Journal of Machine Learning Research 25 (2024) 1-8 Submitted 1/24; Revised 11/24; Published 11/24

TopoX: A Suite of Python Packages for Machine Learning on Topological Domains

Mustafa Hajij* mhajij@usfca.edu
Mathilde Papillon* papillon@ucsb.edu
Florian Frantzen* florian.frantzen@cs.rwth-aachen.de
Jens Agerberg jensag@kth.se
Ibrahem AlJabea ialjab2@lsu.edu
Rubén Ballester ruben.ballester@ub.edu
Claudio Battiloro cbattiloro@hsph.harvard.edu
Guillermo Bernárdez guillermo.bernardez@upc.edu
Tolga Birdal tbirdal@imperial.ac.uk
Aiden Brent aidenjb81@gmail.com
Peter Chin peter.chin@dartmouth.edu
Sergio Escalera sescalera@ub.edu
Simone Fiorellino simone.fiorellino@uniroma1.it
Odin Hoff Gardaa odin.garda@uib.no
Gurusankar Gopalakrishnan ggopalakrishnan@dons.usfca.edu
Devendra Govil dgovil@dons.usfca.edu
Josef Hoppe hoppe@cs.rwth-aachen.de
Maneel Reddy Karri mkarri@dons.usfca.edu
Jude Khouja Jude@latynt.com
Manuel Lecha manuellecha@ub.edu
Neal Livesay nlivesay@ptc.com
Jan Meißner philipp.meissner@rwth-aachen.de
Soham Mukherjee mukher26@purdue.edu
Alexander Nikitin alexander.nikitin@aalto.fi
Theodore Papamarkou theo@zjnu.edu.cn
Jaro Prílepok jaroslav.prilepok@student.manchester.ac.uk
Karthikeyan Natesan Ramamurthy knatesa@us.ibm.com
Paul Rosen prosen@sci.utah.edu
Aldo Guzmán-Sáenz aldo.guzman.saenz@ibm.com
Alessandro Salatiello salatiello.alessandro@gmail.com
Shreyas N. Samaga ssamaga@purdue.edu
Simone Scardapane simone.scardapane@uniroma1.it
Michael T. Schaub schaub@cs.rwth-aachen.de
Luca Scofano luca.scofano@uniroma1.it
Indro Spinelli indro.spinelli@uniroma1.it
Lev Telyatnikov lev.telyatnikov@uniroma1.it
Quang Truong cong.minh.quang.truong.th@dartmouth.edu
Robin Walters r.walters@northeastern.edu
Maosheng Yang m.yang-2@tudelft.nl
Olga Zaghen o.zaghen@uva.nl
Ghada Zamzmi ghadh@mail.usf.edu
Ali Zia ali.zia@anu.edu.au
Nina Miolane ninamiolane@ucsb.edu

Editor: Joaquin Vanschoren

Abstract

We introduce `TopoX`, a Python software suite that provides reliable and user-friendly building blocks for computing and machine learning on topological domains that extend graphs: hypergraphs, simplicial, cellular, path and combinatorial complexes. `TopoX` consists of three packages: `TopoNetX` facilitates constructing and computing on these domains, including working with nodes, edges and higher-order cells; `TopoEmbedX` provides methods to embed

∗. *Equal contribution as first author

topological domains into vector spaces, akin to popular graph-based embedding algorithms such as node2vec; `TopoModelX` is built on top of `PyTorch` and offers a comprehensive toolbox of higher-order message passing functions for neural networks on topological domains. The extensively documented and unit-tested source code of `TopoX` is available under MIT license at https://pyt-team.github.io/.

Keywords: topological deep learning, topological neural networks, graph neural networks, machine learning, Python packages.

1 Introduction

Deep learning traditionally operates within Euclidean domains, focusing on structured data like images (Krizhevsky et al., 2012) and sequences (Sutskever et al., 2014). However, to handle more diverse data types, geometric deep learning (GDL; Bronstein et al., 2021; Zhou et al., 2020; Cao et al., 2020) has emerged. GDL extends deep learning to non-Euclidean data by leveraging geometric regularities like symmetries and invariances. Recently, topological deep learning (TDL; Hajij et al., 2022) has gained attention, exploring models beyond traditional graph-based abstractions to process data with multi-way relations, such as simplicial complexes and hypergraphs. These extensions allow for the representation of diverse data domains encountered in scientific computations (Feng et al., 2019; Schaub et al., 2020; Hajij et al., 2020; Schaub et al., 2021; Roddenberry et al., 2021; Giusti et al., 2023; Yang and Isufi, 2023; Barbarossa and Sardellitti, 2020). Despite theoretical advancements, practical implementation faces challenges due to the lack of accessible software libraries supporting deep learning models with higher-order structures.

