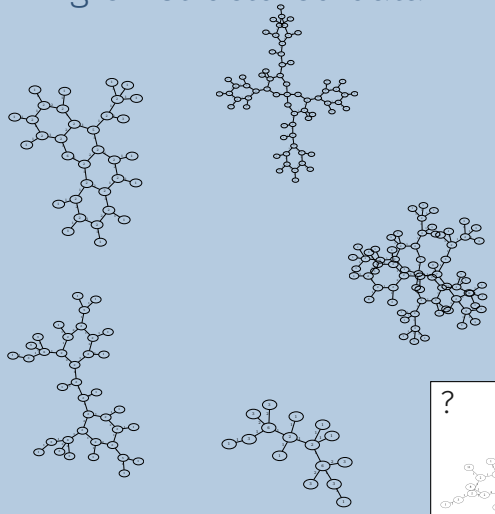
Expressive Graph Embeddings via Homomorphism Counts

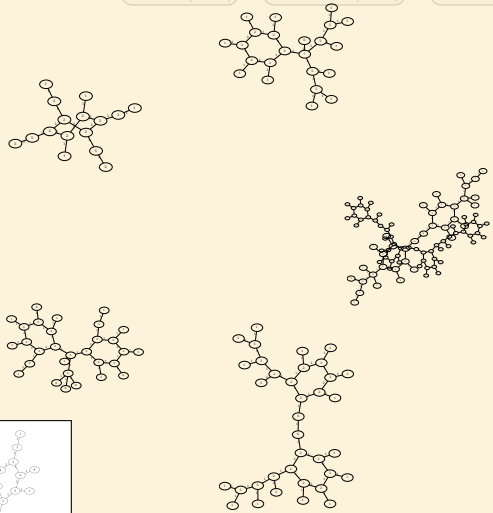
Pascal Welke

CAIML Seminar on 25. November 2024

Learning on structured data



Hu et al (2020) Morris et al (2020) Dwivedi et al (2022)

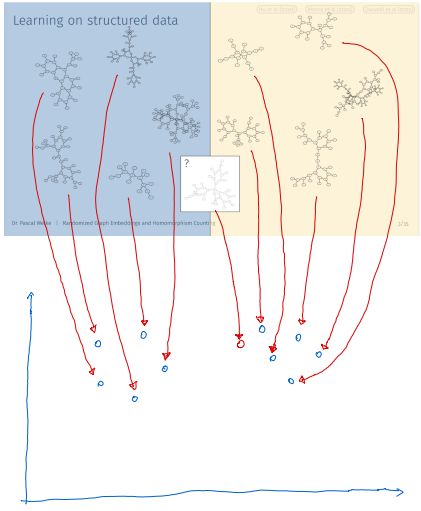


Graph Representation Learning

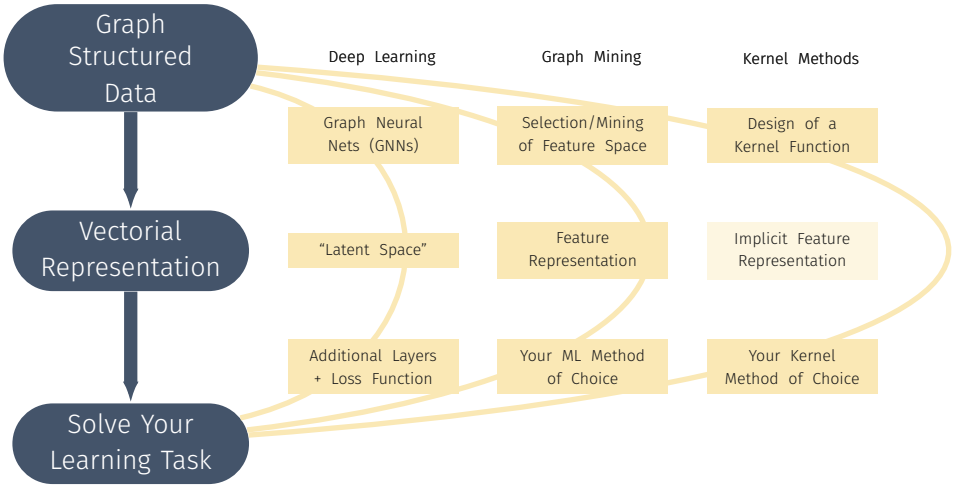
The goal

Vectorial graph representations that

- yield semantically and structurally meaningful distances
- are interpretable
- are adaptable to given data



