

Topological Graph Neural Networks

Edward De Brouwer



DBSSE

ETH zürich

KU LEUVEN

Topological layers for graph classification

TOGL



Topological Graph Neural Networks

Max Horn^{1,2,*} Edward De Brouwer^{1,*} Michael Moor^{1,2} Yves Moreau²
Bastian Rieck^{3,2,*} Karsten Borgwardt^{1,2}

¹Department of Biosystems Science and Engineering, ETH Zurich, 4058 Basel, Switzerland

²IBM Research, Zollikofen, Switzerland

³UG-STATIS, KU LEUVEN, 3001 Leuven, Belgium

*These authors contributed equally.

[†]These authors jointly supervised this work.

Abstract

Graph neural networks (GNNs) are a powerful architecture for tackling graph learning tasks, yet have been shown to be oblivious to essential substructures, such as cycles. We present TOGL, a novel layer that incorporates global topological information of a graph using persistent homology. TOGL can be easily integrated into any type of GNN and is readily more expressive in terms of the Heister-Lefschetz test of isomorphism. Augmenting GNNs with our layer leads to beneficial predictive performance for graph and node classification tasks, both on synthetic data sets, which can be classified by humans using their topology but not by ordinary GNNs, and on real-world data.

1. Introduction

Graphs are a natural description of structured data sets in many domains, including bioinformatics, image processing, and social network analysis. Numerous methods address graph learning problems such as graph classification or node classification. Graph neural networks (GNNs) describe a flexible set of architectures for graph learning tasks and have won many successful applications over recent years [1]. At their core, many GNNs are based on iterative message passing schemes. Since these schemes are collating information over the neighbors of every node, GNNs cannot reasonably capture certain simple topological structures in graphs, such as cycles [2]. These structures, however, are relevant for certain applications, such as the analysis of molecular graphs, whose classification necessitates knowledge about connectivity information [3, 4].

By contrast, methods based on topological features, commonly summarized under the term of topological data analysis (TDA), have shown promising results in machine learning tasks. Focusing on coarse structure—such as the presence or absence of cycles—they can be used to provide multi-scale representations that capture the shape of complex, structured and unstructured data sets. In this paper, we propose a **Topological Graph Layer (TOGL)** that can be easily integrated into any GNN to make it “topology-aware”. We then demonstrate its generic use to augment existing GNNs and increase their expressivity in graph learning tasks. Figure 1 provides a motivational example that showcases the potential benefits of using topological information: high predictive performance is reached earlier for a smaller number of layers.

Max Horn
@ExpectationMax



Edward De Brouwer
@EdwardOnBrew



Michael Moor
@Michael_D_Moor



Yves Moreau



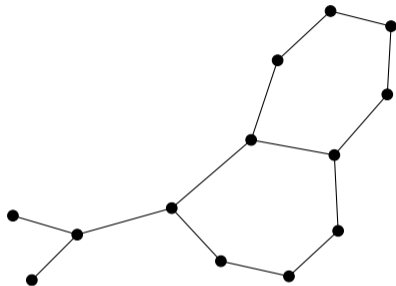
Bastian Rieck
@Pseudomanifold



Karsten Borgwardt
@kmborgwardt

M. Horn[†], E. De Brouwer[†], M. Moor, Y. Moreau, B. Rieck^{†‡} and K. Borgwardt[‡],
'Topological Graph Neural Networks', *ICLR*, 2022, arXiv: 2102.07835 [cs.LG]

How to represent graphs?



- ☆ Graph $G = \{V, E\}$
- ☆ Two graphs G and G' can have a *different* number of vertices.
- ☆ Hence, we require a *vectorised representation* $f: \mathcal{G} \rightarrow \mathbb{R}^d$ of graphs.
- ☆ Such a representation f needs to be *permutation-invariant*.

