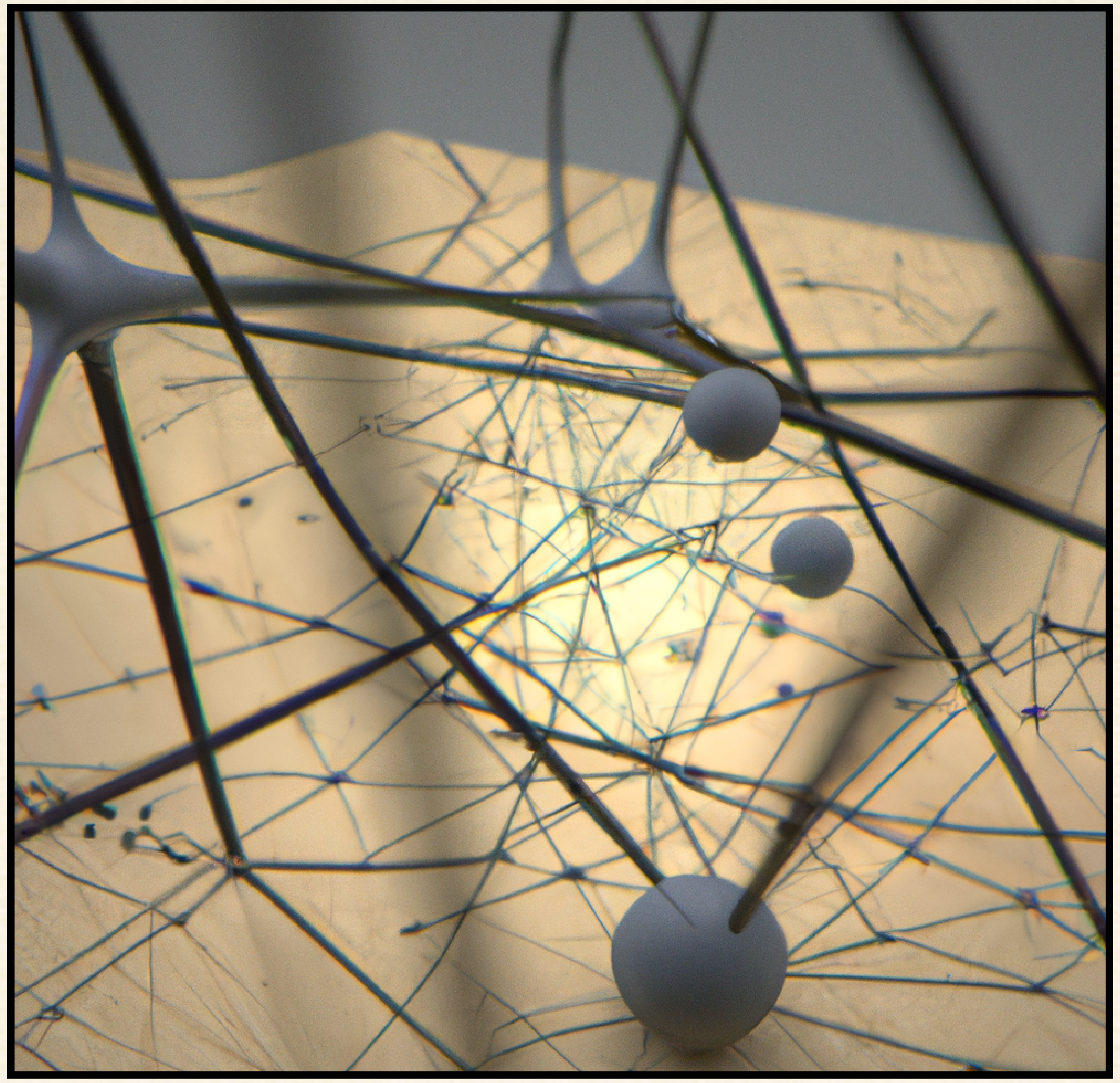


The (expressive) power of graph learning

Floris Geerts (University of Antwerp)

Course

- Is about recent advances in graph learning.
- With an emphasis on the expressive power of learning methods.
  - Self-contained (too some extent).
  - Mostly high-level, but also low-level, so basically all levels.
  - Not all methods or related works are covered.
  - Emphasis on theoretical aspects.

About the speaker

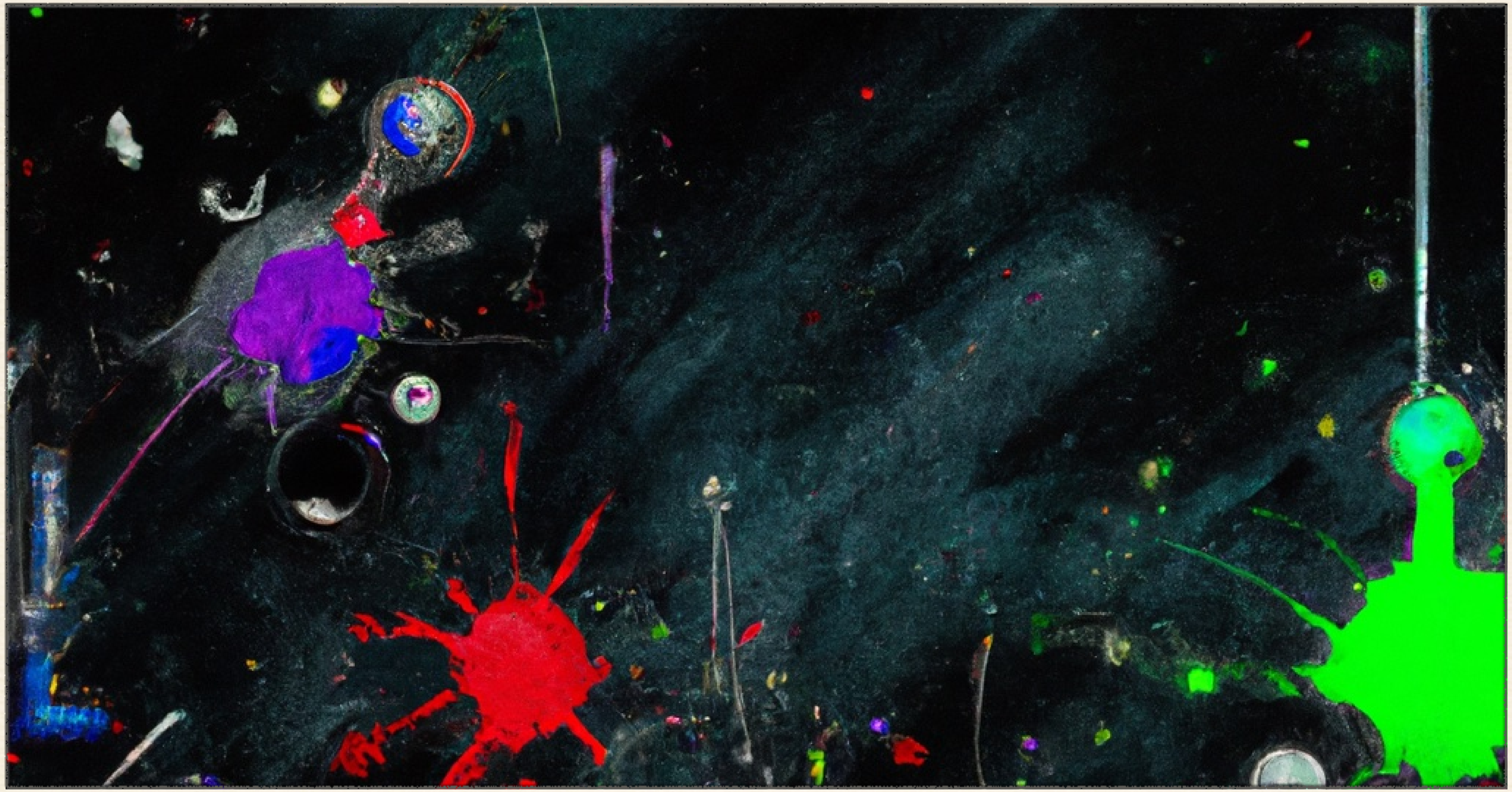
- Background in mathematics, database theory and expressive power of query languages.
- Since 2018, expressive power of linear algebra.
- Natural move to the study of expressive power of graph neural networks.
- Current focus is on generalisation and relational learning.

Outline

- Graph learning and expressive power
- Message Passing Neural Networks
- Boosting power:
  - Feature augmentation
  - Subgraphs
  - Higher-order message-passing

Ask Questions

Indebted to Fabrizio Frasca, Beatrice Bevilacqua and Haggai Maron: Shamelessly :-) borrowed parts of their tutorial on expressive of GNNs at LOG 2022