Graph representation learning



The problem with vectorial graph representations

We want our graph representation function ϕ to be

- permutation-invariant

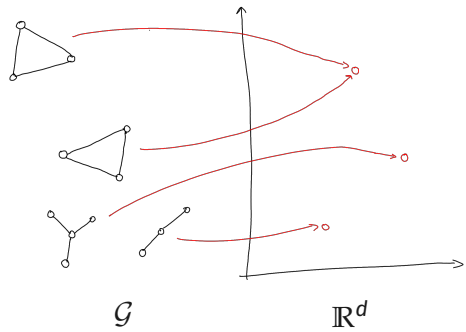
  for all isomorphic graphs

  $$G \simeq H : \ \phi(G) = \phi(H)$$

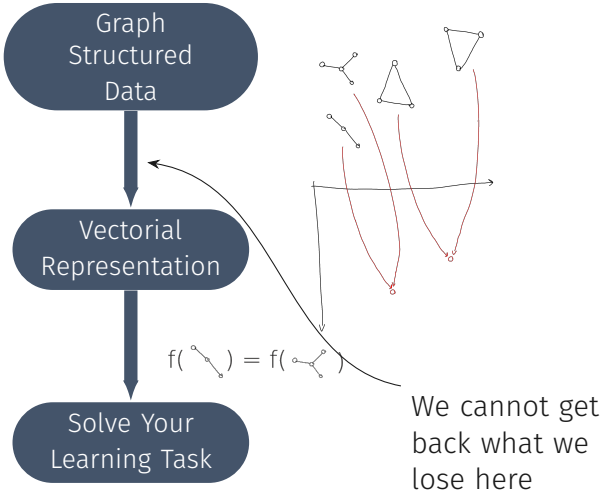
- complete

  for all non-isomorphic graphs

  $$G \not\simeq H : \ \phi(G) \neq \phi(H)$$



Why do we care?



- Unfortunately computing any permutation invariant and complete embedding (or kernel) is as hard as deciding graph isomorphism
- **Typical solution**: drop completeness for efficiency
  - most practical graph kernels, GNNs, Weisfeiler Leman test, …

Homomorphism Counts as Graph Representations

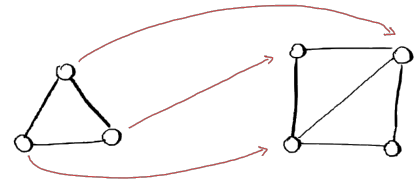
Homomorphism

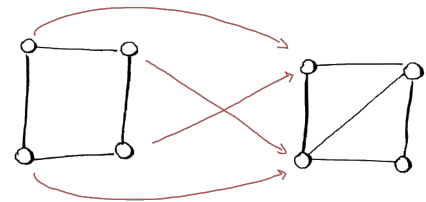
A *homomorphism* from H to G is a function

$$h : V(H) \to V(G)$$

such that

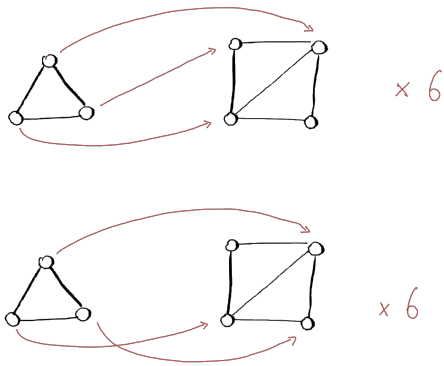
$$(v, w) \in E(H) \implies (h(v), h(w)) \in E(G)$$





Counting Homomorphisms

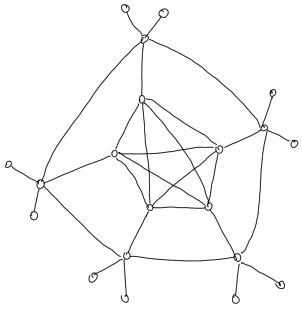
Given H and G, we can ask *how many* homomorphisms exist from H to G?



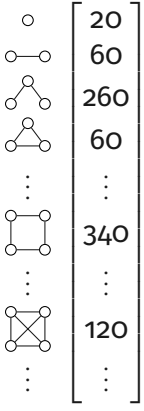
There are twelve homomorphisms from H to G!

An intractable complete graph embedding

G



$\varphi_n(G)$



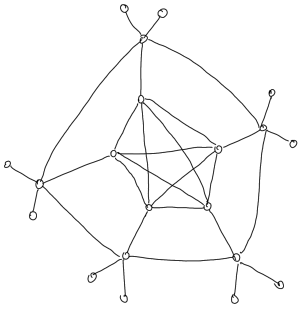
Theorem [Lovász 1967].
Two graphs G and H are isomorphic iff
$\varphi_n(G) = \varphi_n(H)$

We can count homomorphisms (for some graphs) in practice!

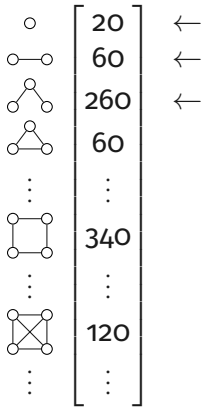
- Homomorphism counting is fixed parameter tractable
- The parameter is called tree-width
- If the pattern H has tree-width k, the homomorphisms from H to any G can be counted in $O(|V(G)|^k)$

An intractable graph embedding

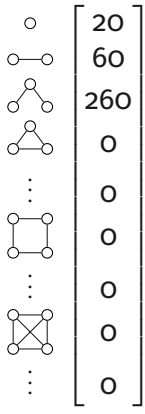
G



$\varphi_n(G)$



$\varphi_{\mathcal{F}}(G)$



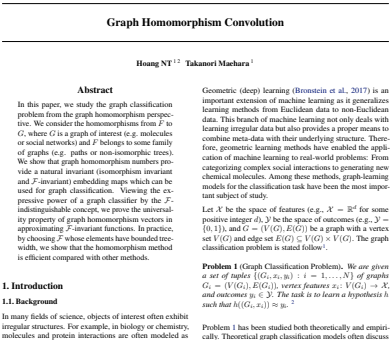
How to select the patterns?

