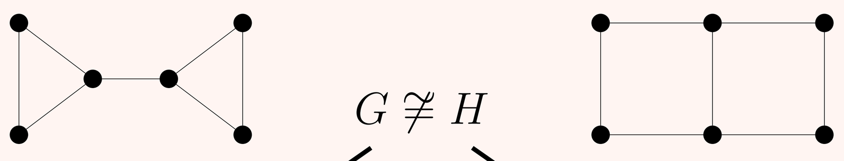
Graph Product Representations

Maximilian Seeliger Fabian Jogl Thomas Gärtner

Machine Learning Group — TU Wien

TL;DR



$G \ncong H$

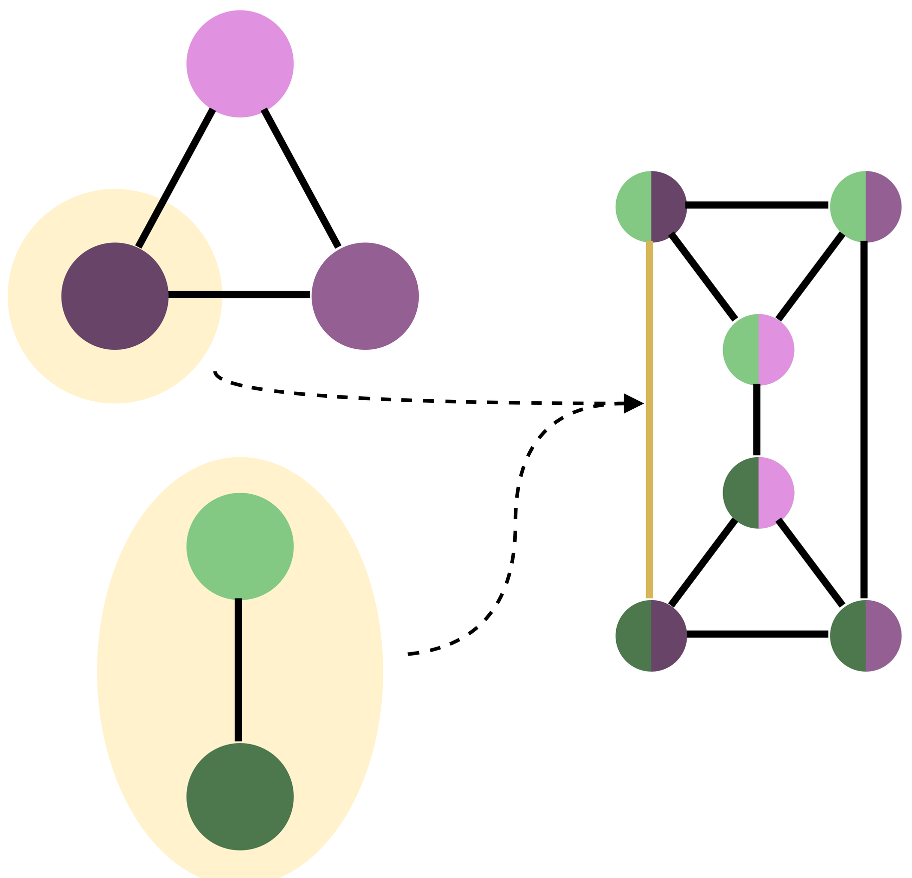
$\mathtt{WL}(G) = \mathtt{WL}(H)$ $\mathtt{PSC}(G) \neq \mathtt{PSC}(H)$

This work:

- Presents Product Substructure Count to utilize product graphs for expressive features (improve WL)
- Identify all small graphs $n \leq 7$
- Improve GNN expressiveness over benchmarks