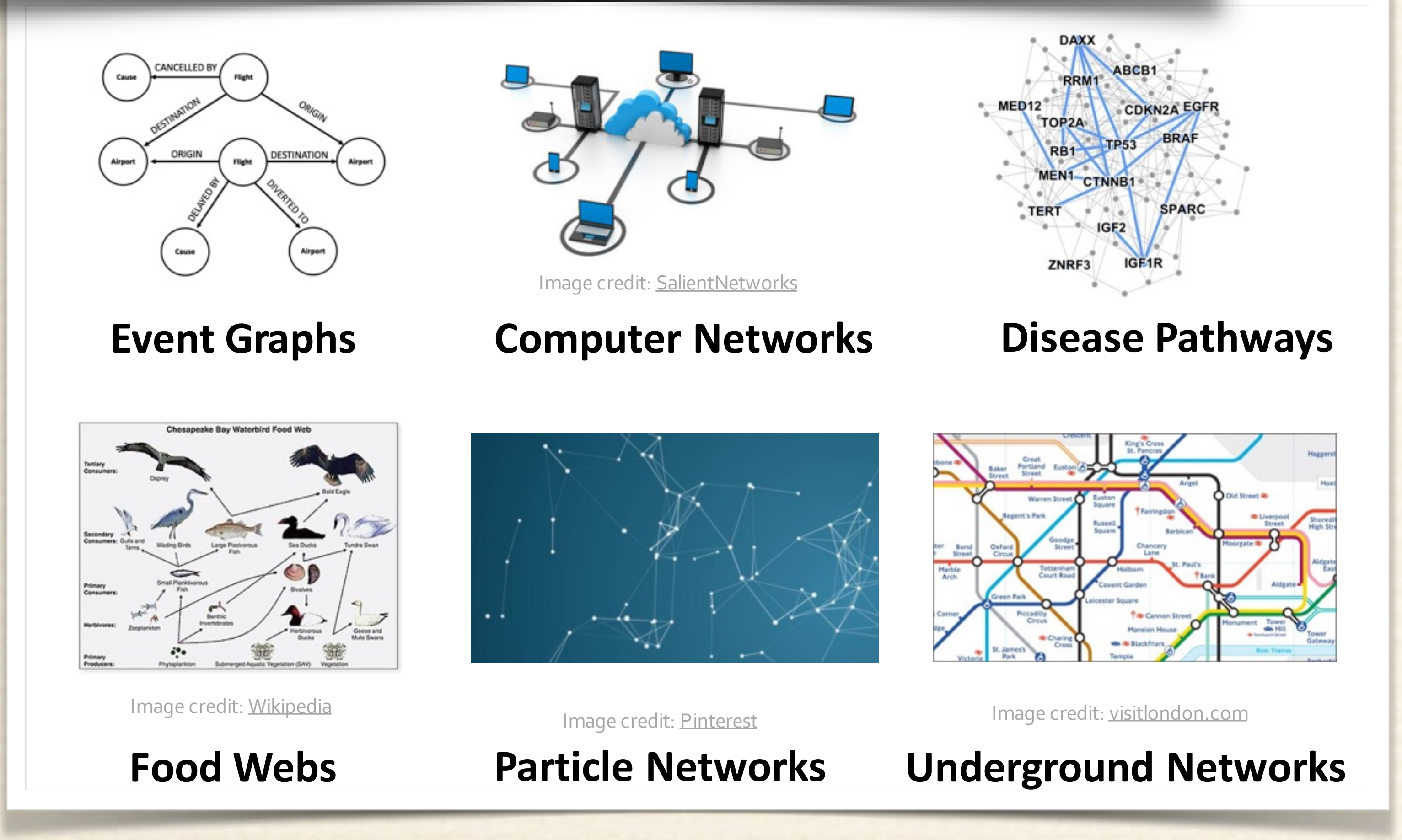


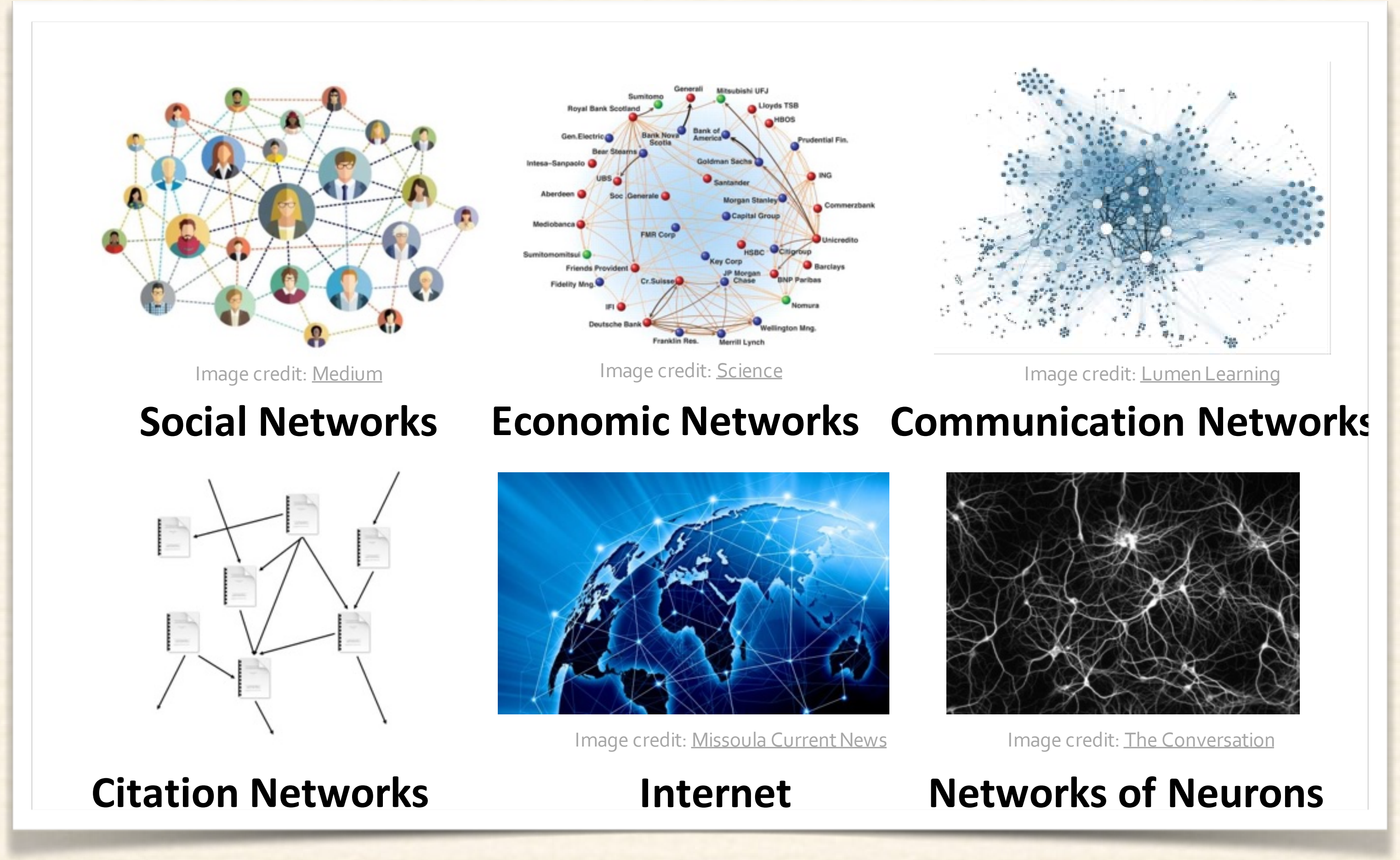
Graph learning

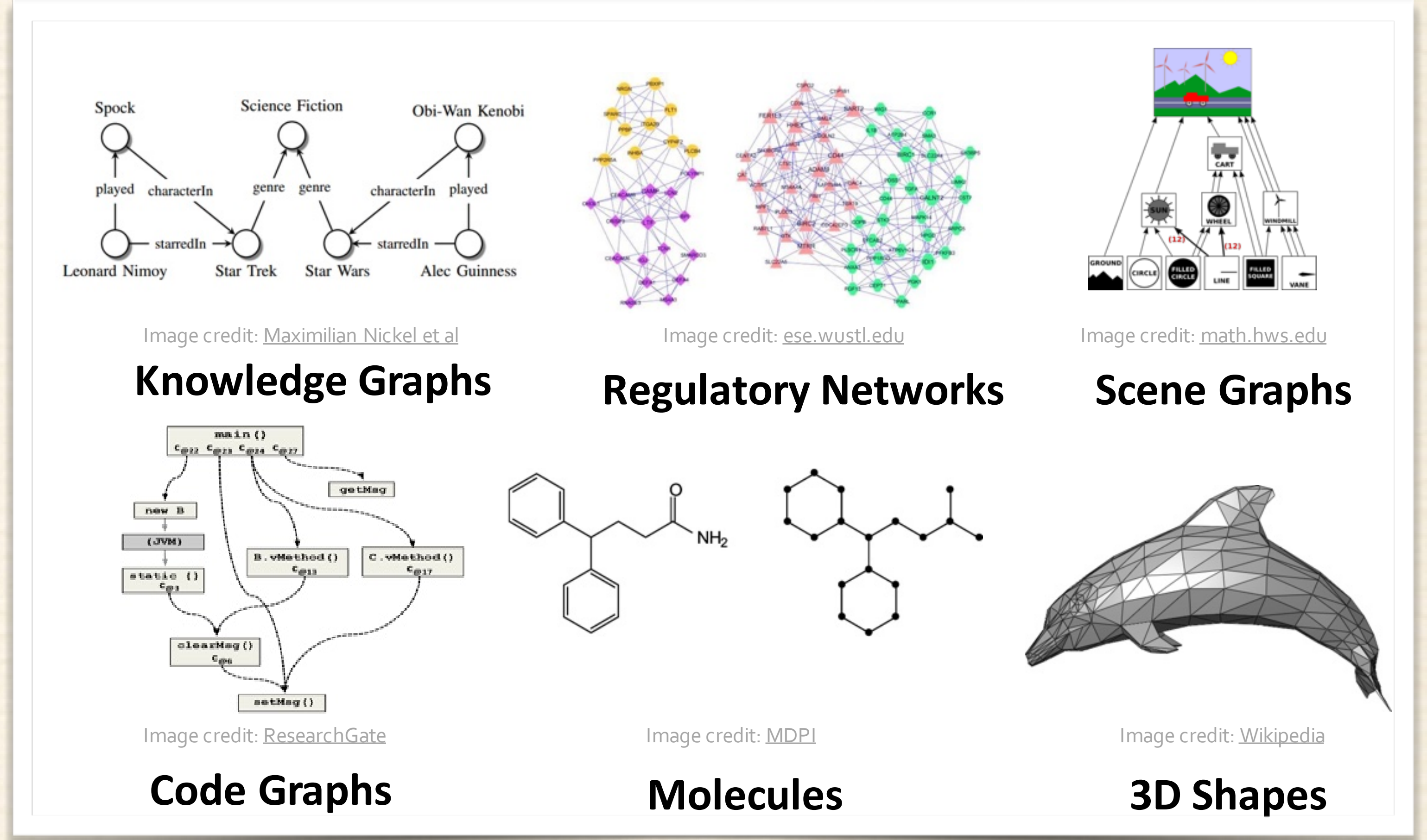
What is it?

Why learning on graphs?

Graphs are everywhere!





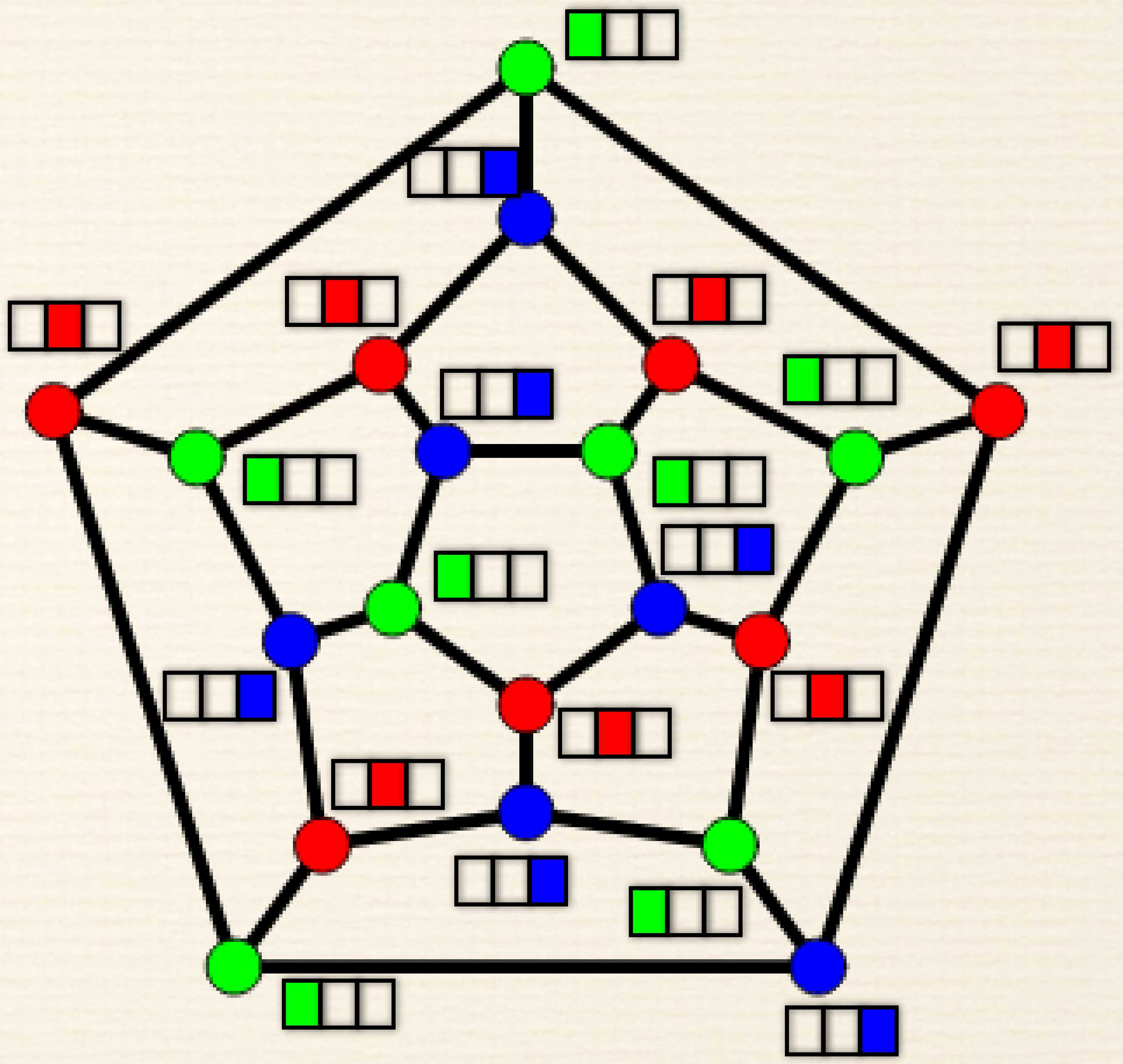


Images: Machine Learning on Graphs,Course by Jure Leskovec

Graphs: One definition to rule them all

- Graph $G = (V_G, E_G, L_G)$ with
  - Vertex set V_G
  - Edge set $E_G \subseteq V_G^2 := V_G \times V_G$
  - Vertex labels: $L_G : V_G \to \Sigma$

