ produces a sequence of vector spaces $PH_i(r)$. The inclusions $\mathrm{Rips}_r(\mathcal{M}) \to \mathrm{Rips}_{r'}(\mathcal{M})$ induce maps $PH_i(r) \to PH_i(r')$, producing the data of *persistence module*. This structure can be compactly described as a functor from $\mathbb{R}$, viewed as a category via its standard order structure, to the category $\mathbf{Vect}_k$ of vector spaces over k. The simplicity of the category $\mathbb{R}$ gives these persistence modules simple structure: they decompose as direct sums of interval modules $I_{[a,b)}$, which have $I_{[a,b)}(r) = k$ for $a \leq r < b$ and zero otherwise. The maps are the identity where possible and the zero map otherwise.

This representation theoretic fact gives a representation[1] of $\mathbb{R}$-indexed persistence modules via *barcodes* or *persistence diagrams*. Each bar $I_{[a,b)}$ in the barcode represents a homology class which is *born* at a and *dies* at b.

[1]Heh.

Submitted to the Topological Data Analysis and Beyond Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

2.2 Multiparameter persistence

The Rips construction produces a filtration of simplicial complexes from a finite metric space; it is natural to consider the behavior of the homology functor over a pair of coherent filtrations. Consider a finite metric space $(\mathcal{M}, d)$ and a filtration function $\rho : \mathcal{M} \to \mathbb{R}$. This data specifies a bifiltration of simplicial complexes given by

$$\mathbb{X}_{r,t} = \mathrm{Rips}_r x \in \mathcal{M} \mid \rho(x) \leq t.$$

There is a natural inclusion $\mathbb{X}_{r,t} \hookrightarrow \mathbb{X}_{r',t'}$ whenever $(r,t) \leq (r',t')$ in the lattice $\mathbb{R} \times R$. Composing with the homology functor produces a 2-parameter persistence module

$$PH_i : \mathbb{R}_+ \times \mathbb{R} \to \mathbf{Vect}; \quad (r,t) \mapsto H_i(X_{r,t}).$$

If we assume that (i) PH_i stabilizes for $r \geq R$ and $t \geq T$ for sufficiently large R, T and (ii) the induced map on homology $H_i(\mathbb{X}_{r,t}) \to H_i(\mathbb{X}_{r',t'})$ is an isomorphism for all but finitely many pairs $(r,t) \leq (r',t')$, then we can restrict the domain of our persistence module PH_i—possibly after a reparameterization of the filtration—to a finite order lattice $L = [m] \times [n]$, where $[n] = \{0, 1, 2, \ldots, n\}$, obtaining a persistence module $M : L \to \mathbf{Vect}$. More generally, this model accepts as inputs signals on any finite lattice L with features extracted from a generalized persistence module supported on L.

While there does not exist a discrete set of invariants for M, we can extract meaningful features. Two particularly informative types of features are the *Hilbert function*

$$\mathrm{Hilb} : L \to \mathbb{Z}_+; \quad \mathbf{x} \mapsto \dim(M_{\mathbf{x}}),$$

and the *multi-graded Betti numbers*[2] $\xi_j : L \to \mathbb{Z}_+$, for $j = 0, 1, 2$. For $M = PH_i$ as above, the Hilbert function counts the number of connected components ($i = 0$), cycles ($i = 1$), or higher dimensional voids ($i > 1$) of the complex $\mathbb{X}_{r,t}$ at each $(r,t) \in L$. The multi-graded Betti numbers, on the other hand, capture information about locations of births and deaths of persistence classes.

2.3 Lattice-theoretic signal processing

[10]

3 Lattice Convolutional Neural Networks

Convolutions for signals defined over $\mathbb{R}^n$ taken as an abelian group are widely used in signal processing. In particular, two-dimensional convolutions have served as an easily parameterized and efficient set of linear operations adapted to the structure of images. Their extreme utility in computer vision problems is owed to the translation equivariance properties of images: humans naturally recognize an image translated via an additive reparameterization as equivalent to the original.

The data of a multidimensional persistence module is also indexed by $\mathbb{R}^n$ or a regular finite subset thereof, but its natural algebraic structure is not that of an abelian group. Rather, with its partial order structure, the indexing set is a lattice. In processing signals associated with the persistence module, it may be useful to take this structure into account rather than imposing the abelian group structure implied by standard convolutions.

To this end, we construct a lattice convolution-based neural network layer suitable for use with features originating from multidimensional persistence modules. We specialize the convolutions described in Section 2.3 to the particular case of regular finite sublattices of $\mathbb{R}^2$. The meet and join operations are easily computed elementwise:

$$(r,t) \wedge (r',t') = (\min(r,r'), \min(t,t')); \quad (r,t) \vee (r',t') = (\max(r,r'), \max(t,t')).$$

A lattice convolution layer takes as input an N_{in}-dimensional signal $f : [m] \times [n] \to \mathbb{R}^{N_{\text{in}}}$ and outputs an N_{out}-dimensional signal $[m] \times [n] \to \mathbb{R}^{N_{\text{out}}}$. The layer's parameters are given by a function

[2] *Caveat lector:* the multi-graded Betti numbers are not the same as the topological Betti numbers $\beta_i(\mathbb{X}) = \dim(H_i(\mathbb{X}))$.

