Expectation-Complete Graph Representations with Homomorphisms

Pascal Welke* [1,2] Maximilian Thiessen* [1] Fabian Jogl [1] Thomas Gärtner [1]

[1]TU Wien [2]University of Bonn

TL;DR

Through the power of random features we devise efficiently computable and expectation complete graph embeddings.

Expressiveness

Graph representation methods are compared to each other in terms of expressiveness. That is, their (theoretical) ability to compute different representations for pairs of non-isomorphic graphs.

For example, MPNNs are at most as expressive as the 1-WL isomorphism test.

High expressiveness is necessary for learning: If your method cannot distinguish two graphs, it cannot learn a function that behaves differently on these graphs.

Completeness

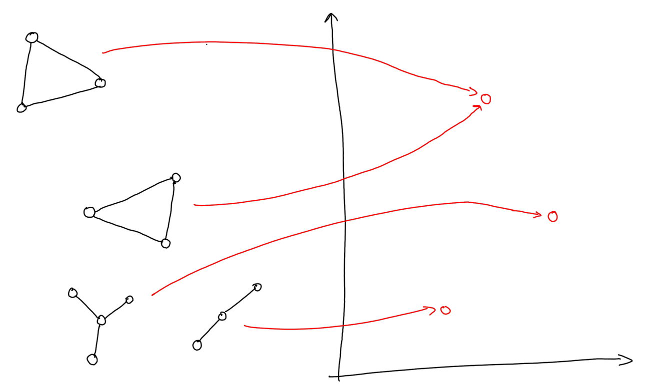
$\mathcal{G}$ the set of all graphs, V a vector space (e.g., $\mathbb{R}^d$)

A graph embedding $\varphi : \mathcal{G} \to V$ is permutation-invariant if for all isomorphic graphs

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

A permutation-invariant graph embedding φ is complete if for all non-isomorphic graphs

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



Originated from complete graph kernels [Gärtner et al., COLT 2003]

Problem

Why do we care about complete graph embeddings?

Allow us to learn/approximate any permutation-invariant function!

Unfortunately computing any such embedding is at least as hard as deciding graph isomorphism

- not known to be NP-hard and not known to be computable in polynomial-time

Typical solution: drop completeness for efficiency

- most practical graph kernels, GNNs, Weisfeiler Leman test, k-WL test, ...

Our solution: keep completeness in expectation!

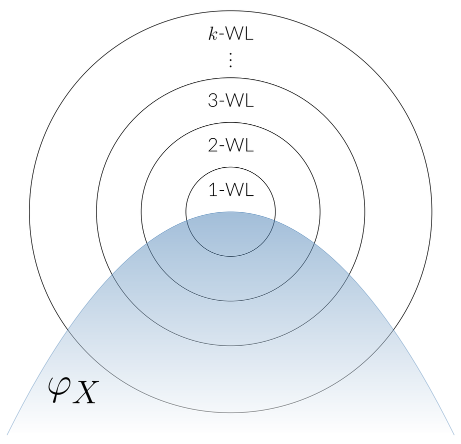
Complete in Expectation

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

We call φ_X complete in expectation if the expectation

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

is a complete graph embedding



What is the benefit?

Sampling $X_1, X_2, X_3, \dots$ will eventually make the joint embedding $(\varphi_{X_1}(G), \varphi_{X_2}(G), \varphi_{X_3}(G), \dots)$ arbitrarily expressive