In this paper, we present `TopoX`, an open-source suite of Python packages designed for machine learning and deep learning operations in topological domains. `TopoX` is organized into three Python packages: `TopoNetX`, `TopoEmbedX`, and `TopoModelX` (see Figure 1). These packages enhance and generalize functionalities found in popular mainstream graph computations and learning tools, enabling them on topological domains. What sets `TopoX` apart is its abstract general design and the exploitation of the resulting modeling flexibility in the implementation of a broad spectrum of topological domains and TDL models (see Table 1). Further, every domain in `TopoNetX` offers utilities to work with various components such as nodes, edges, and higher-order cells. `TopoNetX` also supports computations using incidence matrices, (co)adjacency matrices, and up, down, and Hodge Laplacians. From a representation learning point of view, `TopoEmbedX` provides methods for embedding topological domains, or parts of these domains, into Euclidean domains. `TopoModelX` offers a wide range of TDL models based on a comprehensive implementation of higher-order message passing (Hajij et al., 2022), built on the `PyTorch` framework (Paszke et al., 2019).

The core objectives of `TopoX` are to: facilitate topological research through foundational code and algorithm dissemination; broaden access to the field with user-friendly topological learning tools for the ML community; serve as a resource for topological learning enriched with examples, notebooks, and visualizations; and provide a unified API for topological domains. This API supports the generalization of topological spaces across scientific computations, enhancing interoperability, productivity, learning, and collaboration, while reducing maintenance, promoting code portability, and enabling parallel computing.

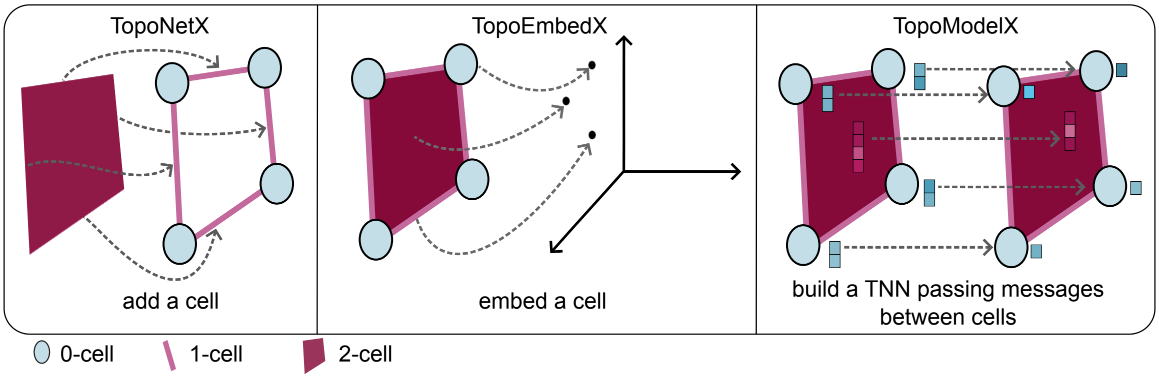

Figure 1: Building blocks from the three packages of the `TopoX` software suite. Left: `TopoNetX` enables building topological domains such as cell complexes. The figure demonstrates adding a 2-cell on a cell complex with `TopoNetX`. Middle: `TopoEmbedX` enables embedding of topological domains inside a Euclidean space. The figure illustrates embedding a 0-cell, a 1-cell and a 2-cell from a cell complex inside a Euclidean space with `TopoEmbedX`. Right: building a topological neural network (TNN) that processes data via higher-order message passing on a cell complex with `TopoModelX`.

2 Implementation Overview

The `TopoNetX` package is organized into three main modules: `classes`, `algorithms`, and `transform`. The `classes` module implements numerous common topological domains, such as `SimplicialComplex`, `CellComplex`, and more; inheriting from the abstract class `Complex`. The `algorithms` module implements spectral methods, distance computation, and connected component analysis in topological domains. The `transform` module facilitates conversions between different topological domains. `TopoNetX` uses `NumPy` and `SciPy` backends, and offers toy datasets and examples to facilitate learning and improve understanding.