Vertex features $\mathbb{R}^d$



Hot-one encoding

Image: Wikipedia

Adjacency matrix representation

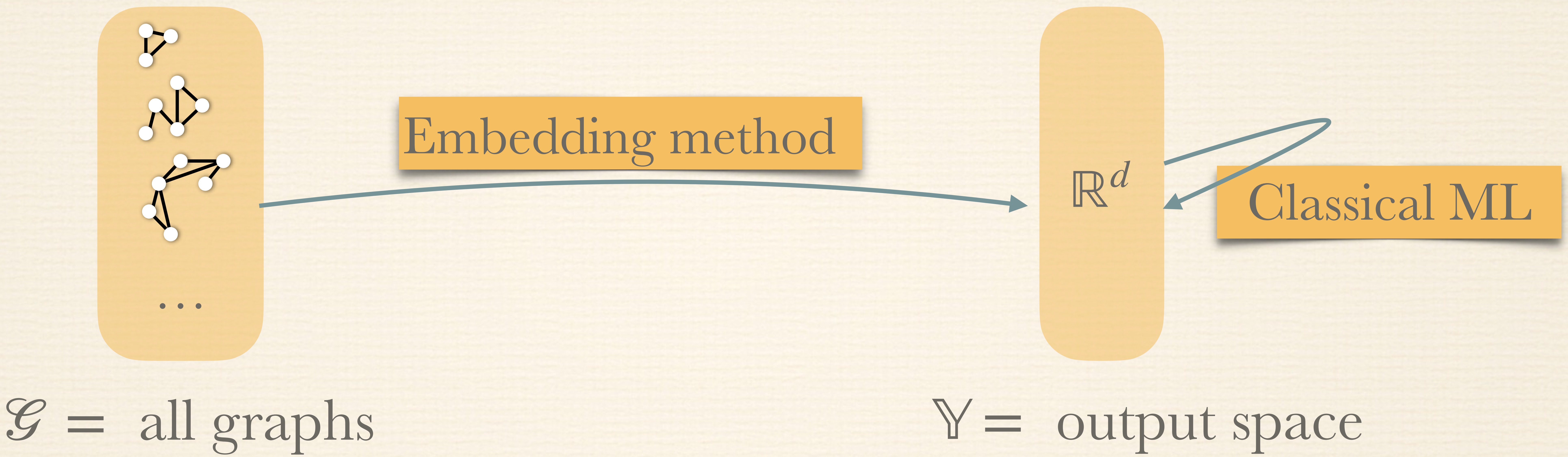
- Graph $G = (V_G, E_G, L_G)$ can also be represented by adjacency matrix A_G and feature matrix F_G

- Let $n = |V_G|$ be the number of vertices. Let $v, w \in [n] := \{1, \dots, n\}$.

adjacency matrix	$A_G \in \mathbb{R}^{n \times n} : (v, w) \mapsto \begin{cases} 1 & (v, w) \in E_G \\ 0 & \text{otherwise} \end{cases}$
feature matrix	$F_G \in \mathbb{R}^{n \times d} : v \mapsto L_G(v)$

- To turn graph into matrix, one needs an ordering on the vertices.

Graph learning



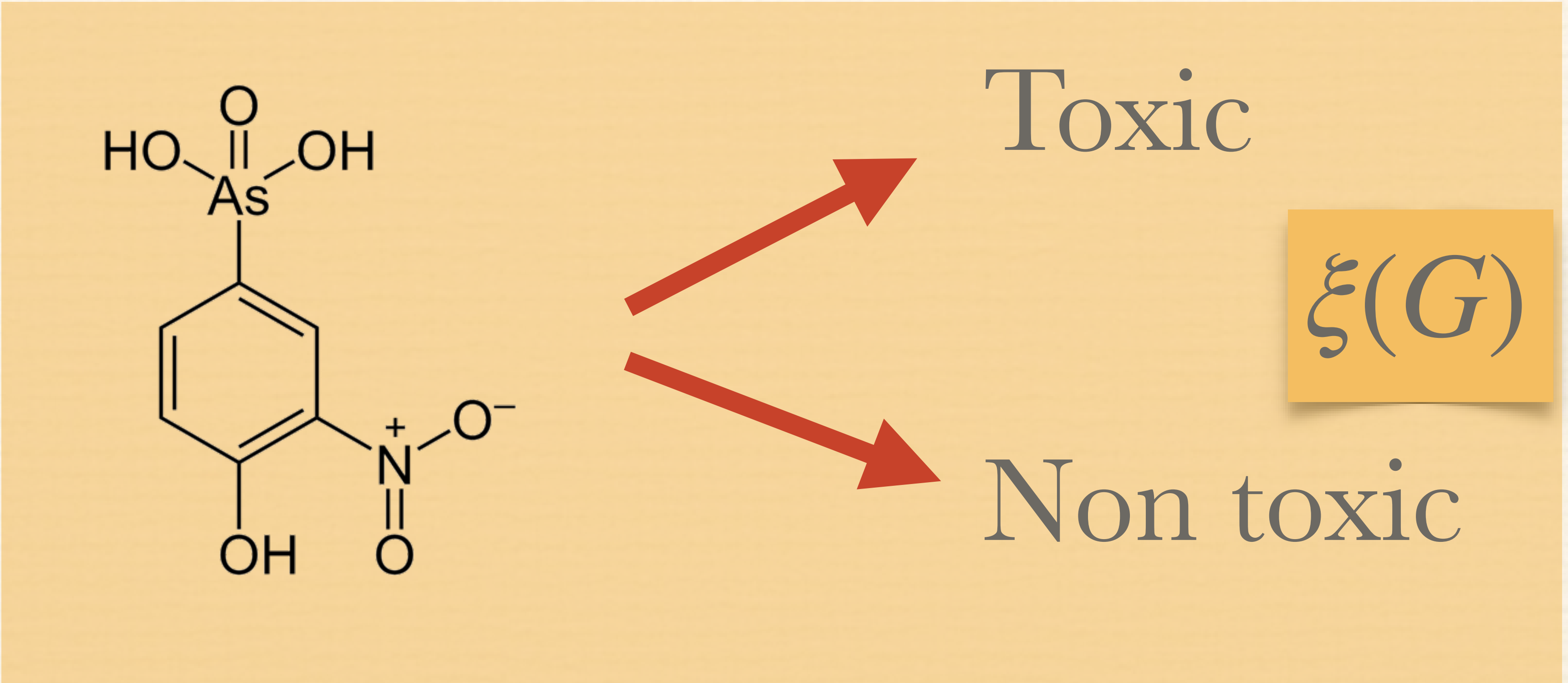
Embeddings

$\mathscr{G} =$ all graphs
$\mathscr{V} =$ all vertices
$\mathbb{Y} =$ output space

- Graph embedding: $\xi : \mathscr{G} \to \mathbb{Y}$
- Vertex embedding: $\xi : \mathscr{G} \to (\mathscr{V} \to \mathbb{Y})$
- p-Vertex embedding: $\xi : \mathscr{G} \to (\mathscr{V}^p \to \mathbb{Y})$

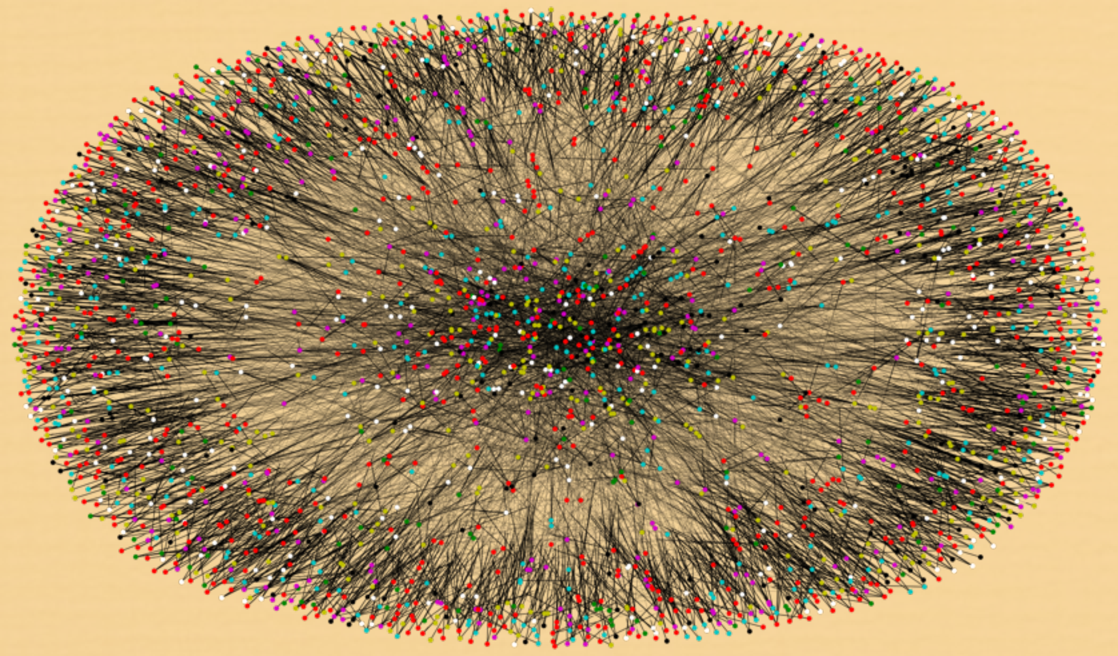
Graph embeddings

- Graph embedding: $\xi : \mathscr{G} \rightarrow \mathbb{Y}$
- Graph classification/regression



Vertex embeddings

- Vertex embedding: $\xi : \mathscr{G} \to (\mathscr{V} \to \mathbb{Y})$
- Vertex classification/regression. For example, prediction of subject of papers.



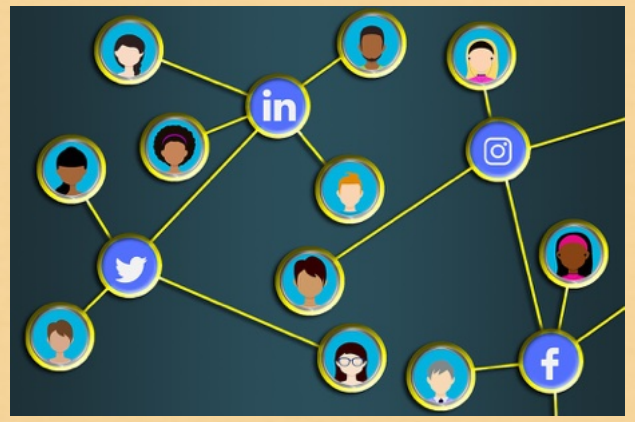
$$\to \begin{matrix} \text{paper}_1 & \to & \text{math} \\ \text{paper}_2 & \to & \text{computer science} \\ \vdots & \vdots & \vdots \\ \text{paper}_n & \to & \text{biology} \end{matrix}$$

$\xi(G, v)$

Images: Cora dataset

p-Vertex embeddings

- p-Vertex embedding: $\xi : \mathscr{G} \to (\mathscr{V}^p \to \mathbb{Y})$
- For example, 2-vertex embeddings: link prediction