- Some patterns are more expensive than others
- Some patterns might be more useful for the task at hand than others

We will now see two variants how to select patterns

Graph Homomorphism Convolution (GHC) NT and Maehara (2020)

- Introduce homomorphism counts as feature vectors of graphs
- Propose to select 'suitable, small' pattern set $\mathcal{F}$
  - The first 13 trees
  - Cycles up to length 7
- Use an SVM with these features



Graph Homomorphism Convolution

Hoang NT [1 2] **Takanori Maehara** [1]

Abstract

In this paper, we study the graph classification problem from the graph homomorphism perspective. We consider the homomorphisms from F to G, where G is a graph of interest (e.g. molecules or social networks) and F belongs to some family of graphs (e.g. paths or non-isomorphic trees). We show that graph homomorphism numbers provide a natural invariant (isomorphism invariant and $\mathcal{F}$-invariant) embedding maps which can be used for graph classification. Viewing the expressive power of a graph classifier by the $\mathcal{F}$-indistinguishable concept, we prove the universality property of graph homomorphism vectors in approximating $\mathcal{F}$-invariant functions. In practice, by choosing $\mathcal{F}$ whose elements have bounded tree-width, we show that the homomorphism method is efficient compared with other methods.

1. Introduction

1.1. Background

In many fields of science, objects of interest often exhibit irregular structures. For example, in biology or chemistry, molecules and protein interactions are often modeled as

Geometric (deep) learning (Bronstein et al., 2017) is an important extension of machine learning as it generalizes learning methods from Euclidean data to non-Euclidean data. This branch of machine learning not only deals with learning irregular data but also provides a proper means to combine meta-data with their underlying structure. Therefore, geometric learning methods have enabled the application of machine learning to real-world problems: From categorizing complex social interactions to generating new chemical molecules. Among these methods, graph-learning models for the classification task have been the most important subject of study.

Let $\mathcal{X}$ be the space of features (e.g., $\mathcal{X} = \mathbb{R}^d$ for some positive integer d), $\mathcal{Y}$ be the space of outcomes (e.g., $\mathcal{Y} = \{0, 1\}$), and $G = (V(G), E(G))$ be a graph with a vertex set $V(G)$ and edge set $E(G) \subseteq V(G) \times V(G)$. The graph classification problem is stated follow[3].

Problem 1 (Graph Classification Problem)**.** *We are given a set of tuples* $\{(G_i, x_i, y_i) : i = 1, \ldots, N\}$ *of graphs* $G_i = (V(G_i), E(G_i))$*, vertex features* $x_i : V(G_i) \to \mathcal{X}$*, and outcomes* $y_i \in \mathcal{Y}$*. The task is to learn a hypothesis* h *such that* $h((G_i, x_i)) \approx y_i$*.* [2]

Problem 1 has been studied both theoretically and empirically. Theoretical graph classification models often discuss

GHC: Experimental results

Table 2. Classification accuracy over 10 experiments

(a) Synthetic datasets

METHODS	CSL	BIPARTITE	PAULUS25
Practical models			
GIN	10.00 ± 0.00	55.75 ± 7.91	7.14 ± 0.00
GNTK	10.00 ± 0.00	58.03 ± 6.84	7.14 ± 0.00
Theory models			
Ring-GNN	$10 \sim 80 \pm 15.7$	55.72 ± 6.95	7.15 ± 0.00
GHC-Tree	10.00 ± 0.00	52.68 ± 7.15	7.14 ± 0.00
GHC-Cycle	$\mathbf{100.0 \pm 0.00}$	$\mathbf{100.0 \pm 0.00}$	7.14 ± 0.00

(b) Benchmark datasets

METHODS	MUTAG	IMDB-BIN	IMDB-MUL
Practical models			
GNTK	89.46 ± 7.03	75.61 ± 3.98	51.91 ± 3.56
GIN	89.40 ± 5.60	70.70 ± 1.10	43.20 ± 2.00
PATCHY-SAN	89.92 ± 4.50	71.00 ± 2.20	45.20 ± 2.80
WL kernel	90.40 ± 5.70	73.80 ± 3.90	50.90 ± 3.80
Theory models			
Ring-GNN	78.07 ± 5.61	73.00 ± 5.40	48.20 ± 2.70
GHC-Tree	89.28 ± 8.26	72.10 ± 2.62	48.60 ± 4.40
GHC-Cycles	87.81 ± 7.46	70.93 ± 4.54	47.41 ± 3.67

- Good results on some synthetic datasets
- Competitive results on (smaller) benchmark datasets

GHC is incomplete

- GHC in practice requires a fixed, user defined choice of the pattern set $\mathcal{F}$
- This allows to bound the expressivity of GHC by an extension of the WL algorithm: k-WL (Neuen (2024))

Expectation-Complete Graph Representations with Homomorphisms




ICML 2023

Pascal Welke*, Maximilian Thiessen*, Fabian Jogl, and Thomas Gärtner

TU Wien
Vienna | Austria
Research Unit Machine Learning

At a glance

- Expressiveness bounded by k-WL
  - GHC
  - MPNNs
  - 'higher-order' GNNs

$\Rightarrow$ choice of architecture implies a fixed limit on what graphs can be distinguished

- What can we do if we don't know anything about our datset?

- We present an architecture which has no upper expressivity bound
- Asymptotically, our graph representation is complete.

$\Rightarrow$ allows to adapt to challenging learning tasks without domain knowledge

$\Rightarrow$ works well in practice

What if we keep completeness …

… in expectation?