The `TopoEmbedX` package supports the learning of representations for all topological domains available in `TopoNetX`. This package contains the module `classes`, which implements topological representation learning algorithms that generalize popular graph-based representation learning algorithms, including `DeepCell` and `Cell2Vec` (Hajij et al., 2020). The `TopoEmbedX` package has an API inspired by `scikit-learn` (Pedregosa et al., 2011) and utilizes the KarateClub (Rozemberczki et al., 2020) backend. While `TopoEmbedX` leverages methods from KarateClub, its functionalities are distinguished by a rigorous foundation in topological representation learning from Hajij et al. (2022). This foundation enables unique, theory-driven contributions, extending the package's capabilities to embed all topological domains in `TopoNetX` with a single unifying API, not just graphs as in KarateClub.

`TopoModelX` is a Python package for topological deep learning, providing efficient tools to implement topological neural networks (TNNs). The package consists of two main modules: `base` and `nn`. The `base` module implements higher-order message passing methods, allowing the construction of general-purpose TNNs using the tensor diagram formalism introduced in Hajij et al. (2022) and surveyed in Papillon et al. (2023b). The `nn` module implements various TNNs on popular topological domains, such as simplicial complexes, cell complexes, hypergraphs, and combinatorial complexes. Each implementation includes a corresponding Jupyter notebook tutorial for a user-friendly initiation into TDL. `TopoModelX` leverages `PyTorch` and `PyG` (PyTorch Geometric) backends (Fey and Lenssen, 2019). The `TopoModelX` package is the outcome of the unifying TDL framework introduced in Hajij et al. (2022) which lead to the coding challenge that crowd-sourced the implementations of TDL models (Papillon et al., 2023a).

3 Comparison and Interaction with Other Packages

The libraries most closely related to `TopoNetX` are `NetworkX` (Hagberg et al., 2008) and `HypernetX` (Liu et al., 2021). They facilitate computations on graphs and hypergraphs, respectively. `TopoNetX` utilizes a similar API to these two libraries to facilitate rapid adoption of topological domains, such as simplicial complexes, cell complexes, and colored hypergraphs. The `XGI` package (Landry et al., 2023) offers hypergraph functionalities similar to `HypernetX` with additional support for simplicial complexes and directed hypergraphs.

The closest package to `TopoEmbedX` is `KarateClub` (Rozemberczki et al., 2020), which is a Python package consisting of methods for unsupervised learning on graph-structured data. `TopoEmbedX` extends the functionality of `KarateClub` to topological domains supported in `TopoNetX`. `PyG` and `DGL` (Wang et al., 2019), which are two popular geometric deep learning packages that support deep learning models on graphs, are closely related to `TopoModelX`. The `DHG` (Deep HyperGraph) package (Feng et al., 2019) supports deep learning models defined on hypergraphs. All three of our packages feature a continuous integration pipeline and are well-tested with code coverage of $\geq 95\%$ per package. This is comparable or better than many related graph-based learning libraries, such as `PyG`, `DGL`, `XGI`, `NetworkX`, and `HyperNetX`. Table 1 provides a comparison between `TopoX` and other Python packages.

4 Usage: Elementary Examples

`TopoNetX` provides a user-friendly interface which allows to create a complex in two main steps: first, instantiate the complex; second, add cells to that complex, as shown in the first three lines of the code snippet below. Furthermore, processing data on a complex requires matrices that describe the (co)adjacency of the incidence relations among cells.

```python
import toponetx as tnx
cell_complex = tnx.CellComplex()
cell_complex.add_cell([1, 2, 3, 4], rank=2)
cell_complex.add_cell([1, 2, 5], rank=2)
L2 = cell_complex.hodge_laplacian_matrix(2)
```

The following code snippet shows how `TopoEmbedX` embeds edges of the Stanford bunny dataset using the Cell2Vec algorithm (Hajij et al., 2020):

```python
import topoembedx as tex
cell_complex = tnx.datasets.stanford_bunny("cell")
model = tex.Cell2Vec()
model.fit(cell_complex, nbhd_type="adj", nbhd_dim={"adj": 1})
embeddings = model.get_embedding()
```