Graph neural networks in a nutshell

- ☆ Learn node representations h_v based on aggregated attributes a_v .
- ☆ Aggregate them over neighbourhoods.
- ☆ Iteration k contains information up to k hops away.
- ☆ Repeat procedure K times.

$$a_v^{(k)} := \text{aggregate}^{(k)}\left(\left\{h_u^{(k-1)} \mid u \in \mathcal{N}_G(v)\right\}\right)$$

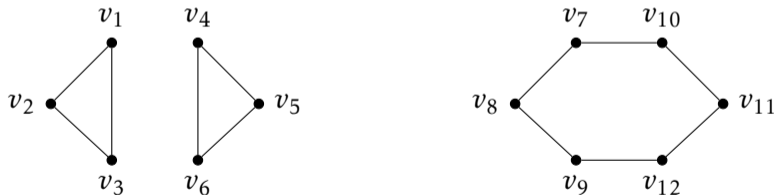
$$h_v^{(k)} := \text{combine}^{(k)}\left(h_v^{(k-1)}, a_v^{(k)}\right)$$

$$h_G := \text{readout}\left(\left\{h_v^{(K)} \mid v \in \mathcal{V}_G\right\}\right)$$

This terminology follows K. Xu, W. Hu, J. Leskovec and S. Jegelka, 'How Powerful are Graph Neural Networks?', *ICLR*, 2019.

Status quo

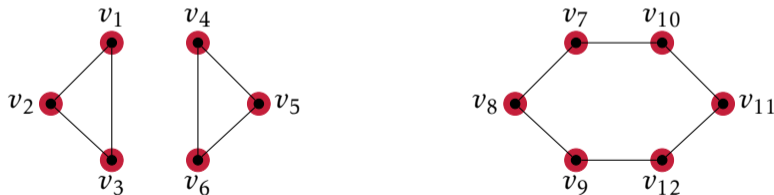
- ☆ GNNs are *at most* as expressive as the Weisfeiler–Lehman test for graph isomorphism, commonly abbreviated as $WL[1]$ ¹.
- ☆ Graphs are topological objects.
- ☆ But GNNs are *incapable* of recognising certain topological structures!



¹K. Xu, W. Hu, J. Leskovec and S. Jegelka, 'How Powerful are Graph Neural Networks?', *ICLR*, 2019.

Status quo

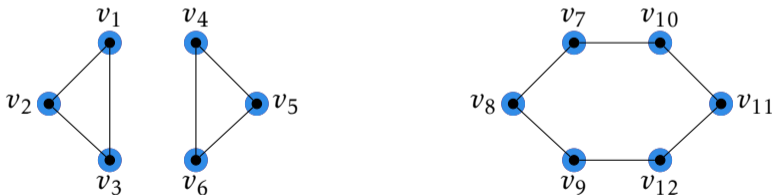
- ☆ GNNs are *at most* as expressive as the Weisfeiler–Lehman test for graph isomorphism, commonly abbreviated as $WL[1]$ ¹.
- ☆ Graphs are topological objects.
- ☆ But GNNs are *incapable* of recognising certain topological structures!



¹K. Xu, W. Hu, J. Leskovec and S. Jegelka, 'How Powerful are Graph Neural Networks?', *ICLR*, 2019.

Status quo

- ☆ GNNs are *at most* as expressive as the Weisfeiler–Lehman test for graph isomorphism, commonly abbreviated as WL[1]¹.
- ☆ Graphs are topological objects.
- ☆ But GNNs are *incapable* of recognising certain topological structures!
- ☆ **What can we gain when imbuing them with knowledge about the topology?**



¹K. Xu, W. Hu, J. Leskovec and S. Jegelka, 'How Powerful are Graph Neural Networks?', *ICLR*, 2019.

Graphs as simplicial complexes

$G = \{V, E\}$ as a simplicial complex K

☆ Let $K_k = \{\sigma \in K : \dim(\sigma) = k\}$, and $\partial_k : K_k \rightarrow C_{k-1}(K) \quad \sigma \rightarrow \sum_{\substack{\tau: \tau \subset \sigma \\ \dim(\tau) = k-1}} \tau$

☆ Topological features as the rank of the homology groups $H_k = \ker \partial_k / \text{im} \partial_{k+1}$

$$K_0 = V, \quad K_1 = E$$

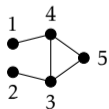
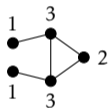
$\rightarrow \text{rank}(H_0) = \beta_0$: number of connected components and $\text{rank}(H_1) = \beta_1$: number of edges.