Expectation complete graph embeddings

Let $\phi_X : \mathcal{G} \to V$ depend on a random variable X drawn from a distr. $\mathcal{D}$ over a set $\mathcal{X}$

We call ϕ_X complete in expectation if the expectation

$$\mathop{\mathbb{E}}_{X \sim \mathcal{D}}[\phi_X(\cdot)] = \sum_{t \in \mathcal{X}} \Pr(X = t)\phi_t(\cdot)$$

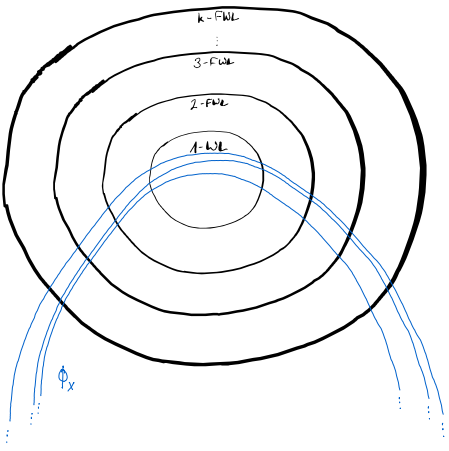
is a complete graph embedding

What is the benefit?

Sampling $X_1, X_2, X_3, \ldots$ will eventually make the joint embedding

$$(\phi_{X_1}(G), \phi_{X_2}(G), \phi_{X_3}(G), \ldots)$$

arbitrarily expressive



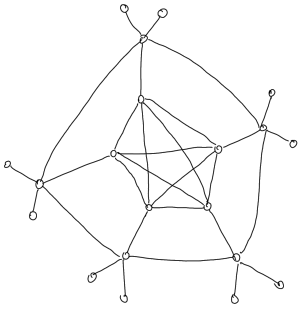
What if we keep completeness …

… in expectation

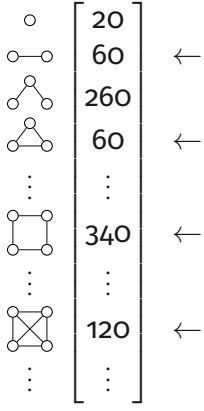
… efficiently

An intractable expectation complete graph embedding

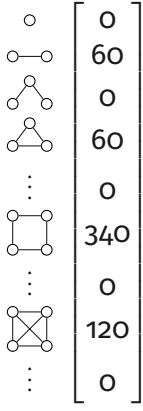
G



$\varphi_n(G)$



$\varphi_{\mathcal{F}}(G)$



Efficient and expectation-complete graph embeddings

- Homomorphism counting is fixed parameter tractable
- We design a distribution $\mathcal{D}$ that weights down expensive patterns

Theorem (ICML 2023)

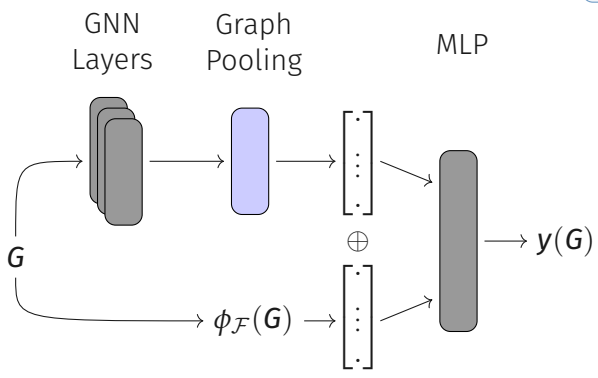
Computing the expectation-complete graph embedding $\phi_X(G)$ with $X \sim \mathcal{D}$ takes polynomial time in $V(G)$ in expectation for all $G \in \mathcal{G}_n$.

- We also showed
  - convergence results
  - universal approximation results

Efficient and expectation-complete GNNs

We can make any (message passing) GNN expectation-complete

ICML 2023
LoG 2023



Empirical results

Table 1. Performance of different GNNs on 9 OGB benchmarks and ZINC. Baseline of a GNN with homorphism counts is the same GNN without homomorphism counts. Results for GNNs with homorphism counts are averaged over 9 different random samples of pattern graphs.

	Top 1 / 2 / 3	Beats baseline
GIN	0% / 0% / 0%	-
GIN+hom	0% / 10% / 10%	100%
GCN	0% / 0% / 0%	-
GCN+hom	10% / 10% / 20%	90%
GIN+F	0% / 10% / 50%	-
GIN+hom +F	20% / 40% / 70%	90%
GCN+F	0% / 50% / 60%	-
GCN+hom+F	70% / 80% / 90%	90%

Table 2. Accuracy on synthetic data

Method	CSL	PAULUS25
GIN	10.00 ± 0.00	7.14 ± 0.00
GNTK	10.00 ± 0.00	7.14 ± 0.00
GHC-Tree	10.00 ± 0.00	7.14 ± 0.00
GHC-Cycle	100.0 ± 0.00	7.14 ± 0.00
WL	10.00 ± 0.00	7.14 ± 0.00
Ours	37.67 ± 9.11	100.0 ± 0.00

An open question and a recent answer

- Our runtime is polynomial in expectation, but
  - We can realistically sample 20-100 patterns
  - (that suffices in practice)
- How can we speedup the runtime while maintaining the theoretical properties?