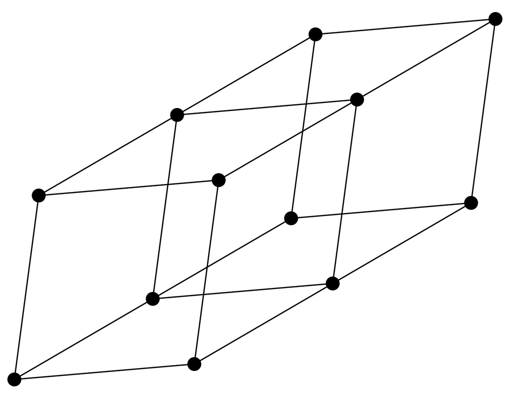
Products of Graphs

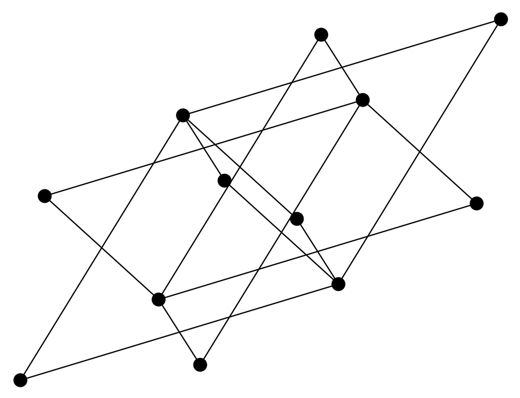
- Binary operation on graphs ($G \circ H = P$)
- $V(P) = V(G) \times V(H)$
- Different edge construction rules (vis.: $K_3 \square K_2$):



Cartesian Graph Product ($\square$). Two vertices are adjacent if they are adjacent in one graph and identical in the other.

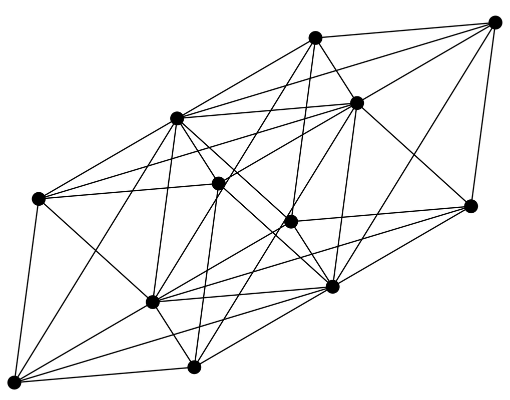
Direct Graph Product ($\times$). Vertices in the product graph are adjacent if they are adjacent in both original graphs.

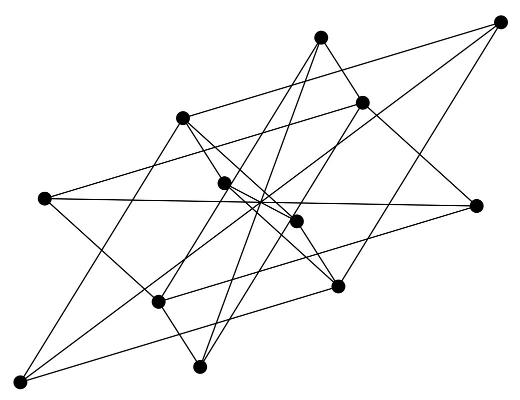




Strong Graph Product ($\boxtimes$). Combines the Cartesian and Direct products, where vertices are adjacent if they are adjacent in either one of them.

Modular Graph Product (∇). Two vertices are adjacent if they are either adjacent in both original graphs, or not adjacent in both.





Product as Unary Transformation

We use graph products as a unary transformation, parametrized by a fixed factor-graph F. Example for Cartesian graph product:

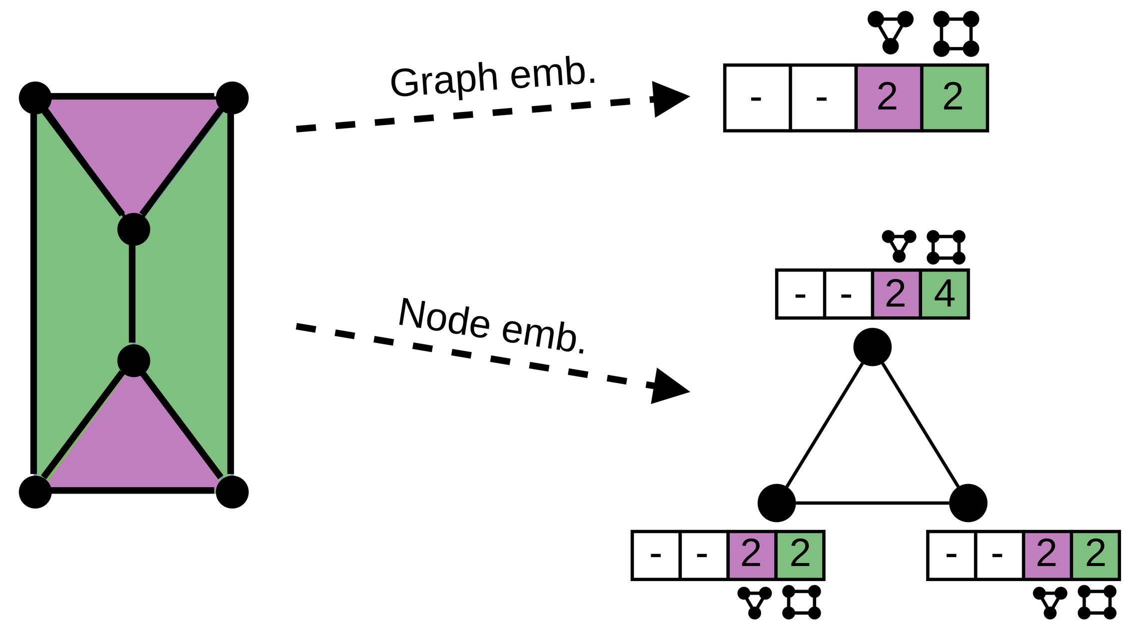
$$\square_F(G) = G \square F$$

- Representational enhancement of cycle space
- Correspondence lemmas for paths to cycles

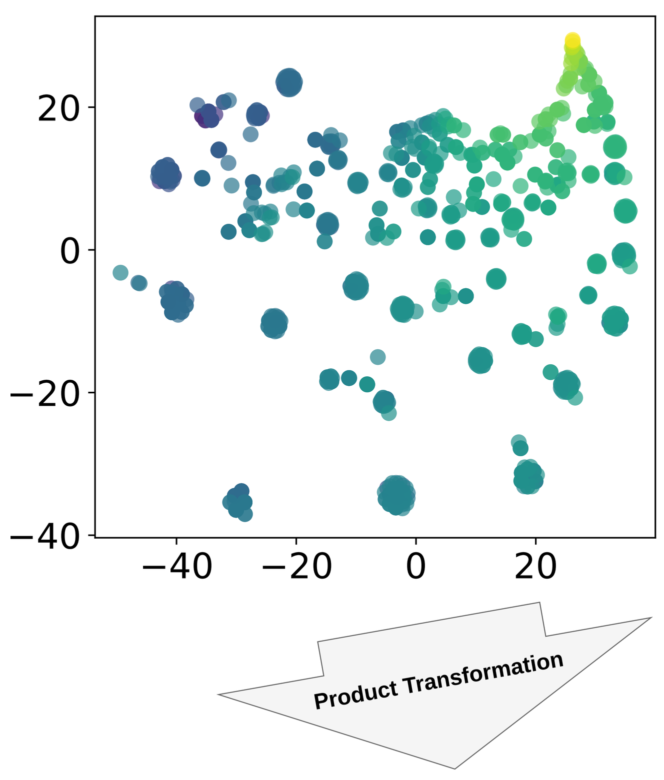
Product Substructure Count (PSC)

Input: graph G,
Fixed parameters: factor graph F, type of substructure $\mathcal{S}$