The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.

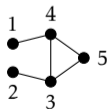
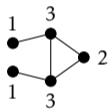


The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.

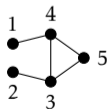
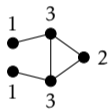


The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.

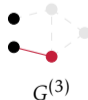
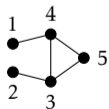
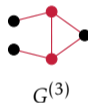
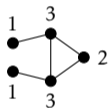


The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.

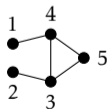
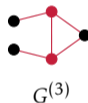
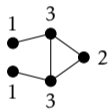


The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.

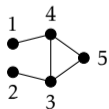
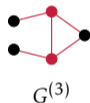
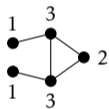


The expressivity of a filtration

Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ with $f^{(1)} < \dots < f^{(n)}$.

Filtration $\emptyset = G^{(0)} \subseteq G^{(1)} \subseteq \dots \subseteq G^{(n)} = G = (V, E)$

$V^{(i)} := \{v \in V \mid f(x^{(v)}) \leq f^{(i)}\}$, $E^{(i)} := \{v, w \in E \mid \max\{f(x^{(v)}), f(x^{(w)})\} \leq f^{(i)}\}$.



Persistence Diagrams

$$\emptyset \subset G^{(0)} \subset G^{(1)} \subset \dots \subset G^{(N)} \subset G$$

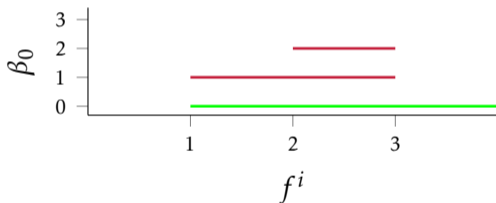
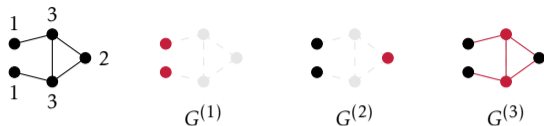
Let $\beta_k^{i,j}$ the k -th *persistent* Betti number, that is the number of topological features of dimension k that *persist* from $G^{(i)}$ to $G^{(j)}$.

- ☆ $k = 0$: number of connected components
- ☆ $k = 1$: number of cycles

We can build a *persistence diagram* by storing each point (f_i, f_j) with multiplicity

$$\mu_k^{i,j} = (\beta_k^{i,j-1} - \beta_k^{i,j}) - (\beta_k^{i-1,j-1} - \beta_k^{i-1,j}) \quad (1)$$

Persistence Diagrams

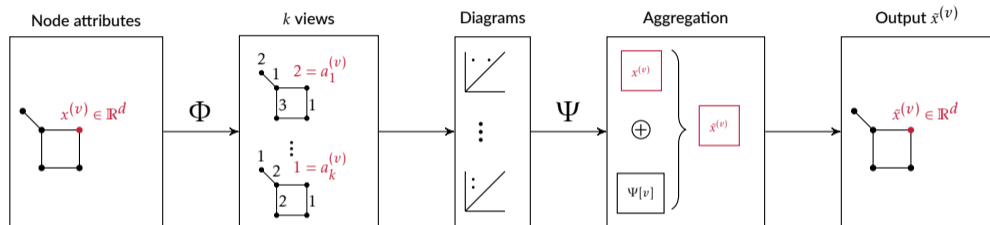


We know how to *learn* a filtration², can we create a **layer** that neatly integrates with arbitrary GNNs?

²C. D. Hofer, F. Graf, B. Rieck, M. Niethammer and R. Kwitt, 'Graph Filtration Learning', *ICML*, 2020, pp. 4314–4323.