Estimating homomorphism counts instead of exact computation might work well

- Beaujean et al (2021)
- BSc thesis 2023
- KDD 2020
- fast and precise in practice

Homomorphism Counts as Node Representations

Connecting homomorphism counting and message passing

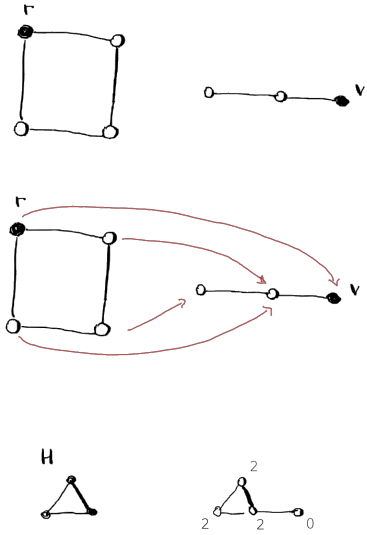
- So far, message passing and homomorphism counting have touched, but not really interacted
- Homomorphism counts can also be included in the message passing

Rooted homomorphism counting

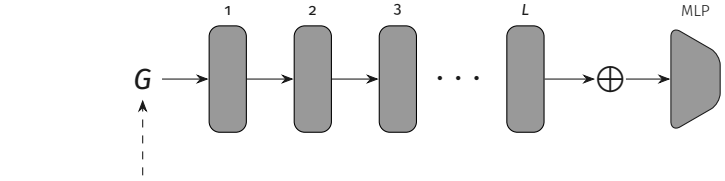
A *rooted graph* (G, v) is a graph G with a special root $v \in V(G)$

A rooted homomorphism h from (H, r) to (G, v) is a homomorphism h with $h(r) = v$

We can now count rooted homomorphisms for any node v in G



GNNs with Local Graph Parameters ($\mathcal{F}$-MPNNs) Barceló et al (2021)



- This architecture is more expressive than WL
- It is incomparable to 2-WL
- Can be bounded by $\mathcal{F}$-WL (!)

Graph Neural Networks with Local Graph Parameters

Pablo Barceló[1,2], Floris Geerts[3], Juan Reutter[1,2], Maksimilian Ryschkov[3]
[1] Department of Computer Science, PUC, Chile
[2] Millennium Institute for Foundational Research on Data, Chile
[3] Department of Computer Science, University of Antwerp, Belgium
[pbarcelo,jreutter]@ing.puc.cl, [floris.geerts,maksimilian.ryschkov]@uantwerpen.be

Abstract

Various recent proposals increase the distinguishing power of Graph Neural Networks (GNNs) by propagating features between k-tuples of vertices. The distinguishing power of these "higher-order" GNNs is known to be bounded by the k-dimensional Weisfeiler-Leman (WL) test, yet their $O(n^k)$ memory requirements limit their applicability. Other proposals infuse GNNs with local higher-order graph structural information from the start, hereby inheriting the desirable $O(n)$ memory requirement from GNNs at the cost of a one-time, possibly non-linear, preprocessing step. We propose local graph parameter enabled GNNs as a framework for studying the latter kind of approaches. We precisely characterize their distinguishing power, in terms of a variant of the WL test, and in terms of the graph structural properties that they can take into account. Local graph parameters can be added to any GNN architecture, and are cheap to compute. In terms of expressive power, our proposal lies in the middle of GNNs and their higher-order counterparts. Further, we propose several techniques to aid in choosing the right local graph parameters. Our results connect GNNs with deep results in finite model theory and finite variable logics. Our experimental evaluation shows that adding local graph parameters often has a positive effect on a variety of GNNs, datasets and graph learning tasks.

Experimental Results

(a) Results for the ZINC dataset show that homomorphism (hom) counts of cycles improve every model. We compare the mean absolute error (MAE) of each model without any homomorphism count (baseline), against the model augmented with the hom count, and with subgraph isomorphism (iso) counts of C_3–C_{10}.

MODEL	MAE (BASE)	MAE (HOM)	MAE (ISO)
GAT	0.47±0.02	**0.22±0.01**	0.24±0.01
GCN	0.35±0.01	**0.20±0.01**	0.22±0.01
GraphSage	0.44±0.01	**0.24±0.01**	0.24±0.01
MoNet	0.25±0.01	0.19±0.01	**0.16±0.01**
GatedGCN	0.34±0.05	**0.1353±0.01**	0.1357±0.01

(b) The effect of different cycles for the GAT model over the ZINC dataset, using mean absolute error.

SET ($\mathcal{F}$)	MAE
NONE	0.47±0.02
$\{C_3\}$	0.45±0.01
$\{C_4\}$	0.34±0.02
$\{C_6\}$	0.31±0.01
$\{C_5, C_6\}$	0.28±0.01
$\{C_3, \ldots, C_6\}$	0.23±0.01
$\{C_3, \ldots, C_{10}\}$	**0.22±0.01**

Table 2: Results for the PATTERN dataset show that homomorphism counts improve all models except GatedGCN. We compare weighted accuracy of each model without any homomorphism count (baseline) against the model augmented with the counts of the set $\mathcal{F}$ that showed best performance (best $\mathcal{F}$).

MODEL + BEST $\mathcal{F}$	ACCURACY BASELINE	ACCURACY BEST
GAT $\{K_3, K_4, K_5\}$	78.83 ± 0.60	85.50 ± 0.23
GCN $\{K_3, K_4, K_5\}$	71.42 ± 1,38	82.49 ± 0.48
GraphSage $\{K_3, K_4, K_5\}$	70.78 ± 0,19	85.85 ± 0.15
MoNet $\{K_3, K_4, K_5\}$	85.90 ± 0.03	**86.63 ± 0.03**
GatedGCN $\{\emptyset\}$	86.15 ± 0.08	86.15 ± 0.08

Table 3: All models improve the Hits@50 metric over the COLLAB dataset. We compare each model without any homomorphism count (baseline) against the model augmented with the counts of the set of patterns that showed best performance (best $\mathcal{F}$).