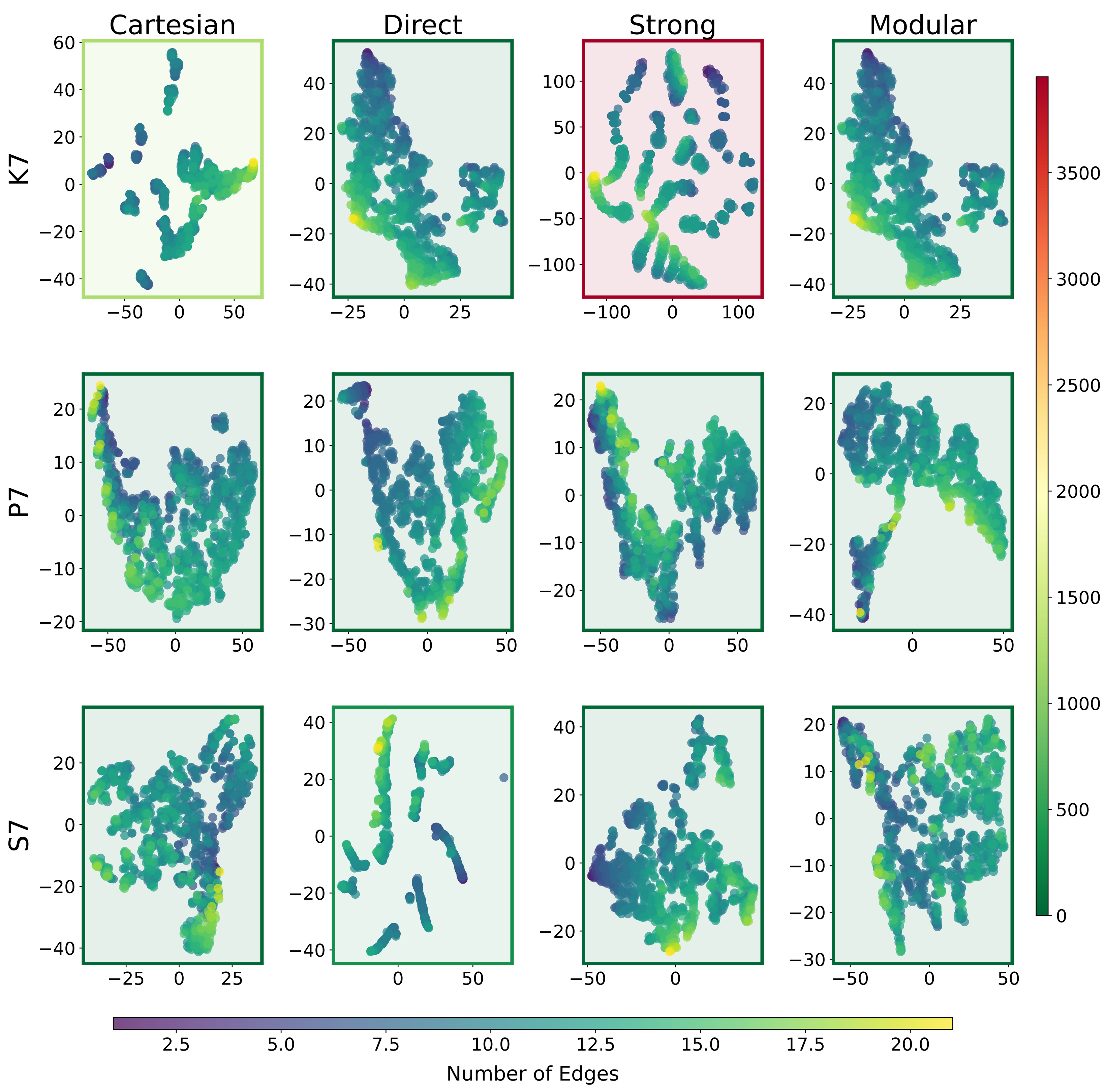
1. construct product graph using F
2. count all substructures of type $\mathcal{S}$
3. represent graph/node by substructure counts



Distinguishing Graphs



	Graph Products			
Factor	Cartesian	Direct	Strong	Modular
K_3	1245	0	3952	0
K_5	1244	0	3952	0
K_7	1244	0	3952	0
P_3	1	332	11	0
P_5	0	12	0	0
P_7	0	5	0	0
S_3	0	350	5	0
S_5	0	362	5	0
S_7	0	327	5	0



PSC as heuristic for isomorphism testing: Different embeddings → non-isomorphic. We experimentally analyze all connected graphs with $|V| \leq 7$. Looking at the set of all chordless cycles over several configurations of product operator and factor graphs, we can see that many combinations have 0 collisions — pairs of non-isomorphic graphs with same embedding. Weisfeiler-Leman test on the same set of graphs has 20 collisions. Product operation disperses the embeddings

Representation Learning

- Initial features with node-level PSC
- Approximate distribution over graph datasets better, by PSC using basis cycles.
- WL cannot count basis cycles

	Only degree	Cartesian + K_3	Modular + P_3
IMDB-BINARY	0.891 ± 0.005	0.901 ± 0.003	$\mathbf{0.926} \pm 0.005$
IMDB-MULTI	0.614 ± 0.009	0.628 ± 0.005	$\mathbf{0.640} \pm 0.005$
REDDIT-BINARY	0.986 ± 0.002	$\mathbf{0.996} \pm 0.001$	0.995 ± 0.002
SYNTHETIC	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
SYNTHETIC (no attr)	0.585 ± 0.011	0.584 ± 0.014	$\mathbf{0.593} \pm 0.022$
Synthie	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
MSRC-9	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
MUTAG	0.990 ± 0.006	$\mathbf{1.000} \pm 0.000$	0.995 ± 0.003
ENZYMES	0.984 ± 0.004	0.984 ± 0.005	$\mathbf{0.996} \pm 0.001$

Model expressiveness (larger → more expressive, measured by training accuracies)

Take-Home Message

- We cannot learn functions, that we cannot represent
- Graph products are useful for counting substructures, that WL cannot capture

maximilian.seeliger@tuwien.ac.at