$\xi(G, v, w)$

$\to$ (Joe, Anna) $\mapsto$ link

(Paolo, Mohammed) $\mapsto$ no link

$\ldots$

Graph learning tasks

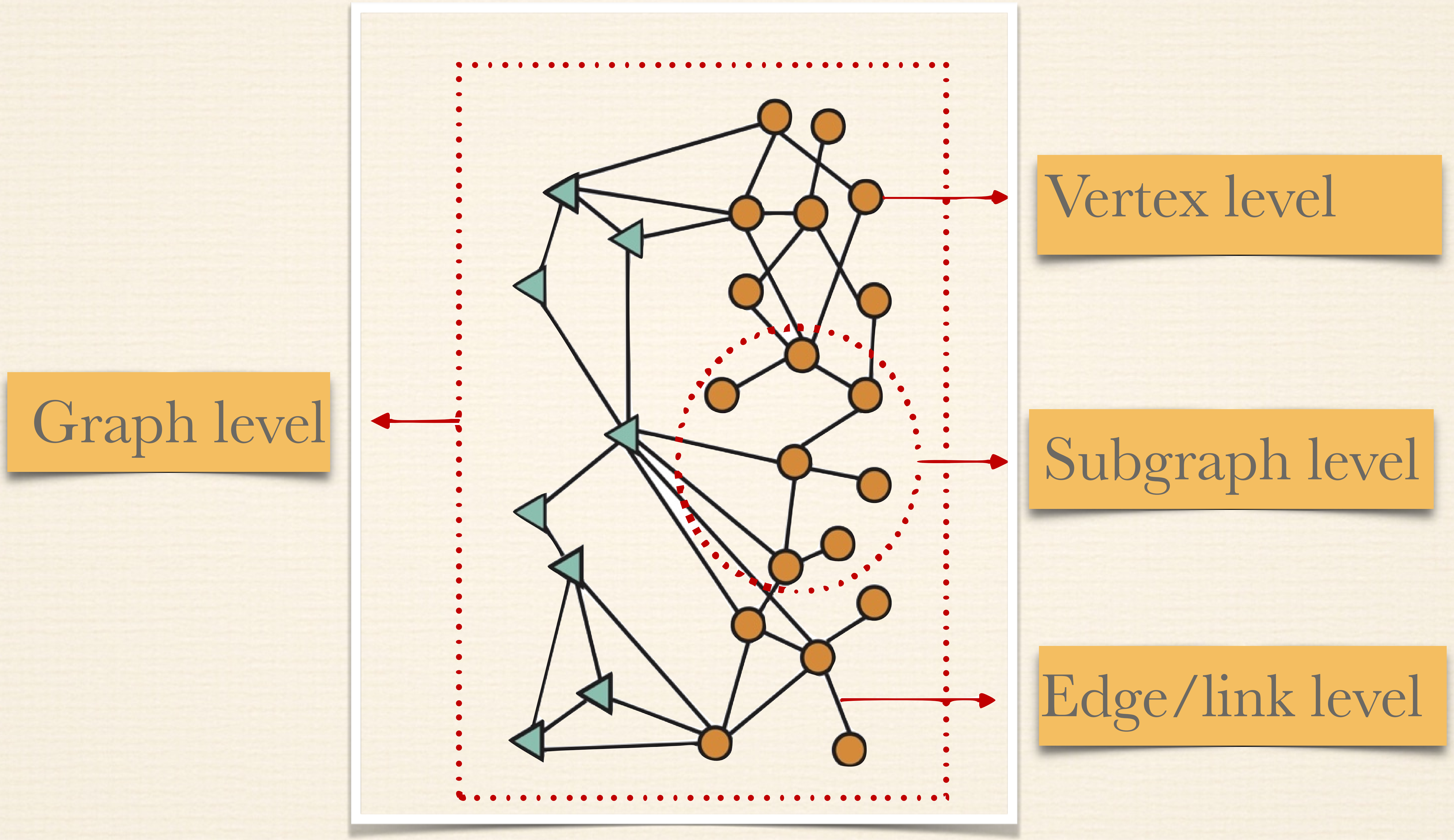
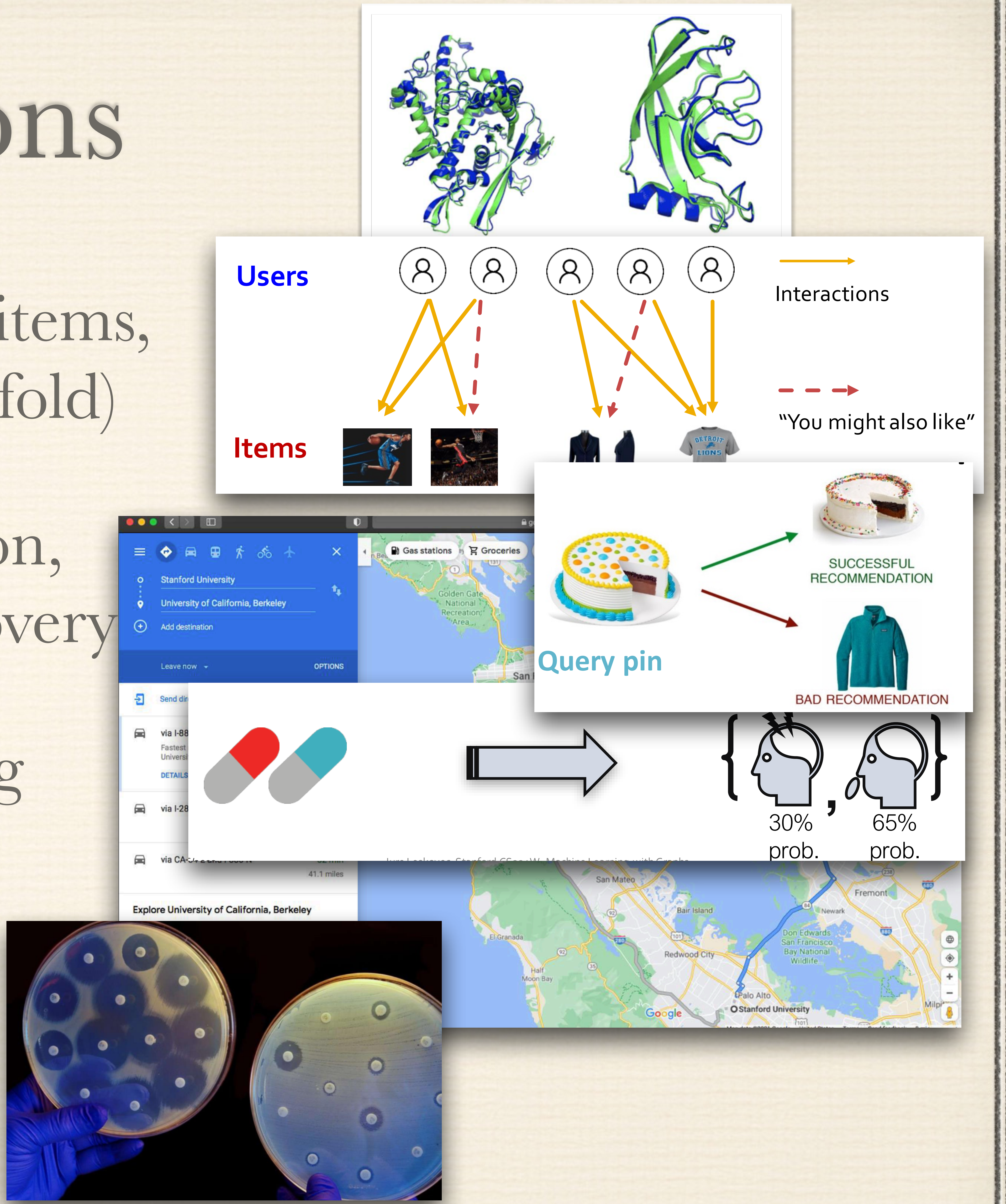


Image: Machine Learning on Graphs, course Jure Leskovec

Applications

- Vertex classification: categorise online user/items, location amino acids (protein folding, alpha fold)
- Link prediction: knowledge graph completion, recommender systems, drug side effect discovery
- Graph classification: molecule property, drug discovery
- Subgraph tasks: traffic prediction

GRAPH LEARNING HAS BECOME KEY DATA SCIENCE COMPONENT



Images: Machine Learning on Graphs, course by Jure Leskovec

Graph learning

- We want to learn an unknown embedding $\Xi : \mathcal{G} \to (\mathcal{V}^p \to \mathbb{Y})$

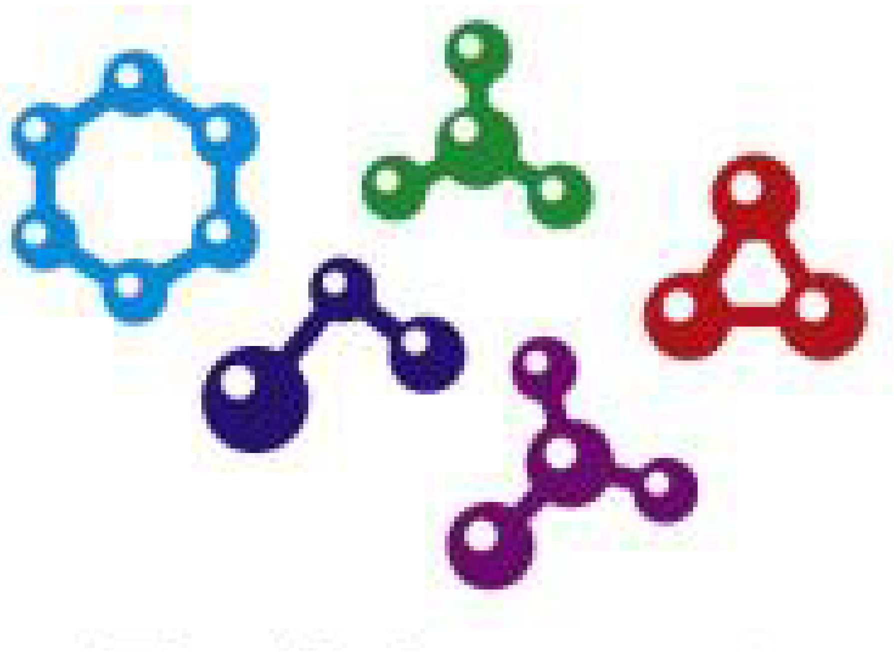
What does this mean???

- The embedding Ξ is partially revealed by means of a training set

$$\mathcal{T} := \{(G_1, \mathbf{v}_1, y_1), \ldots, (G_\ell, \mathbf{v}_\ell, y_\ell)\} \subseteq \mathcal{G} \times \mathcal{V}^p \times \mathbb{Y}$$

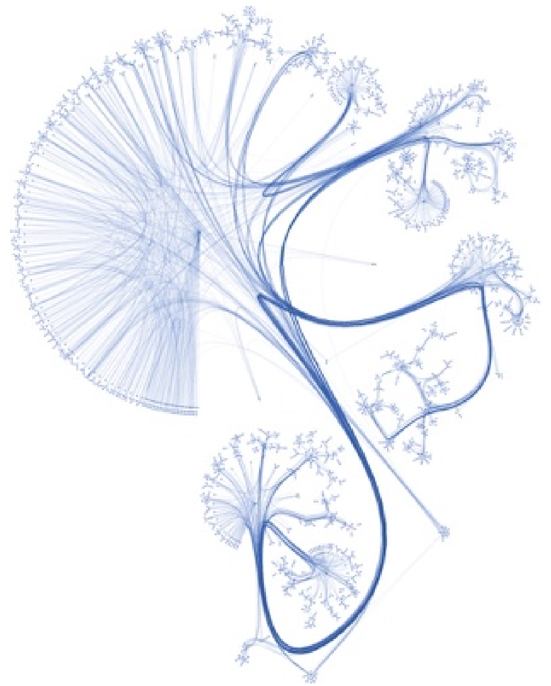
$\Xi(G_1, \mathbf{v}_1)$ $\Xi(G_\ell, \mathbf{v}_\ell)$