MODEL + BEST $\mathcal{F}$	HITS@50 BASELINE	HITS@50 BEST
GAT $\{K_3\}$	50.32±0.55	52.87±0.87
GCN $\{K_3, K_4, K_5\}$	51.35±1.30	**54.60±1.01**
GraphSage $\{K_3\}$	50.33±0.68	51.39±1.23
MoNet $\{K_4\}$	49.81±1.56	51.76±1.38
GatedGCN $\{K_3\}$	51.00±2.54	51.57±0.68

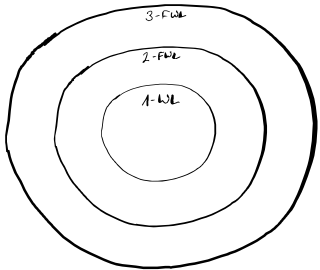
Insights?

- By adding homcounts to the node labels before message passing, we get an architecture that is at least as expressive as message passing
- Cycle counting seems to be important ;)

GNNs can Count Homomorphisms – Implicitly

Practical problem

- 1-WL is sometimes not expressive enough
- In particular, it is insensitive to the number of cycles
- 2-FWL is already impractical



Weisfeiler and Leman Go Loopy: A New Hierarchy for Graph Representational Learning

NeurIPS 2024 (oral)

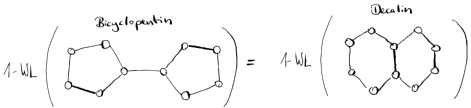
Raffaele Paolino*, Sohir Maskey*, Pascal Welke, and Gitta Kutyniok

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

TU WIEN

At a glance

- Property prediction for small molecules is one main application area of GNNs
- Number and type of cycles in molecules is important
- But

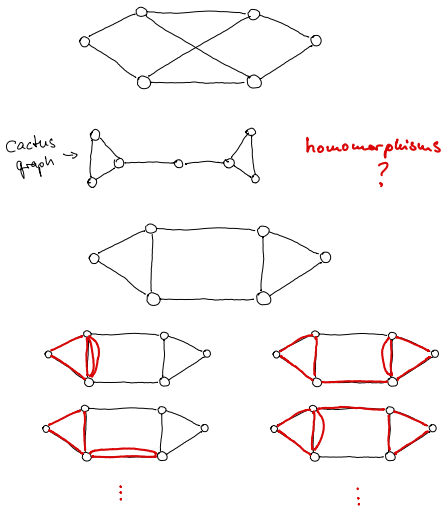


- we propose a generalized message passing architecture
- it can distinguish graphs with different r-cycle counts
- it can homomorphism-count all r-cactus graphs (strictly more expressive than 1-WL)
- fast in practice, s.o.t.a. results

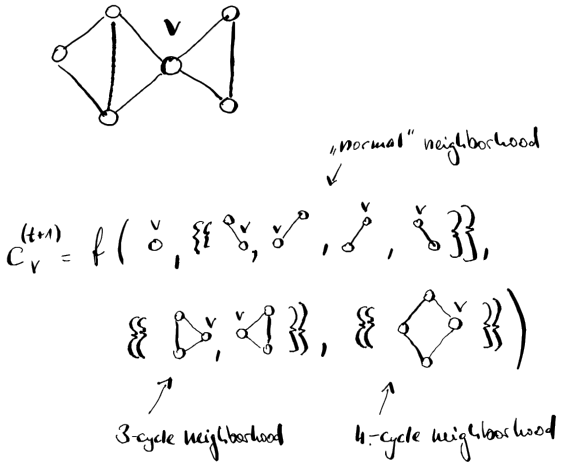
Contributions

A novel GNN architecture that is parametrized by cycle length r that

- is efficient on sparse graphs
- can subgraph count all cycles of length up to r
- can homomorphism count all r-cactus graphs



A glimpse at the implementation



- Generalized message passing over multiple sets of local "neighborhoods"
- Cycles can be enumerated quickly on many sparse graphs
  Horváth et al (2004)
- Cycle representations can be computed with GINs

A complete representation for cycles:

Empirical results

Table 4: Normalized test MAE (↓) on graph regression, QM9 dataset. Top three models as 1st, 2nd, 3rd.

Model	μ	α	$\varepsilon_{\text{homo}}$
1-GNN	0.493	0.78	0.00321
1-2-3-GNN	0.476	0.27	0.00337
DTNN	0.244	0.95	0.00388
Deep LRP	0.364	0.298	0.00254
PPGN	0.231	0.382	0.00276
NestedGNN	0.428	0.290	0.00265
I2-GNN	0.428	0.230	0.00261
DRFWL GNN	0.346	0.222	0.00226
5-ℓGIN	0.350 ± 0.011	0.217 ± 0.025	0.00205 ± 0.00005

Table 3: Test MAE (↓) on graph regression, ZINC dataset. Top three models as 1st, 2nd, 3rd.