$g : [m] \times [n] \to \mathbb{R}^{N_{\text{out}} \times N_{\text{in}}}$. If we label the entries of $f(x, y)$ by f_i and the entries of $g(x, y)$ by g^i_j The layer then acts by

$$\text{MeetConv}(f)(x, y)^j = \sum_i (f_i *_\wedge g^i_j)(x, y) = \sum_i \sum_{(a,b) \in [m] \times [n]} f_i(x \wedge a, y \wedge b) g^i_j(a, b)$$

in the case of convolution with respect to the meet operation, and

$$\text{JoinConv}(f)(x, y)^j = \sum_i (f_i *_\vee g^i_j)(x, y) = \sum_i \sum_{(a,b) \in [m] \times [n]} f_i(x \vee a, y \vee b) g^i_j(a, b)$$

in the case of convolution with respect to the join operation.

Convolutional neural networks are uesful in part because the convolution kernels (here the functions g) can have very small support, enforcing locality and reducing the number of parameters that must be learned. In the standard convolutional setting, these kernels are implicitly supported in a neighborhood of the origin, but the location of the kernel is not usually explicitly specified. In the lattice setting, we do need to specify where the kernel resides. Just as when we treat our domain as an abelian group, the kernel should be supported near the identity, when we treat our domain as a lattice, the kernel should be supported near the neutral element of the operation. That is, for a meet convolution, g should have support near the maximum (m, n), and for a join convolution, g should have support near the minimum $(0, 0)$.

4 Experiments

We use a small portion of the Princeton ModelNet dataset as a source of finite metric spaces. This dataset consists of hundreds of 3-dimensional CAD models representing objects from 40 classes. We select two of the classes and sample points from the 3d models to produce finite metric spaces embedded in $\mathbb{R}^3$. We then compute the corresponding multidimensional persistence modules, from which we produce features used as an input to a convolutional neural net classifier.

The pipline thus begins with a 3d model, which is sampled to produce a point cloud in $\mathbb{R}^3$. This point cloud then produces a bifiltered simplicial complex, whose persistent homology we calculate using RIVET [8], producing lattice-indexed signals given by the Hilbert function and the multi-graded Betti numbers. These are then passed to the classifier, which produces a class prediction.

As the filter function on these data sets, we use

$$\rho_{\text{codense}}(x; k) = \frac{k \operatorname{diam}(\mathcal{M})}{\sum_{y \in N_k(x)} d(x, y)},$$

where $N_k(x)$ is the set of the k nearest neighbors to x. This is the *codensity* filtration, so named because the points in the densest regions of $\mathcal{M}$ will appear first. A folk theorem is that the two-parameter persistent homology of a Rips/codensity bifiltration is stable under non-Hausdorff perturbations: *the (Rips) persistent homology of a point sample and another obtained by adding a small number of points at random are close with respect to the interleaving distance*.

We compare the performance of two convolutional networks on this classification task. One uses the lattice-convolution based layers described in Section 3, and the other uses standard convolutional layers. Each has three convolutional layers followed by three fully connected layers.

5 Discussion

Discuss.

References

[1] Ulrich Bauer. Ripser: Efficient computation of Vietoris-Rips persistence barcodes. *arXiv:1908.02518 [cs, math]*, August 2019.

[2] Rickard Brüel-Gabrielsson, Bradley J. Nelson, Anjan Dwaraknath, Primoz Skraba, Leonidas J. Guibas, and Gunnar Carlsson. A Topology Layer for Machine Learning. *arXiv:1905.12200 [cs, math, stat]*, April 2020.

[3] Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46:255–308, 2009.

[4] Gunnar Carlsson and Afra Zomorodian. The Theory of Multidimensional Persistence. *Discrete & Computational Geometry*, 42(1):71–93, July 2009.

[5] Robert Ghrist. Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society*, 45:61–75, 2008.

[6] Gregory Henselman and Robert Ghrist. Matroid filtrations and computational persistent homology. *arXiv:1606.00199 [math]*, October 2017.

[7] Christoph Hofer, Roland Kwitt, Marc Niethammer, and Andreas Uhl. Deep learning with topological signatures. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1634–1644. Curran Associates, Inc., 2017.

[8] Michael Lesnick and Matthew Wright. Interactive visualization of 2-d persistence modules. *arXiv:1512.00180 [cs, math]*, December 2015.

[9] 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):17, August 2017.

[10] Markus Pueschel. A discrete signal processing framework for meet/join lattices with applications to hypergraphs and trees. May 2019.

[11] Chi Seng Pun, Kelin Xia, and Si Xian Lee. Persistent-Homology-based Machine Learning and its Applications – A Survey. *arXiv:1811.00252 [math]*, November 2018.