Our Approach: Sampling from the Lovász Vector

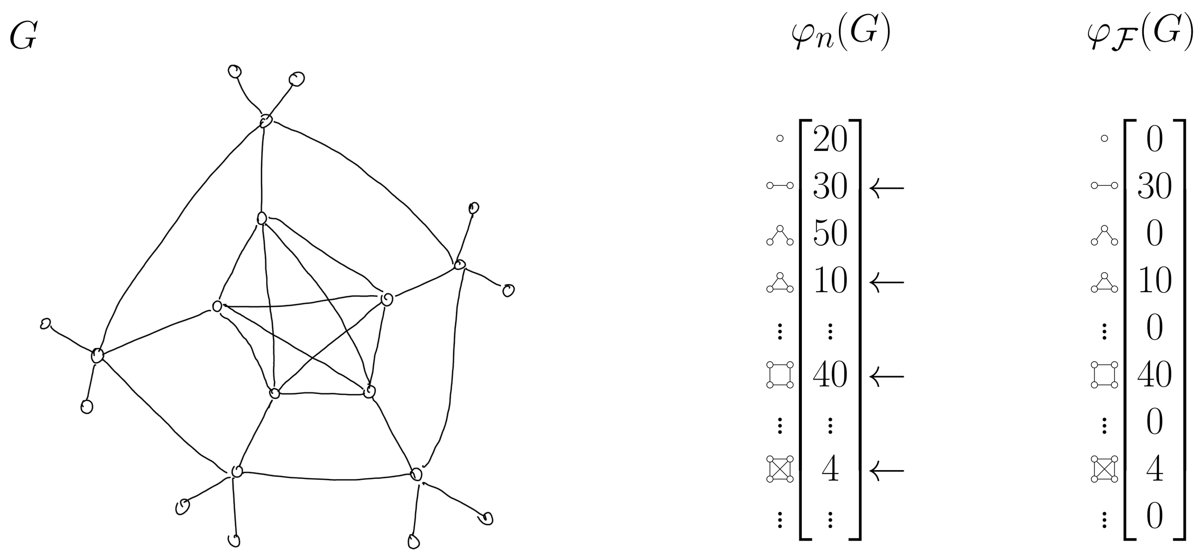
Let $\mathcal{G}_n$ be the set of all graphs with at most n vertices.

- the parameter n is typically the size of the largest graph in the sample.

Theorem. Let $\mathcal{D}$ be a distribution with full support on $\mathcal{G}_n$ and $G \in \mathcal{G}_n$. The graph embedding

$$\varphi_F(G) = \hom(F, G)e_F$$

with $F \sim \mathcal{D}$ is complete in expectation.



Proposed embedding: sample multiple pattern graphs F

- draw a finite sample $\mathcal{F}$ i.i.d from $\mathcal{D}$ and represent any graph $G \in \mathcal{G}_n$ by

$$\varphi_{\mathcal{F}}(G) = \sum_{F\in\mathcal{F}} \varphi_F(G)$$

- reduces the variance of the embedding
- currently $\ell = |\mathcal{F}|$ is a fixed hyperparameter (e.g., $\ell = 30$)

Efficient Sampling Scheme

Computing $\hom(F, G)$ is NP-hard in general.

If we take the treewidth of pattern F into account the runtime is [Díaz et al., 2002]:

$$\mathcal{O}\left(|V(F)||V(G)|^{\mathrm{tw}(F)+1}\right)$$

Idea: define distribution $\mathcal{D}$ on $\mathcal{G}_n$ s.t. runtime is polynomial in expectation!

Theorem. There exists a distribution $\mathcal{D}$ such that computing the expectation complete graph embedding $\varphi_F(G)$ takes polynomial time in $|V(G)|$ in expectation for all $G \in \mathcal{G}_n$.

General recipe:

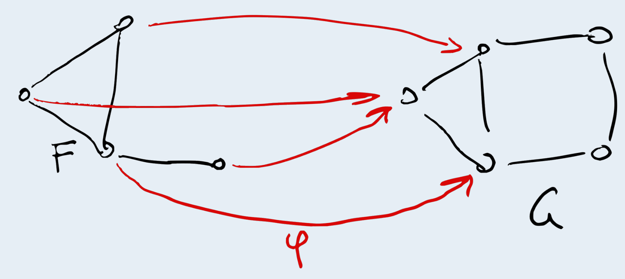
1. pick n as the maximum number of vertices in the training set
2. sample treewidth upper bound k
3. sample a maximal graph F' with treewidth k
4. take a random subgraph F of F'

E.g., $k \sim \mathrm{Poi}(\lambda)$ with $\lambda \leq \frac{1+d\log n}{n}$ guarantees runtime $\mathcal{O}\left(|V(G)|^{d+2}\right)$

Homomorphisms

Let F, G be graphs. A map $\varphi : V(F) \to V(G)$ is a graph homomorphism if φ preserves edges:

$$\{v, w\} \in E(F) \text{ implies } \{\varphi(v), \varphi(w)\} \in E(G)\,.$$



φ does not have to be injective (!)

$\hom(F, G)$: number of homomorphisms from F to G.