Training sets



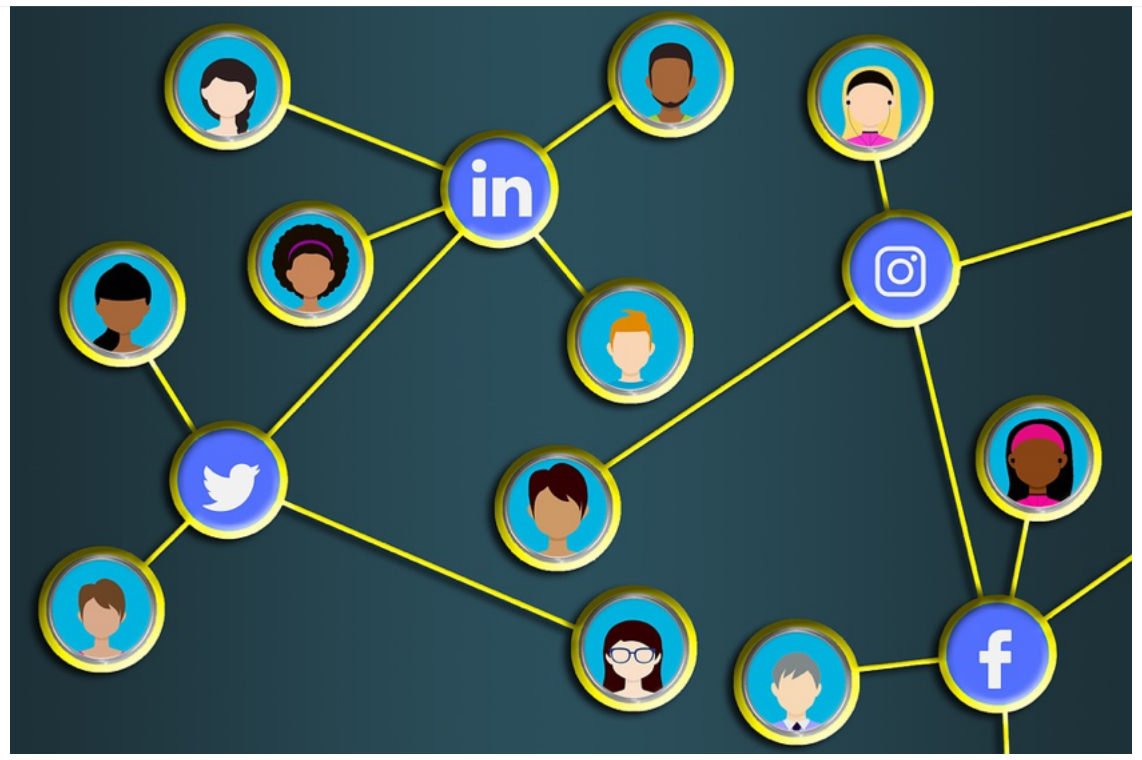
(molecule,yes/no)

Graph classification



(cora, paper, topic)

Vertex classification



(social,p_x,p_y, yes/no)

Link prediction

Graph learning: hypothesis class

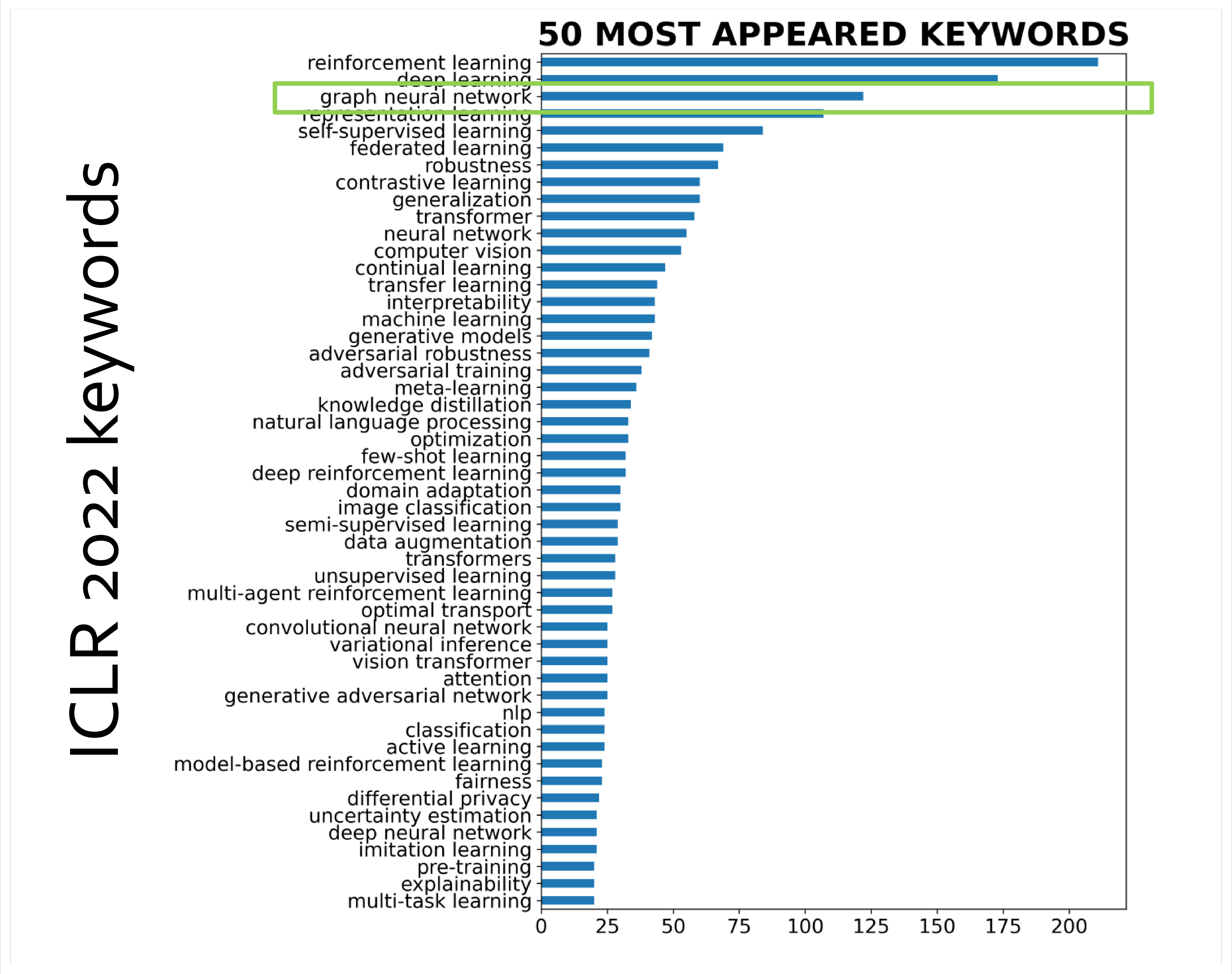
- We want to find the best model consistent with training set $\mathcal{T}$

What does this mean???

- Models are selected from an hypothesis class $\mathcal{H}$
- In the graph setting $\mathcal{H}$ consists of embeddings

Hypothesis classes

Hypothesis Class Explosion

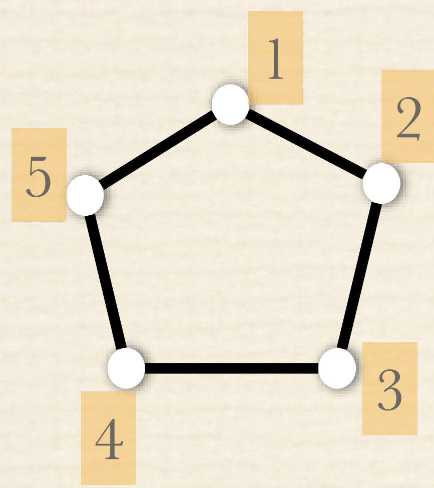


Count	
Large Language Models	318
Reinforcement Learning	201
Graph Neural Networks	123
Diffusion Models	112
Deep Learning	110
Representation Learning	107

Images: Machine Learning on Graphs, course J. Leskovec

What's new?

$\mathcal{G}$

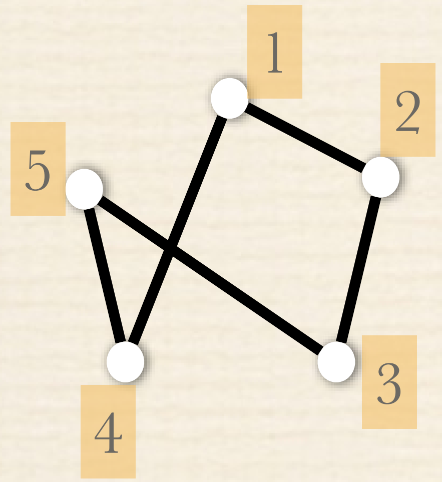


$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

flatten

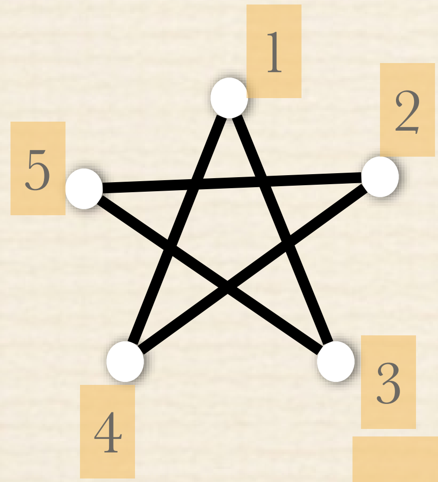
$\mathbb{Y} = \mathbb{R}^{10}$

(0,1,0,0,1,1,0,1,0,0,0,1,0,1,0,0,0,1,0,1,1,0,0,1,0) →



$$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

(0,1,0,1,0,1,0,1,0,0,0,1,0,0,1,1,0,0,0,1,0,0,1,1,0) →



$$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

(0,0,1,1,0,0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,1,1,0,0) →

permuted adjacency matrices

Deep neural network

Support vector machines

…

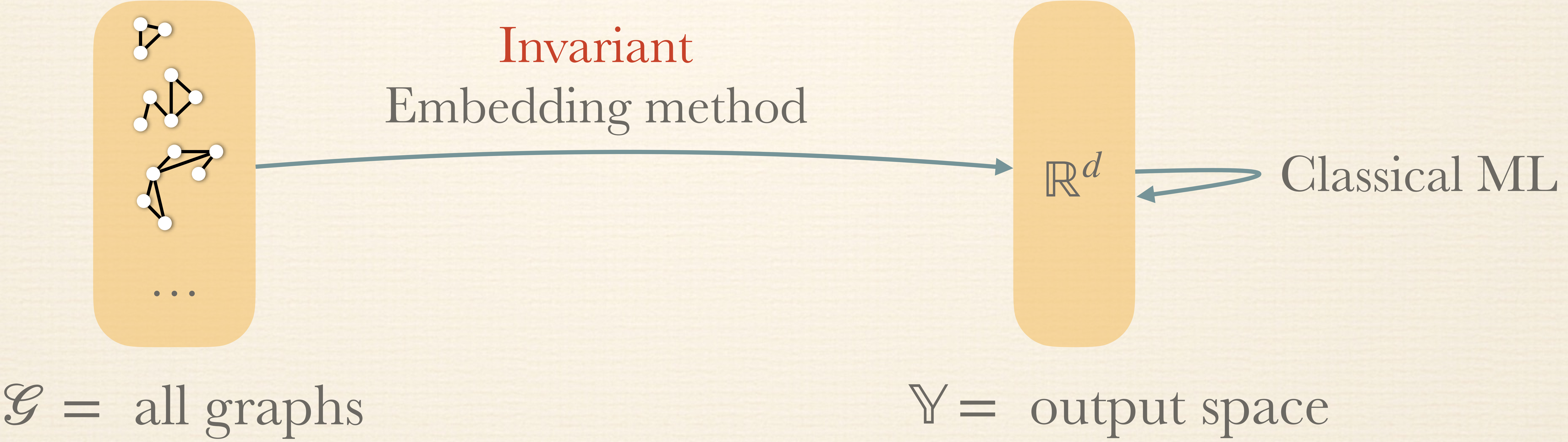
Different representation ⇒ different result

Images: flaticon.com, Noun project

Graph learning

Classical embedding methods depend on representation

E.g., think of **M**ulti**L**ayer**P**erceptron on vector representation of flattened adjacency matrix