Topological graph neural networks

Overview



- ☆ Use a node map $\Phi: \mathbb{R}^d \rightarrow \mathbb{R}^k$ to create k different filtrations of the graph.
- ☆ Use a coordinatisation function Ψ to create *compatible* representations of the node attributes.

Expressivity of a GNN

Typical GNN architectures are *no more expressive* than the Weisfeiler–Lehman test.

Theorem

Persistent homology is *at least* as expressive as WL[1], i.e. if the WL[1] label sequences for two graphs G and G' diverge, there exists an injective filtration f such that the corresponding persistence diagrams \mathcal{D}_0 and \mathcal{D}'_0 are not equal.

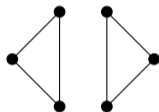
Proof sketch.

We first show how to construct an appropriate filtration function f from a WL[1] label sequence. Since f is not necessarily injective, we show that there is an injective function \tilde{f} that is arbitrarily close to f and whose corresponding persistence diagrams $\widetilde{\mathcal{D}}_0, \widetilde{\mathcal{D}}'_0$ do not coincide. ■

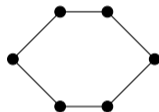
Expressivity of a GNN

There's more!

There are non-isomorphic graphs that $WL[1]$ **cannot distinguish**, but persistent homology can:



G



G'

We have $\beta_0(G) = \beta_1(G) = 2$, because G consists of two connected components and two cycles, whereas $\beta_0(G') = \beta_1(G') = 1$ as G' only consists of one connected component and one cycle.

Experiments

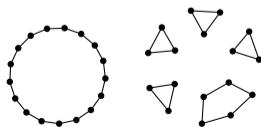
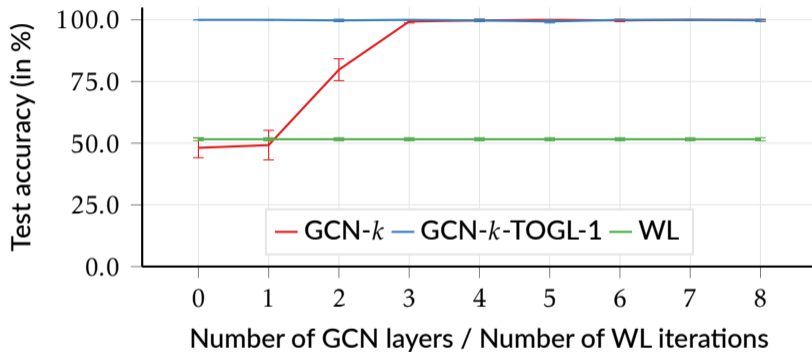
- ☆ Take GCN architecture with 4 convolutional layers (GCN-4).
- ☆ Replace second layer by TOGL.

Plan

- 1** Assess expressivity on synthetic data sets.
- 2** Assess predictive performance on data sets without node features.
- 3** Assess predictive performance on benchmark data sets.

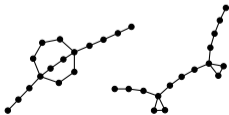
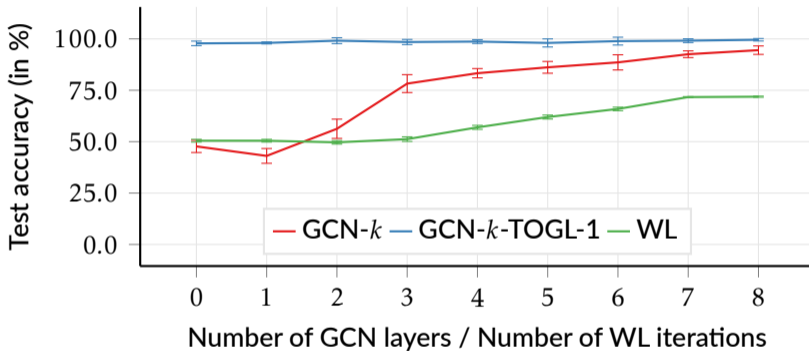
Expressivity

Cycles data set



Expressivity

Necklaces data set



Classifying graphs/nodes based on structural features alone

Existing data sets tend to 'leak' information into node attributes, thus decreasing the utility of topological features. Hence, we replaced all node features by random ones.