The Lovász Vector

Let $\varphi_n(G) = \hom(\mathcal{G}_n, G) = (\hom(F, G))_{F\in\mathcal{G}_n}$ denote the Lovász vector of G for $\mathcal{G}_n$.

Theorem [Lovász, 1968]. Two arbitrary graphs $G, H \in \mathcal{G}_n$ are isomorphic iff $\varphi_n(G) = \varphi_n(H)$.

That means that $\varphi_n(\cdot)$ is complete!

Properties of Homomorphism Counts

hom ({o}, G) = |V(G)|

hom ({o—o}, G) = 2|E(G)|

hom ({o, o—o, …}, G)
≙ degree sequence of G

hom ({o, o—o, …}, G)
≙ eigenvalues of adj (G)

hom ({F | F is a tree}, G) ≙ 1-WL ≙ GNNs

hom ({ F | tw(F) ≤ k}, G) ≙ k-WL ≙ k-GNNs

treewidth of F ("tree-likeness")

Counting subgraphs [Curticapean et al., STOC 2017]

sub (, G) =
1/2 hom (, G) − hom (, G)
− hom (, G) − 1/2 hom (, G)
− 1/2 hom (, G) + 3/2 hom (, G)
+ 5/2 hom (, G) − hom (, G)

Universality [NT and Maehara, ICML 2020]: Any permutation-invariant function

$$f : \mathcal{G} \to \mathbb{R}^d$$

can be approximated arbitrarily well by a polynomial of

$$\{\hom(F, G) \mid F \in \mathcal{G}\}$$

Working on Arbitrary Graph Sizes

If we cannot restrict the size of graphs at inference time, we can define a kernel on $\mathcal{G}_\infty$ without restricting to $\mathcal{G}_n$ for some $n \in \mathbb{N}$. We define the countable-dimensional vector

$$\overline{\varphi}_\infty(G) = \left(\hom_{|V(G)|}(F, G)\right)_{F\in\mathcal{G}_\infty}$$

where

$$\hom_{|V(G)|}(F, G) = \begin{cases} \hom(F, G) & \text{if } |V(F)| \leq |V(G)|\,, \\ 0 & \text{if } |V(F)| > |V(G)|\,. \end{cases}$$

That is, $\overline{\varphi}_\infty(G)$ is the projection of $\varphi_\infty(G)$ to the subspace that gives us the homomorphism counts for all graphs of *size at most* of G. Note that this is a well-defined map of graphs to a subspace of the ℓ^2 space, i.e., sequences $(x_i)_i$ over $\mathbb{R}$ with $\sum_i |x_i|^2 < \infty$.

Theorem. $\overline{\varphi}_\infty$ is complete.

Theorem. $\overline{\varphi}_X$ is complete in expectation.

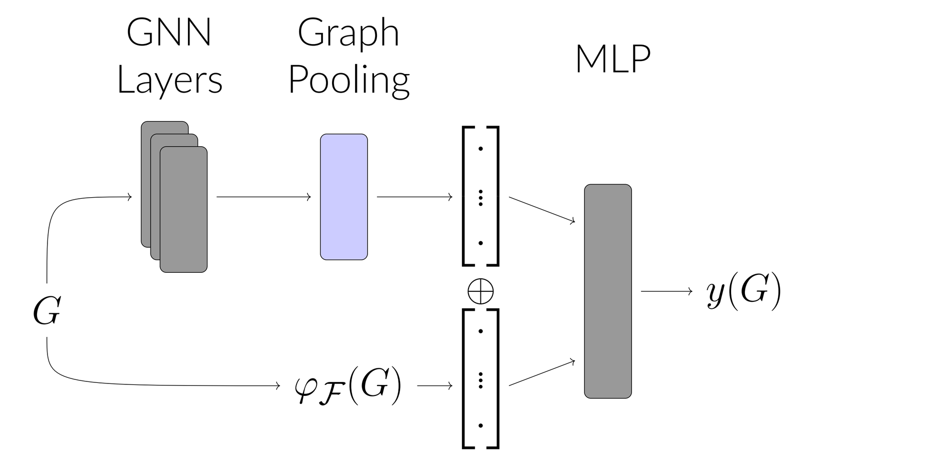
The map $\overline{\varphi}_\infty$ even maps all graphs into an inner product space and allows to compute norms or distances, and to apply kernel methods.