A desired property: Invariance

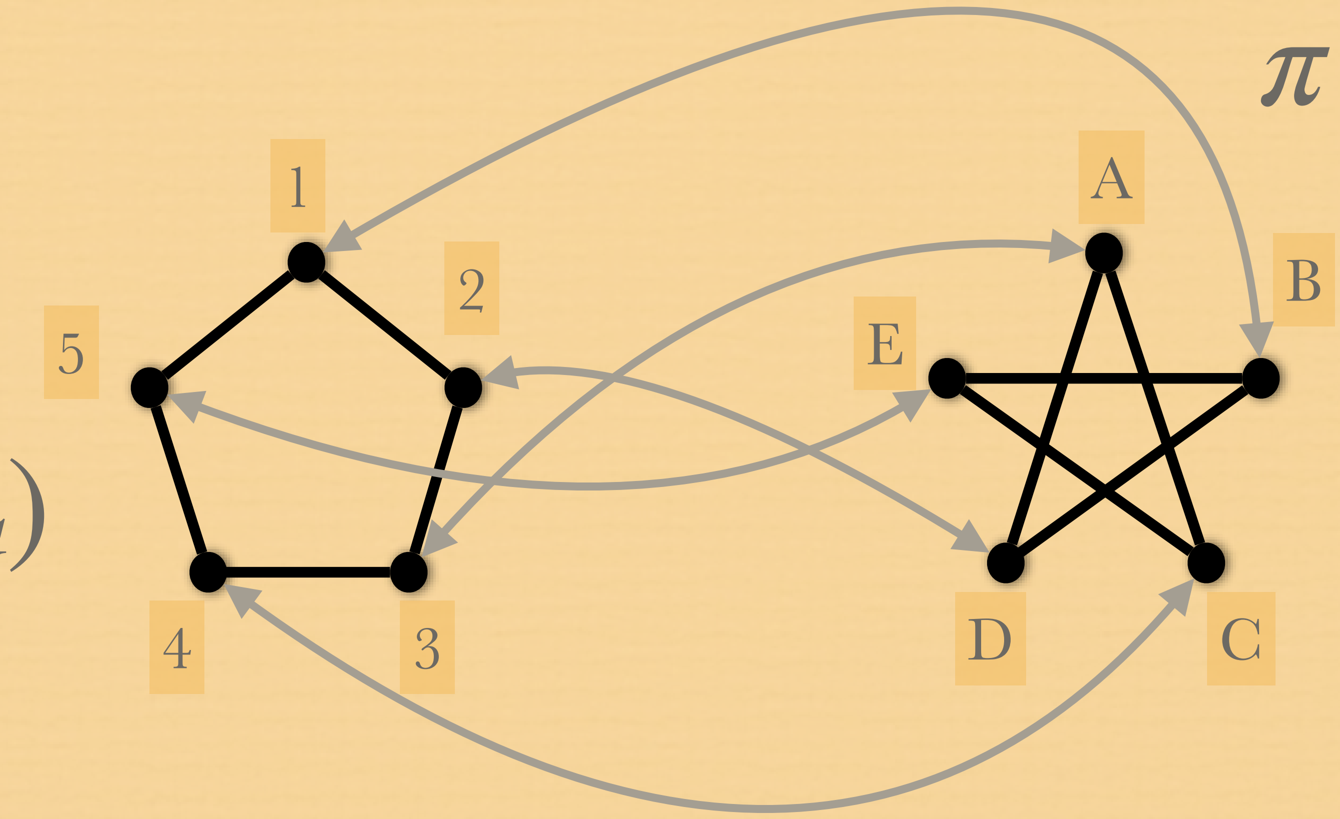
- Embeddings should be invariant, that is, independent of the chosen graph representation.

- Invariance is defined in terms of graph isomorphisms.

$G \cong H$

$\pi : V_G \to V_H$

$G = (V_G, E_G)$



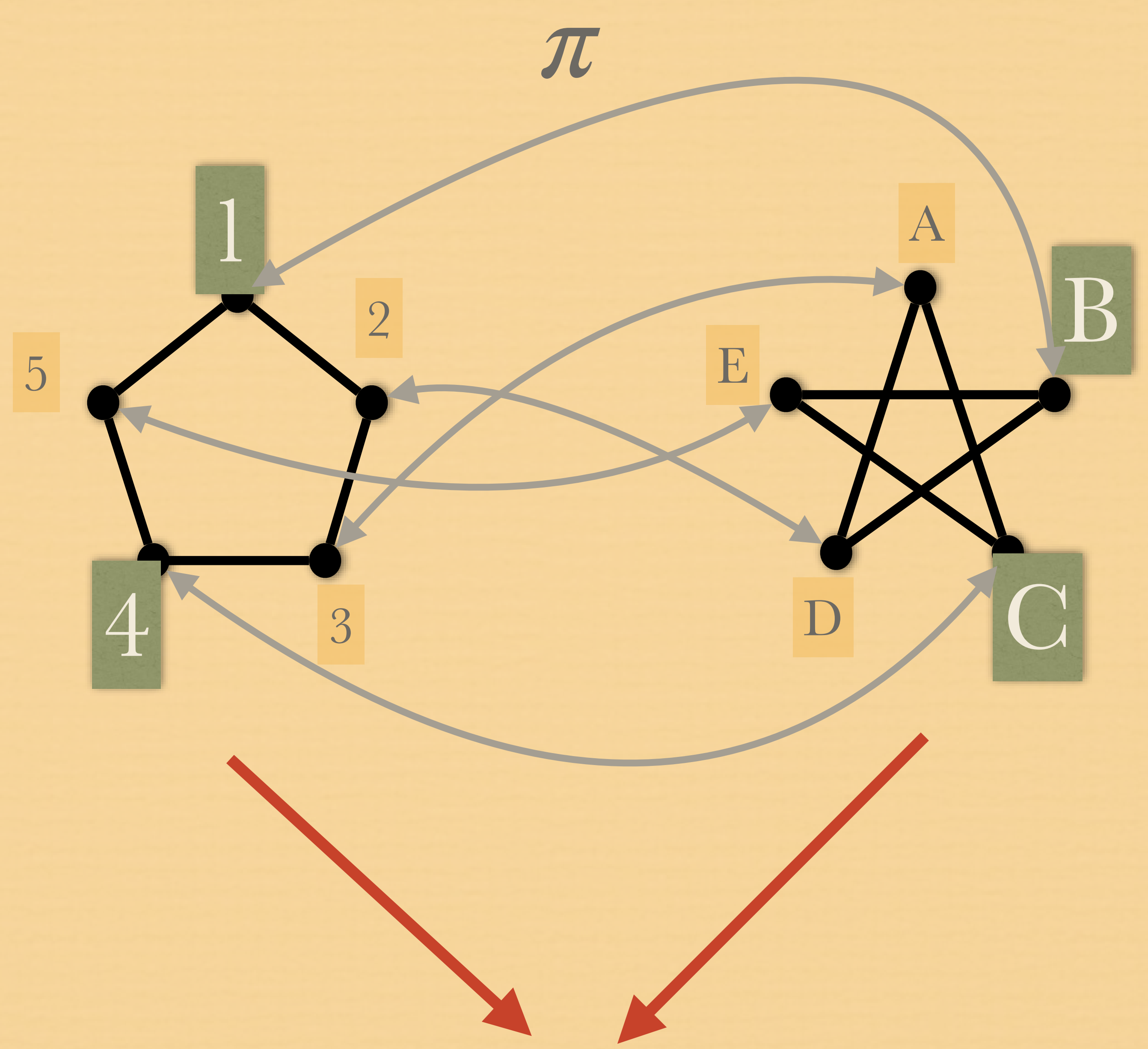
$H = (V_H, E_H)$

- The mapping π is a bijective vertex function satisfying $(v, v') \in E_G \iff (\pi(v), \pi(w)) \in E_H$ also $L_G(v) = L_H(\pi(v))$ must hold.

Invariant embeddings

for all π, G and $\mathbf{v} \in V_G^p : \xi(G, \mathbf{v}) = \xi(\pi(G), \pi(\mathbf{v}))$

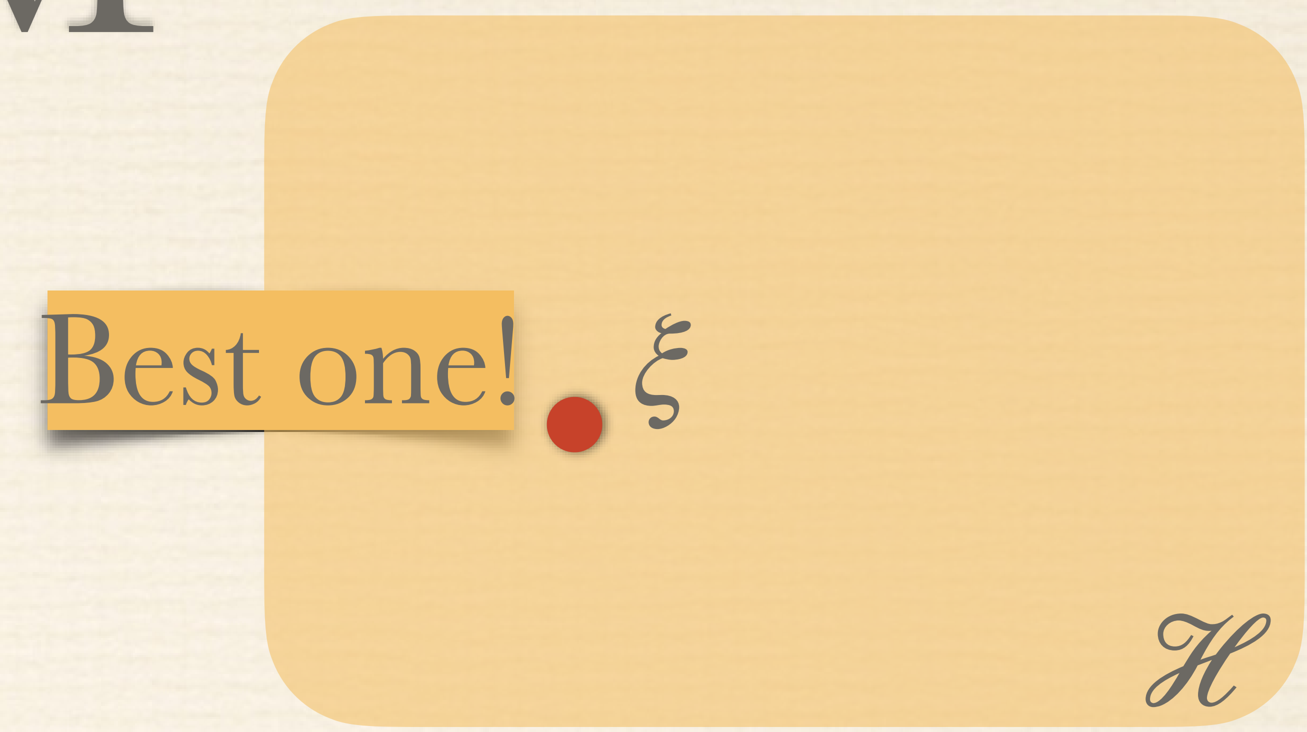
Isomorphism



(1,4) and (B,C) have same embedding in $\mathbb{Y}$

We typically assume invariant embedding methods (unless said otherwise)

Graph learning: ERM



- Given training set $\mathscr{T}$ and hypothesis class $\mathscr{H}$
- Empirical risk minimisation:

Find embedding ξ in $\mathscr{H}$ which minimises empirical loss

$$\frac{1}{\ell}\sum_{i=1}^{\ell} \text{loss}(\xi(G_i, \mathbf{v}_i), y_i))$$

Loss function is a mapping from $\mathbb{Y} \times \mathbb{Y} \to \mathbb{R}$

Loss functions

- Choice depends on learning task (regression, classification,…)

- L1: $\text{loss}(y_{predicted}, y_{true}) := |y_{predicted} - y_{true}|$
- L2: $\text{loss}(y_{predicted}, y_{true}) := (y_{predicted} - y_{true})^2$
- (Binary) cross entropy:
  $\text{loss}(y_{predicted}, y_{true}) := y_{true}\log(y_{predicted} + (1 - y_{true})\log(1 - y_{predicted})$

Graph learning

- Graph learning systems solve ERM using back propagation and gradient descent…

$$\hat{\xi} : \arg\min_{\xi \in \mathcal{H}} \frac{1}{\ell} \sum_{i=1}^{\ell} \text{loss}(\xi(G_i, \mathbf{v}_i), y_i))$$

What one really wants

- Simply predicting labels for training data is insufficient.
- We want to predict labels for graphs not in the training data.
- We will assume the presence of distribution $\mathscr{D}$ over $\mathscr{G} \times \mathbb{Y}$.

We do not know the true distribution $\mathscr{D}$, as it represents real-world unseen data.

Risk minimisation

- Risk minimisation: Find embedding $\tilde{\xi}$ in $\mathscr{H}$ which minimises expected loss over $\mathscr{D}$:

$$\tilde{\xi} := \arg\min_{\xi \in \mathscr{H}} \mathrm{Prob}_{(G,y)\sim\mathscr{D}}[\xi(G) \neq y]$$

- RM focuses on minimising errors over all the data according to their distribution.