Comparison between `TopoNetX` and other Python packages		
Packages	Domains	Operations
`TopoNetX`	Graphs, colored hypergraphs, simplicial complexes, path complexes, cell complexes, combinatorial complexes	`add_cell`, `add_node`, `add_simplex`, `adjacency_matrix`, `coadjacency_matrix`, `incidence_matrix`, `hodge_laplacian_matrix`
`NetworkX`	Graphs	`add_node`, `add_edge`, `adjacency_matrix`, `incidence_matrix`, `laplacian_matrix`
`HyperNetX`	Hypergraphs	`add_node`, `add_edge`, `adjacency_matrix`, `incidence_matrix`
`XGI`	Simplicial complexes, hypergraphs, dihypergraph	`add_node`, `add_edge`, `adjacency_matrix`, `boundary_matrix`, `hodge_laplacian`
Comparison between `TopoEmbedX` and other Python packages		
Packages	Domains	Embedding algorithms
`TopoEmbedX`	Graphs, colored hypergraphs, simplicial complexes, path complexes, cell complexes, combinatorial complexes	Cell2Vec, DeepCell, CellDiff2Vec, HigherOrderLaplacianEigenMap, HOPE, HOGLEE,
`Karateclub`	Graphs	Node2Vec, Graph2Vec, Diff2Vec, GL2Vec, IGE, Role2Vec, GraRep
Comparison between `TopoModelX` and other Python packages		
Packages	Domains	Push-forward operators
`TopoModelX`	Graphs, (colored) hypergraphs, simplicial complexes, path complexes, cell complexes, combinatorial complexes	(Higher order) message passing, merge operator, split operator
`DGL`	Graphs	Message passing
`DHG`	Graphs, hypergraphs	Message passing, hypergraph message passing
`PyG`	Graphs	Message passing

Table 1: `TopoX` provides a user-friendly and comprehensive suite for building blocks and computing on topological domains. The table shows a comparison between `TopoX` and other Python packages.

The following code snippet shows how to instantiate, and run the forward-pass of an example of simplicial neural network (SAN; Battiloro et al., 2024) with `TopoModelX`:

```
from topomodelx.nn.simplicial.san import SAN
from topomodelx.utils.sparse import from_sparse
sc = tnx.datasets.karate_club("simplicial")
x = torch.tensor([*sc.get_simplex_attributes("edge_feat").values()])
L1_down = from_sparse(sc.down_laplacian_matrix(rank=1))
L1_up = from_sparse(sc.up_laplacian_matrix(rank=1))
san_model = SAN(in_channels=2, hidden_channels=16, n_layers=2)
san_model(x, L1_down, L1_up)
```

`TopoNetX` provides a high-level declarative interface. In `TopoEmbedX`, each embedding algorithm is compatible with all complexes available in `TopoNetX`. In `TopoModelX`, TNNs are classified according to the topological domain upon which they are defined. In each package, the directories that contain examples and notebooks offer an abundance of code snippets to assist users in starting their journey with `TopoX`.

Acknowledgments and Disclosure of Funding

M. H. acknowledges support from the National Science Foundation, award DMS-2134231. N. M. and M. P. acknowledge support from the National Science Foundation, Award DMS-2134241. S. E. and R. B. acknowledge support from Spanish project PID2022-136436NB-I00 and ICREA under the ICREA Academia programme. R. B. acknowledges support from an FPU contract FPU21/00968, and by the Departament de Recerca i Universitats de la Generalitat de Catalunya (2021 SGR 00697). T. B. acknowledges support from the Engineering and Physical Sciences Research Council [grant EP/X011364/1]. T. K. D. acknowledges support from the National Science Foundation, Award CCF 2049010. N. L. acknowledges support from the Roux Institute and the Harold Alfond Foundation. T. P. acknowledges support from Zhejiang Normal University (distinguished professorship start-up grant YS304124985). R. W. acknowledges support from the National Science Foundation, Award DMS-2134178. P. R. acknowledges support from the National Science Foundation, Award IIS-2316496. M. T. S. and F.F. acknowledge funding by the Ministry of Culture and Science (MKW) of the German State of North Rhine-Westphalia (NRW Rückkehrprogramm). M.T.S and J. H. acknowledge funding from the European Union (ERC, HIGH-HOPeS, 101039827). Views and opinions expressed are however those of M. T. S. only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency; neither the European Union nor the granting authority can be held responsible for them. The authors acknowledge the ICML Topological Deep Learning challenge that jump-started the development of `TopoModelX`.

References

Sergio Barbarossa and Stefania Sardellitti. Topological signal processing over simplicial complexes. *IEEE Transactions on Signal Processing*, 68:2992–3007, 2020.