Empirical Results

Our method with $\ell = 50$ sampled patterns and the $\overline{\varphi}_\infty$ embedding

DATA SET → ↓ MODEL	MOLBACE roc-auc ↑	MOLCLINTOX roc-auc ↑	MOLBBBP roc-auc↑	MOLSIDER roc-auc ↑	MOLTOXCAST roc-auc ↑
GIN	82.2 ± 2.0	61.2 ± 4.5	60.9 ± 2.4	57.5 ± 1.4	57.1 ± 0.8
GIN+hom	$\mathbf{82.7 \pm 1.8}$	$\mathbf{61.5 \pm 4.1}$	$\mathbf{63.0 \pm 1.1}$	$\mathbf{58.4 \pm 1.2}$	$\mathbf{58.1 \pm 0.5}$
GCN	81.4 ± 2.4	68.4 ± 3.6	59.2 ± 1.0	58.2 ± 1.3	58.6 ± 0.6
GCN+hom	$\mathbf{84.6 \pm 1.3}$	63.4 ± 4.7	$\mathbf{61.2 \pm 0.7}$	$\mathbf{59.2 \pm 1.2}$	$\mathbf{59.4 \pm 0.4}$
GIN+F	75.5 ± 3.0	84.8 ± 3.7	67.2 ± 1.5	57.7 ± 1.8	61.8 ± 0.8
GIN+hom+F	$\mathbf{76.4 \pm 2.6}$	$\mathbf{86.9 \pm 3.5}$	$\mathbf{68.8 \pm 1.3}$	$\mathbf{58.4 \pm 1.5}$	$\mathbf{63.2 \pm 0.8}$
GCN+F	82.2 ± 1.4	88.2 ± 3.0	66.4 ± 2.6	59.3 ± 1.6	65.7 ± 0.4
GCN+hom+F	81.3 ± 1.6	$\mathbf{90.4 \pm 2.0}$	$\mathbf{70.8 \pm 1.2}$	$\mathbf{60.0 \pm 1.9}$	$\mathbf{65.8 \pm 0.8}$
	MOLLIPO rmse ↓	MOLTOX21 roc-auc ↑	MOLESOL rsmse ↓	MOLHIV roc-auc ↑	ZINC mae ↓
GIN	1.062 ± 0.025	65.4 ± 1.9	1.852 ± 0.044	69.1 ± 2.2	1.262 ± 0.017
GIN+hom	$\mathbf{1.006 \pm 0.017}$	$\mathbf{67.5 \pm 1.1}$	$\mathbf{1.746 \pm 0.096}$	$\mathbf{71.0 \pm 1.9}$	$\mathbf{1.231 \pm 0.014}$
GCN	1.056 ± 0.035	66.7 ± 0.7	1.855 ± 0.073	69.1 ± 2.2	1.281 ± 0.013
GCN+hom	$\mathbf{0.986 \pm 0.015}$	$\mathbf{66.8 \pm 1.1}$	$\mathbf{1.735 \pm 0.066}$	$\mathbf{72.2 \pm 1.4}$	$\mathbf{1.26 \pm 0.014}$
GIN+F	0.739 ± 0.019	75.4 ± 0.9	1.197 ± 0.061	76.5 ± 2.0	0.208 ± 0.005
GIN+hom+F	$\mathbf{0.71 \pm 0.021}$	75.2 ± 0.8	$\mathbf{1.014 \pm 0.044}$	$\mathbf{77.7 \pm 1.5}$	$\mathbf{0.174 \pm 0.005}$
GCN+F	1.188 ± 1.387	77.2 ± 0.6	1.197 ± 0.069	78.3 ± 1.0	0.234 ± 0.007
GCN+hom+F	$\mathbf{0.816 \pm 0.282}$	$\mathbf{78.0 \pm 0.6}$	$\mathbf{0.991 \pm 0.045}$	$\mathbf{78.8 \pm 1.3}$	$\mathbf{0.207 \pm 0.008}$

Expectation-Complete GNNs



Future Work

Choose number of patterns ℓ and distribution $\mathcal{D}$ adaptively:

- stop sampling when expressive enough
- pick $\mathcal{D}$ based on the task or a given dataset

Going beyond expressiveness: similarity!

- if $G \approx H$ then $\varphi(G) \approx \varphi(H)$
- possible solution: cut distance