Generalisation Error

- We want to find a hypothesis $\xi \in \mathscr{H}$ that does well on the training data (small empirical loss $L_{\mathscr{T}}(\xi)$)

- But also has small expected loss $L_{\mathscr{D}}(\xi)$

- We want the generalisation error $L_{\mathscr{D}}(\xi) - L_{\mathscr{T}}(\xi)$ to be small.

- Statistical learning theory provides bounds on training data guaranteeing small generalisation error.

Generalisation error

- We want the generalisation error $L_{\mathscr{D}}(\xi) - L_{\mathscr{T}}(\xi)$ to be small.

Theorem (Vapnik and Chervonenkis 1964)

- For $\delta > 0$, with probability $1 - \delta$ (in our selection of training data $\mathscr{T}$ of size m) for all $\xi \in \mathscr{H}$:

$$L_{\mathscr{D}}(\xi) - L_{\mathscr{T}}(\xi) \leq \sqrt{\frac{2d \log(\frac{em}{d})}{m}} + \sqrt{\frac{\log(\frac{1}{\delta})}{2m}}$$

- where d is the VC dimension of $\mathscr{H}$

Vladimir Vapnik and Alexey Chervonenkis: The uniform convergence of frequencies of the appearance of events to their probabilities (1964)

Graph learning

- Graph learning systems solve ERM using back propagation and gradient descent…

$$\hat{\xi} : \arg\min_{\xi \in \mathcal{H}} \frac{1}{\ell} \sum_{i=1}^{\ell} \mathsf{loss}(\xi(G_i, \mathbf{v}_i), y_i))$$

Our focus will be on the expressive power of hypothesis classes $\mathcal{H}$

Expressive power

- Which embeddings can be expressed by embeddings in $\mathscr{H}$?
- Which embeddings can be approximated by embeddings in $\mathscr{H}$?
- Which inputs can be separated/distinguished by embeddings in $\mathscr{H}$?
- What is the relationship between expressiveness and generalisation of $\mathscr{H}$?

Notions of expressivity

- Let $\Xi : \mathcal{G} \to (\mathcal{V}^p \to \mathbb{Y})$ be a *p*-vertex embedding

$\mathcal{H}$ can express Ξ if there exists a $\xi \in \mathcal{H}$ such that for all $G \in \mathcal{C}, \mathbf{v} \in V_G^p$:
$$\xi(G, \mathbf{v}) = \Xi(G, \mathbf{v})$$

- Let $\Xi : \mathcal{G} \to \{0,1\}$ indicator function for connected graphs.
- Can we find hypothesis in $\xi \in \mathcal{H}$ such that $\xi(G) = \Xi(G)$ for all graphs G

Notions of expressivity

Separation/distinguishing power of $\mathscr{H}$

$$\rho(\mathscr{H}) := \{(G, \mathbf{v}, H, \mathbf{w}) \mid \forall \xi \in \mathscr{H} : \xi(G, \mathbf{v}) = \xi(H, \mathbf{w})\}$$

- All pairs of inputs that cannot be separated by any embedding in $\mathscr{H}$

- Can we find hypothesis in $\xi \in \mathscr{H}$ such that $\xi(G) \neq \xi(H)$ for any connected graph G and disconnected graph H?

Distinguishing power

- Strongest power: $\mathscr{H}$ powerful enough to detect non-isomorphic graphs
- Weakest power: $\mathscr{H}$ cannot differentiate any two graphs

Distinguishing power

- Allows for comparing different classes of embeddings methods!

$$\rho(\text{methods1}) \subseteq \rho(\text{methods2})$$

Methods1 is more powerful than Methods2
Methods 2 is bounded by Methods 1 in power

$$\rho(\text{methods1}) = \rho(\text{methods2})$$

Both methods are as powerful

- Allows for comparing embedding methods with algorithms, logic, …

Expressive power in ML community

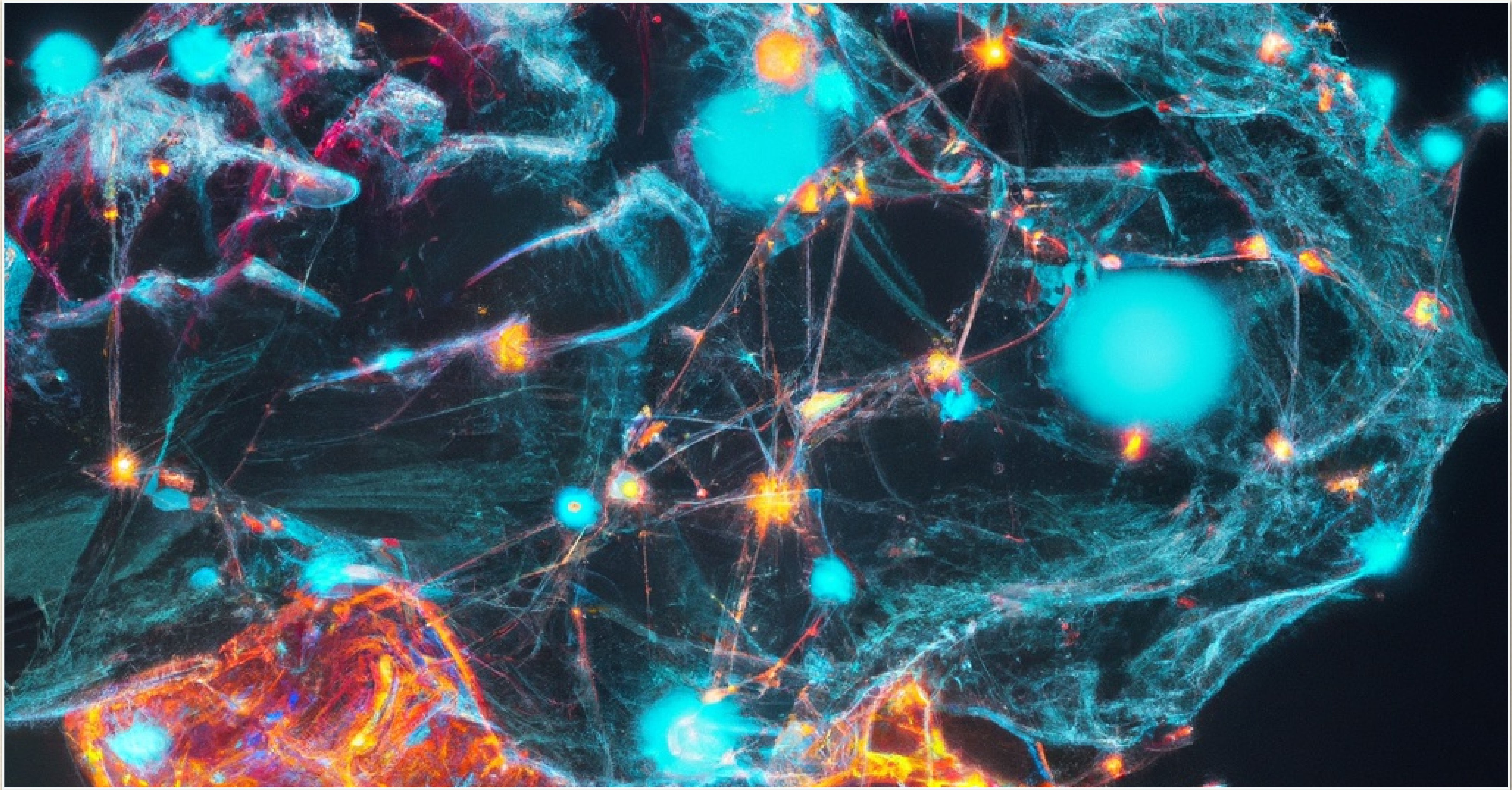
- Focus has been on distinguishing power of classes $\mathscr{H}$ of embedding methods.
- Goal is to characterise $\rho(\mathscr{H})$ in a way to sheds light on what graph properties a learning method can detect/use.
- We see an example shortly for $\mathscr{H}$ = the class of Message-Passing Neural Networks (MPNNs)
- Recent work addresses uniform expressiveness.

Expressive power in ML community

- Search for increase in expressive power has led to surge of new methods of graph learning.
- Despite theoretical underpinning… still a bit of alchemy to find the right method…


We will gradually fill in this landscape
with graph learning methods
Expressiveness
$\mathcal{H}$
Complexity

Questions?



Message Passing Neural Networks

(Still) the most popular type of Graph Learning Architecture

A little graph embedding history

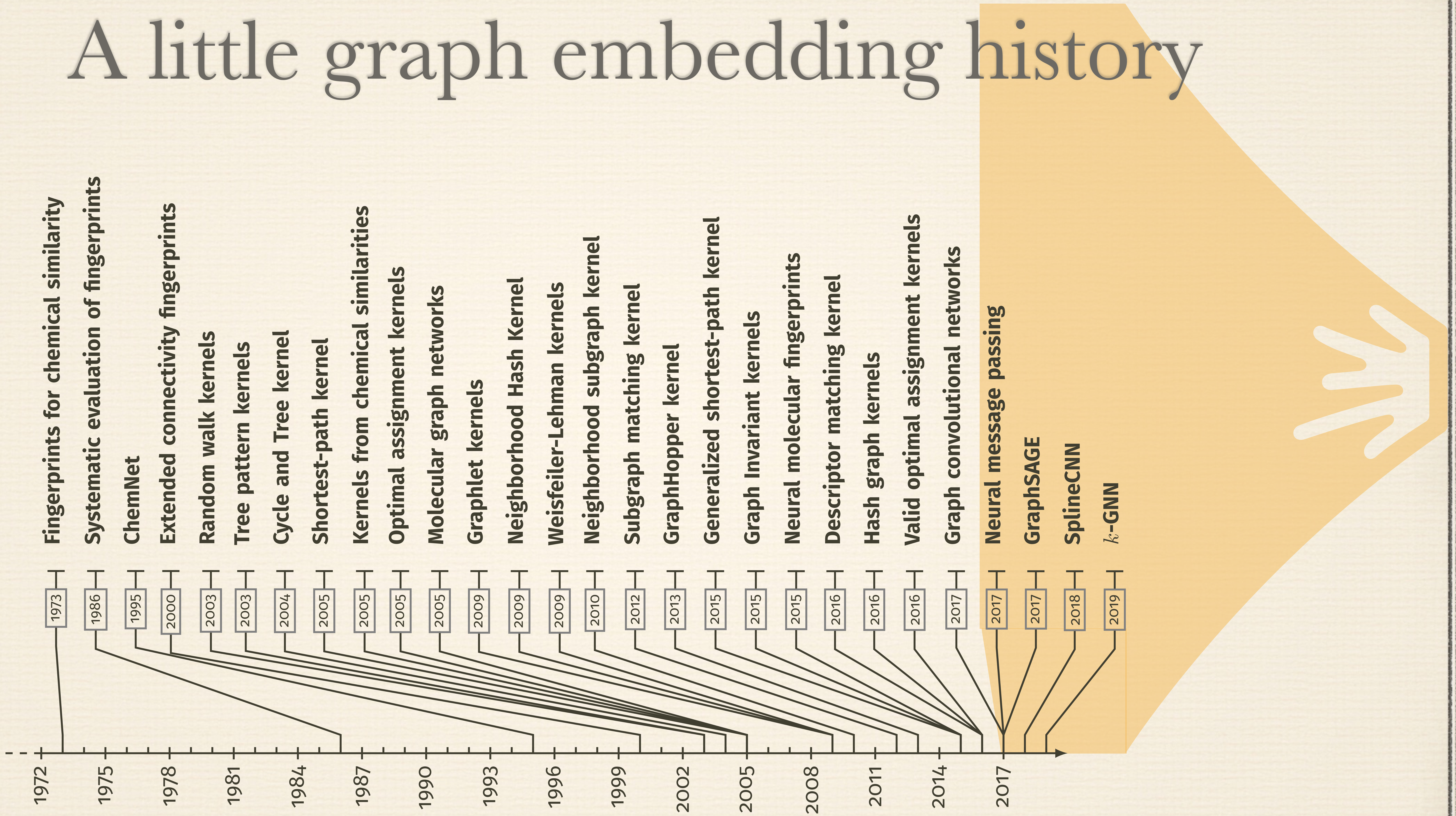
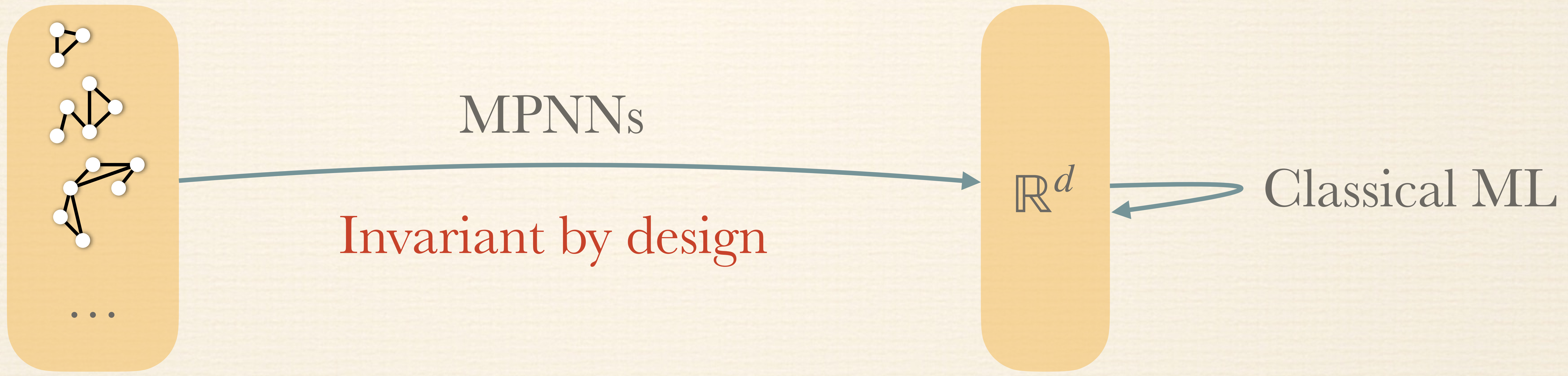