Model	ZINC12K	ZINC250K
GIN	0.163 ± 0.004	0.088 ± 0.002
GCN	0.321 ± 0.009	-
GAT	0.384 ± 0.007	-
GSN	0.115 ± 0.012	-
CIN	0.079 ± 0.006	0.022 ± 0.002
NestedGNN	0.111 ± 0.003	0.029 ± 0.001
SUN	0.083 ± 0.003	-
GNNAK+	0.080 ± 0.001	-
I2-GNN	0.083 ± 0.001	0.023 ± 0.001
DRFWL GNN	0.077 ± 0.002	0.025 ± 0.003
SignNet	0.084 ± 0.004	0.024 ± 0.003
HIMP	0.151 ± 0.006	0.036 ± 0.002
PathNN	0.090 ± 0.004	-
5-ℓGIN	0.072 ± 0.002	0.022 ± 0.001

Open questions

We have seen different hierarchies of expressiveness

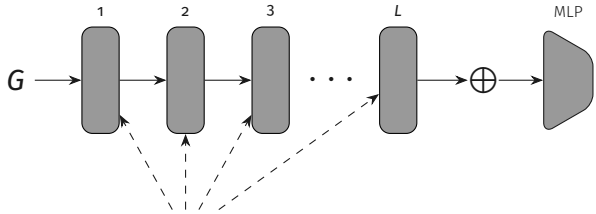
- increasing the size of $\mathcal{F}$ in NT and Maehara (2020)
- Barceló et al (2021) s $\mathcal{F}$-WL hierarchy
- the r-loopy WL test of NeurIPS 2024

How are they connected?

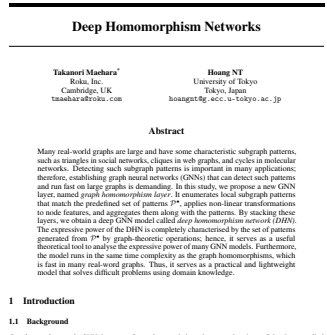
Can we collect most of our results in one architecture?

Deep Homomorphism Networks

Maehara and Hoang (2024)



- Message passing can be generalized to homomorphism counting
- We have to use a node-weighted variant of homomorphisms, though



Deep Homomorphism Networks

Takanori Maehara[*]
Roka, Inc.
Cambridge, UK
tmaehara@roka.com

Hoang NT
University of Tokyo
Tokyo, Japan
hoangnt@g.ecc.u-tokyo.ac.jp

Abstract

Many real-world graphs are large and have some characteristic subgraph patterns, such as triangles in social networks, cliques in web graphs, and cycles in molecular networks. Detecting such subgraph patterns is important in many applications; therefore, establishing graph neural networks (GNNs) that can detect such patterns and run fast on large graphs is demanding. In this study, we propose a new GNN layer, named *graph homomorphism layer*. It enumerates local subgraph patterns that match the predefined set of patterns $\mathcal{P}^*$, applies non-linear transformations to node features, and aggregates them along with the patterns. By stacking these layers, we obtain a deep GNN model called *deep homomorphism network (DHN)*. The expressive power of the DHN is completely characterised by the set of patterns generated from $\mathcal{P}^*$ by graph-theoretic operations; hence, it serves as a useful theoretical tool to analyse the expressive power of many GNN models. Furthermore, the model runs in the same time complexity as the graph homomorphisms, which is fast in many real-word graphs. Thus, it serves as a practical and lightweight model that solves difficult problems using domain knowledge.

1 Introduction

1.1 Background

Deep Homomorphism Network architecture

- Homomorphism counts can be weighted by the node weights
- Node weights can be computed by learnable functions
- Suitable pattern sets $\mathcal{P}$ allow to obtain architectures as powerful as our previous examples

$$\hom\left((F,\mu_F),(G,x)\right) = \sum_{\varphi \in \hom(F,G)} \prod_{p \in V(F)} \mu_F\left(x_{\varphi(p)}\right)$$

↑ elementwise

$$\mathrm{GHL}_{\mathcal{P}}\left((G,x);\ \rho,\ \{\mu_{p,\cdot} : \mathcal{P} \to \mathcal{P}^{\ell}\}\right) := \rho\left(\hom\left((P,\mu_{p,\cdot}),(G,x)\right) : P \in \mathcal{P}\right)$$

Putting everything together

- MPNN:
  - Multilayer DHN on the singleton and the edge
- GHC with patterns $\mathcal{F}$
  - A single layer DHN on $\mathcal{F}$
- $\mathcal{F}$-MPNNs
  - A single layer DHN on $\mathcal{F}$
  - Then multilayer DHN on the singleton and the edge
- r-loopy MPNNs
  - Multilayer DHN on singleton, edge, and cycles up to length r

Only issue is: It does not (yet) work well in practice

Concluding Remarks

Concluding Remarks

- Homomorphism-based methods work well in theory and practice
  NT and Maehara (2020) ICML 2023 NeurIPS 2024 Barceló et al (2021) Maehara and Hoang (2024)
- There is much more...
  - Intricate results linking homomorphism counting and the k-WL test
    Dell et al (2018) Neuen (2024) Lanzinger and Barceló (2024)
  - Characterizing expressivity of higher-order GNNs via homomorphism counts
    Zhang et al (2024)
  - Generalization bounds of GNNs using homomorphism counts
    Li et al (2024)
  - Homomorphism bases (aka spasms) of patterns allow to compute and learn(!) very powerful graph invariants
    Jin et al (2024) Dell et al (2018) Curticapean et al (2017)