Claudio Battiloro, Lucia Testa, Lorenzo Giusti, Stefania Sardellitti, Paolo Di Lorenzo, and Sergio Barbarossa. Generalized simplicial attention neural networks. *IEEE Transactions on Signal and Information Processing over Networks*, 10:833–850, 2024.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: grids, groups, graphs, geodesics, and gauges. *arXiv preprint:2104.13478*, 2021.

Wenming Cao, Zhiyue Yan, Zhiquan He, and Zhihai He. A comprehensive survey on geometric deep learning. *IEEE Access*, 8:35929–35949, 2020.

Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01), 2019.

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Lorenzo Giusti, Claudio Battiloro, Lucia Testa, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Cell attention networks. In *IJCNN*, pages 1–8, 2023.

Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab (LANL), Los Alamos, NM (United States), 2008.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell complex neural networks. *NeurIPS 2020 Workshop TDA and Beyond*, 2020.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal K Dey, Soham Mukherjee, Shreyas N Samaga, et al. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, volume 25, pages 1097–1105, 2012.

Nicholas W. Landry, Maxime Lucas, Iacopo Iacopini, Giovanni Petri, Alice Schwarze, Alice Patania, and Leo Torres. XGI: A Python package for higher-order interaction networks. *Journal of Open Source Software*, 8(85):5162, May 2023.

Xu T Liu, Jesun Firoz, Andrew Lumsdaine, Cliff Joslyn, Sinan Aksoy, Brenda Praggastis, and Assefaw H Gebremedhin. Parallel algorithms for efficient computation of high-order line graphs of hypergraphs. In *2021 IEEE 28th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, pages 312–321. IEEE, 2021.

Mathilde Papillon, Mustafa Hajij, Audun Myers, , Helen Jenne, Johan Mathe, Theodore Papamarkou, Aldo Guzmán-Sáenz, Neal Livesay, Tamal Dey, Abraham Rabinowitz, Aiden Brent, Alessandro Salatiello, Alexander Nikitin, Ali Zia, Claudio Battiloro, Dmitrii Gavrilev, German Magai, Gleb Bazhenov, Guillermo Bernardez, Indro Spinelli, Jens Agerberg, Kalyan Nadimpalli, Lev Telyatninkov, Luca Scofano, Lucia Testa, Manuel Lecha, Maosheng Yang, Mohammed Hassanin, Odin Hoff Gardaa, Olga Zaghen, Paul Hausner, Paul Snopoff, Rubén Ballester, Sadrodin Barikbin, Sergio Escalera, Simone Fiorellino, Henry Kvinge, Karthikeyan Natesan Ramamurthy, Paul Rosen, Robin Walters, Shreyas N. Samaga, Soham Mukherjee, Sophia Sanborn, Tegan Emerson, Timothy Doster, Tolga Birdal, Abdelwahed Khamis, Simone Scardapane, Suraj Singh, Tatiana Malygina, Yixiao Yue, and Nina Miolane. ICML 2023 topological deep learning challenge: Design and results. In *Proceedings of 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML)*, volume 221, pages 3–8. PMLR, 2023a.

Mathilde Papillon, Sophia Sanborn, Mustafa Hajij, and Nina Miolane. Architectures of topological deep learning: a survey on topological neural networks. *arXiv preprint arXiv:2304.10031*, 2023b.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: an imperative style, high-performance deep learning library. In *NeurIPS*, volume 32, pages 8024–8035. 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

T. Mitchell Roddenberry, Nicholas Glaze, and Santiago Segarra. Principled simplicial neural networks for trajectory prediction. In *ICML*, pages 9020–9029. PMLR, 2021.

Benedek Rozemberczki, Oliver Kiss, and Rik Sarkar. Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20)*, page 3125–3132. ACM, 2020.

Michael T. Schaub, Austin R. Benson, Paul Horn, Gabor Lippner, and Ali Jadbabaie. Random walks on simplicial complexes and the normalized Hodge 1-Laplacian. *SIAM Review*, 62(2):353–391, 2020.

Michael T. Schaub, Yu Zhu, Jean-Baptiste Seby, T. Mitchell Roddenberry, and Santiago Segarra. Signal processing on higher-order networks: livin' on the edge... and beyond. *Signal Processing*, 187:108149, 2021.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep Graph Library: a graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*, 2019.

Maosheng Yang and Elvin Isufi. Convolutional learning on simplicial complexes. *arXiv preprint arXiv:2301.11163*, 2023.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: a review of methods and applications. *AI Open*, 1:57–81, 2020.