Method	Graph classification				Node classification		
	DD	ENZYMES	MNIST	PROTEINS	Pattern		
GCN-4	68.0±3.6	22.0±3.3	76.2± 0.5	68.8± 2.8	85.5	±	0.4
GCN-3-TOGL-1	75.1±2.1	30.3±6.5	84.8± 0.4	73.8± 4.3	86.6	±	0.1
GIN-4	75.6±2.8	21.3±6.5	83.4± 0.9	74.6± 3.1	84.8	±	0.0
GIN-3-TOGL-1	76.2±2.4	23.7±6.9	84.4± 1.1	73.9± 4.9	86.7	±	0.1
GAT-4	63.3±3.7	21.7±2.9	63.2±10.4	67.5± 2.6	73.1	±	1.9
GAT-3-TOGL-1	75.7±2.1	23.5±6.1	77.2±10.5	72.4± 4.6	59.6	±	3.3

Classifying benchmark data sets

While we improve baseline classification performance, the best performance is *not* driven by the availability of topological structures!

Method	Graph classification							Node classification
	CIFAR-10	DD	ENZYMES	MNIST	PROTEINS-full	IMDB-B	REDDIT-B	CLUSTER
GATED-GCN-4	67.3 ± 0.3	72.9 ± 2.1	65.7 ± 4.9	97.3 ± 0.1	76.4 ± 2.9	—	—	60.4 ± 0.4
WL	—	77.7 ± 2.0	54.3 ± 0.9	—	73.1 ± 0.5	71.2 ± 0.5	78.0 ± 0.6	—
WL-OA	—	77.8 ± 1.2	58.9 ± 0.9	—	73.5 ± 0.9	74.0 ± 0.7	87.6 ± 0.3	—
GCN-4	54.2 ± 1.5	72.8 ± 4.1	65.8 ± 4.6	90.0 ± 0.3	76.1 ± 2.4	68.6 ± 4.9	92.8 ± 1.7	57.0 ± 0.9
GCN-3-TOGL-1	61.7 ± 1.0	73.2 ± 4.7	53.0 ± 9.2	95.5 ± 0.2	76.0 ± 3.9	72.0 ± 2.3	89.4 ± 2.2	60.4 ± 0.2
GIN-4	54.8 ± 1.4	70.8 ± 3.8	50.0 ± 12.3	96.1 ± 0.3	72.3 ± 3.3	72.8 ± 2.5	81.7 ± 6.9	58.5 ± 0.1
GIN-3-TOGL-1	61.3 ± 0.4	75.2 ± 4.2	43.8 ± 7.9	96.1 ± 0.1	73.6 ± 4.8	74.2 ± 4.2	89.7 ± 2.5	60.4 ± 0.2
GAT-4	57.4 ± 0.6	71.1 ± 3.1	26.8 ± 4.1	94.1 ± 0.3	71.3 ± 5.4	73.2 ± 4.1	44.2 ± 6.6	56.6 ± 0.4
GAT-3-TOGL-1	63.9 ± 1.2	73.7 ± 2.9	51.5 ± 7.3	95.9 ± 0.3	75.2 ± 3.9	70.8 ± 8.0	89.5 ± 8.7	58.4 ± 3.7

What did we learn ?

- ☆ When topological structure is important, the addition of a topology-aware layer boosts performance.
- ☆ The position of the topology layer is an hyper-parameter and does impact performance.
- ☆ Should we incorporate higher-order structures (such as cliques) ?
- ☆ What do we gain from learning a filtration function (compared to fixed filtration) ?

Thank you !



Max Horn

🐦 @ExpectationMax

arxiv/2102.07835



Edward De Brouwer

🐦 @EdwardOnBrew



Michael Moor

🐦 @Michael_D_Moor



Yves Moreau



Bastian Rieck

🐦 @Pseudomanifold



Karsten Borgwardt

🐦 @kmborgwardt