References

Pablo Barceló, Floris Geerts, Juan L Reutter, Maksimilian Ryschkov (2021) Graph neural networks with local graph parameters. In: Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N Dauphin, Percy Liang, Jennifer Wortman Vaughan (eds) Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual, pp 25,280–25,293, URL `https://proceedings.neurips.cc/paper/2021/hash/d4d8d1ac7e00e9105775a6b660dd3cbb-Abstract.html`

Paul Beaujean, Florian Sikora, Florian Yger (2021) Graph homomorphism features: Why not sample? In: Machine Learning and Principles and Practice of Knowledge Discovery in Databases - International Workshops of ECML PKDD 2021, Virtual Event, September 13-17, 2021, Proceedings, Part I, Springer, Communications in Computer and Information Science, vol 1524, pp 216–222, DOI `10.1007/978-3-030-93736-2_17`, URL `https://doi.org/10.1007/978-3-030-93736-2_17`

Andrei Dragos Brasoveanu, Fabian Jogl, Pascal Welke, Maximilian Thiessen (2023) Extending graph neural networks with global features. In: Learning on Graphs Conference (LoG), URL `https://openreview.net/forum?id=aisVQy6R2k`

Radu Curticapean, Holger Dell, Dániel Marx (2017) Homomorphisms are a good basis for counting small subgraphs. In: Hamed Hatami, Pierre McKenzie, Valerie King (eds) Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017, ACM, pp 210–223, DOI `10.1145/3055399.3055502`, URL `https://doi.org/10.1145/3055399.3055502`

Holger Dell, Martin Grohe, Gaurav Rattan (2018) Lovász meets weisfeiler and leman. In: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, Donald Sannella (eds) 45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, LIPIcs, vol 107, pp 40:1–40:14, DOI `10.4230/LIPICS.ICALP.2018.40`, URL `https://doi.org/10.4230/LIPIcs.ICALP.2018.40`

Vijay Prakash Dwivedi, Ladislav Rampášek, Mikhail Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, Dominique Beaini (2022) Long range graph benchmark. In: Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, URL `https://openreview.net/forum?id=in7XC5RcjEn`

References

Tamás Horváth, Thomas Gärtner, Stefan Wrobel (2004) Cyclic pattern kernels for predictive graph mining. In: Won Kim, Ron Kohavi, Johannes Gehrke, William DuMouchel (eds) Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004, ACM, pp 158–167, DOI 10.1145/1014052.1014072

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, Jure Leskovec (2020) Open graph benchmark: Datasets for machine learning on graphs. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual, URL https://proceedings.neurips.cc/paper/2020/hash/fb60d411a5c5b72b2e7d3527cfc84fd0-Abstract.html

Emily Jin, Michael M Bronstein, Ismail Ilkan Ceylan, Matthias Lanzinger (2024) Homomorphism counts for graph neural networks: All about that basis. In: Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024, OpenReview.net, URL https://openreview.net/forum?id=zRrzSLwNHQ

Matthias Lanzinger, Pablo Barceló (2024) On the power of the weisfeiler-leman test for graph motif parameters. In: The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024, OpenReview.net, URL https://openreview.net/forum?id=FddFxi08J3

Shouheng Li, Floris Geerts, Dongwoo Kim, Qing Wang (2024) Towards bridging generalization and expressivity of graph neural networks. URL https://arxiv.org/abs/2410.10051, 2410.10051

Takanori Maehara, NT Hoang (2024) Deep homomorphism networks. In: The Thirty-eighth Annual Conference on Neural Information Processing Systems

Tobias Mette (2023) Hops for homomorphism count estimation. BSc Thesis, University of Bonn

References

Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, Marion Neumann (2020) Tudataset: A collection of benchmark datasets for learning with graphs. CoRR abs/2007.08663, URL https://arxiv.org/abs/2007.08663, 2007.08663

Daniel Neuen (2024) Homomorphism-distinguishing closedness for graphs of bounded tree-width. In: Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, Daniel Lokshtanov (eds) 41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, March 12-14, 2024, Clermont-Ferrand, France, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, LIPIcs, vol 289, pp 53:1–53:12, DOI 10.4230/LIPICS.STACS.2024.53, URL https://doi.org/10.4230/LIPIcs.STACS.2024.53

Hoang NT, Takanori Maehara (2020) Graph homomorphism convolution. In: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, PMLR, Proceedings of Machine Learning Research, vol 119, pp 7306–7316, URL http://proceedings.mlr.press/v119/nguyen20c.html

Raffaele Paolino*, Sohir Maskey*, Pascal Welke, Gitta Kutyniok (2024) Weisfeiler and leman go loopy: A new hierarchy for graph representational learning. 2403.13749

Pascal Welke, Florian Seiffarth, Michael Kamp, Stefan Wrobel (2020) HOPS: probabilistic subtree mining for small and large graphs. In: Rajesh Gupta, Yan Liu, Jiliang Tang, B Aditya Prakash (eds) SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), ACM, pp 1275–1284, DOI 10.1145/3394486.3403180, URL https://doi.org/10.1145/3394486.3403180

Pascal Welke*, Maximilian Thiessen*, Fabian Jogl, Thomas Gärtner (2023) Expectation-complete graph representations with homomorphisms. In: International Conference on Machine Learning (ICML), URL https://proceedings.mlr.press/v202/welke23a.html, 2306.05838

Bohang Zhang, Jingchu Gai, Yiheng Du, Qiwei Ye, Di He, Liwei Wang (2024) Beyond Weisfeiler-Lehman: A quantitative framework for GNN expressiveness. In: International Conference on Learning Representations (ICLR)