Image: Christopher Morris

Message passing neural networks

A class of invariant vertex and graph embedding methods

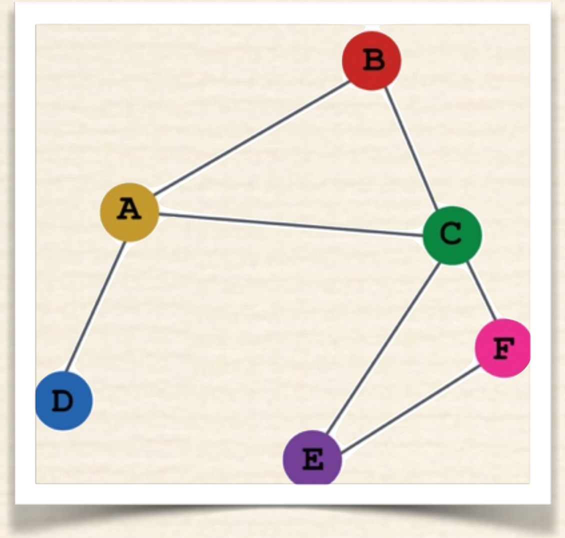


$\mathcal{G} =$ all graphs

$\mathbb{Y} =$ output space

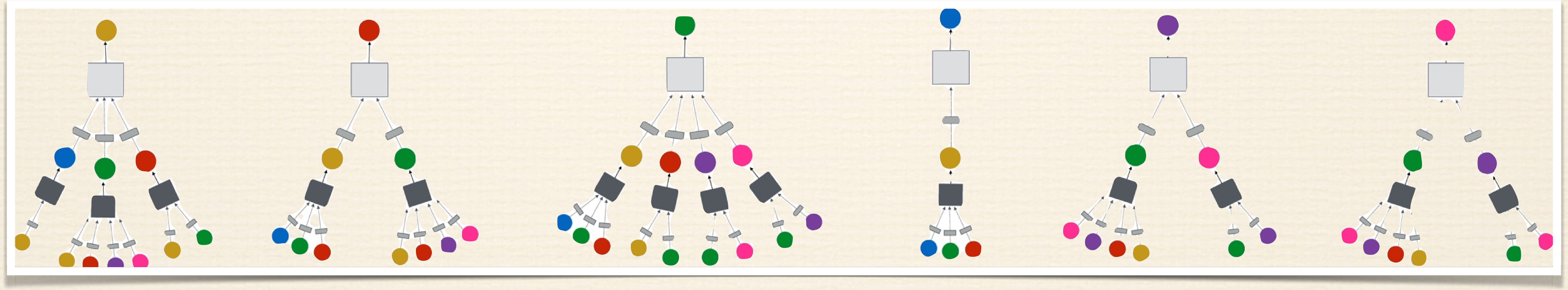
Scarcelli et al.: *The graph neural network model* (2005),
Hamilton et al.: *Inductive representation learning on large graphs* (2017)
Gilmer et al.: *Neural message passing for quantum chemistry* (2017)

Idea behind MPNNs: Neighbour aggregation



Every vertex defines a computation graph

Neural networks



Images: Machine Learning on Graphs, Course by Jure Leskovec

MPNNs: Vertex embedding

$$\xi(G,v) := \xi^{(L)} \circ \xi^{(L-1)} \circ \dots \cdot \circ \xi^{(0)}(G,v)$$

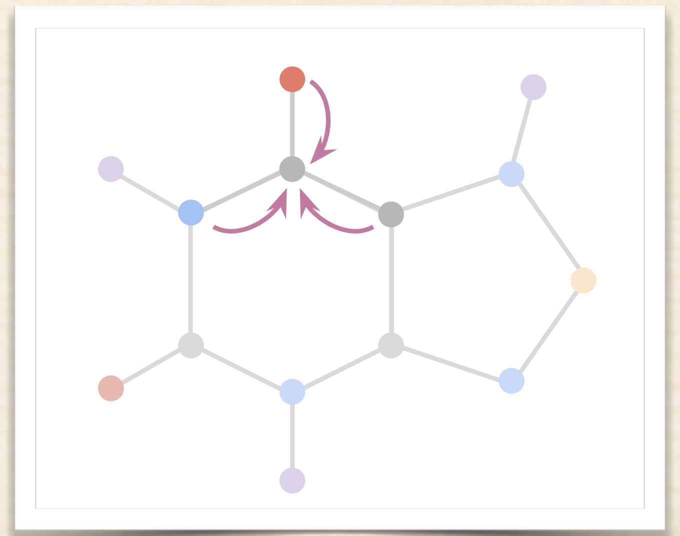
Message Passing Layers $\xi^{(i)}(G,v) \in \mathbb{R}^d$

neighbourhoods

$$\xi^{(0)}(G,v) := \text{Hot-one encoding of label of vertex } v \in \mathbb{R}^d$$

$$\xi^{(t)}(G,v) := \mathsf{Upd}^{(t)}\left(\xi^{(t-1)}(G,v), \mathsf{Agg}^{(t)}\left(\{\{\xi^{(t-1)}(G,v), \xi^{(t-1)}(G,u) \mid u \in N_G(v)\}\}\right)\right) \in \mathbb{R}^d$$

Message Passing between v and its neighbours $u \in N_G(v)$



Update and aggregate function contain learnable parameters (NNs)

MPNNs: Graph embedding

$$\rho(G) := \rho \circ \xi^{(L)} \circ \xi^{(L-1)} \circ \cdots \cdot \circ \xi^{(0)}(G, v)$$

Readout

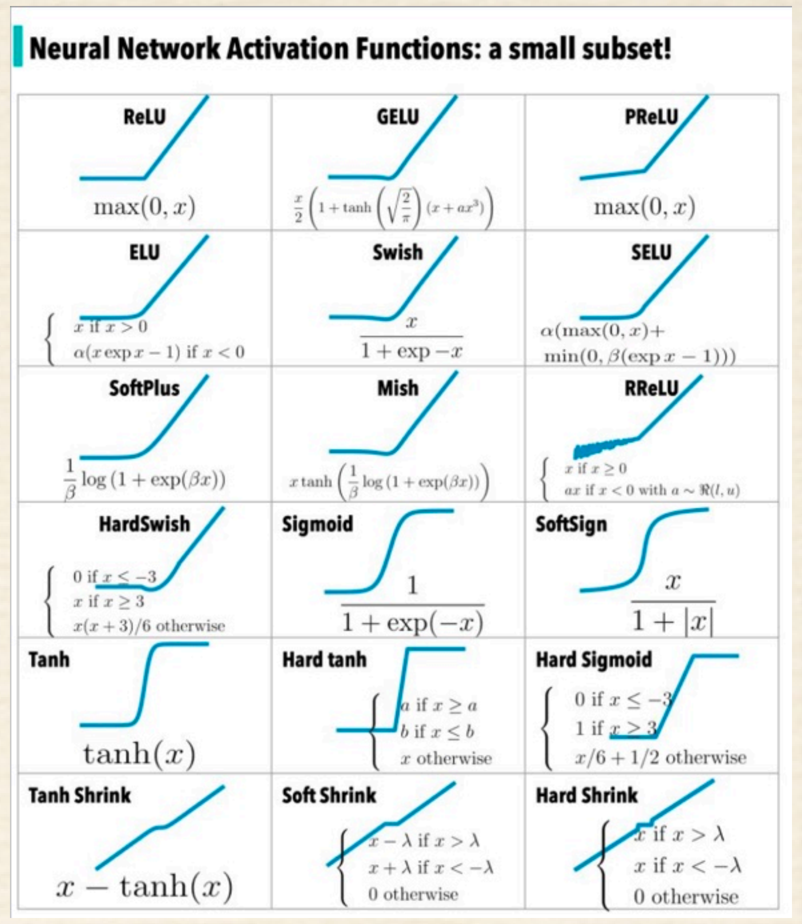
$$\rho(G) := \mathsf{Readout}\Big(\{\{\xi^{(L)}(G, v) \mid v \in V_G\}\}\Big) \in \mathbb{R}^d$$

Has learnable parameters

Aggregation over all vertices

Typical choices for update, aggregate and readout: Multilayer Perceptrons

MPNN example: GNN 101



- Non-linear activation function σ (ReLU, sign, sigmoid, …)
- $\mathbf{F}^{(t)}_{v\bullet} \in \mathbb{R}^d$ denotes embedding of vertex v
- Weight matrices $\mathbf{W}^{(t)}_1 \in \mathbb{R}^{d \times d}$ and $\mathbf{W}^{(t)}_2 \in \mathbb{R}^{d \times d}$ and bias vector $\mathbf{b} \in \mathbb{R}^{1 \times d}$

$$\mathbf{F}^{(0)}_{v\bullet} := L_G(v)$$

← Embedding vertex labels

$$\mathbf{F}^{(t)}_{v\bullet} := \sigma\left(\mathbf{F}^{(t-1)}_{v\bullet}\mathbf{W}^{(t)}_1 + \sum_{u \in N_G(v)} \mathbf{F}^{(t-1)}_{u\bullet}\mathbf{W}^{(t)}_2 + \mathbf{b}^{(t)}\right)$$

Aggregation over neighbours

Matrix form $$\mathbf{F}^{(t)} := \sigma\left(\mathbf{F}^{(t-1)}\mathbf{W}^{(t)}_1 + \mathbf{A}\mathbf{F}^{(t-1)}\mathbf{W}^{(t)}_2 + \mathbf{B}^{(t)}\right)$$

↑ adjacency matrix

Image: TheAiEdge.io

GNN 101: Graph embedding

- Weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ and and bias vector $\mathbf{b} \in \mathbb{R}^{1 \times d}$

$$\mathbf{F}^{(t)} := \sigma\left(\sum_{v \in V_G} \mathbf{F}^{(L)}\mathbf{W} + \mathbf{b}\right)$$

Aggregation over all vertices

ERM: Find best parameters $\mathbf{W}_1^{(1)}, \ldots, \mathbf{W}_1^{(L)}, \mathbf{W}_2^{(1)} \ldots, \mathbf{W}_2^{((L)}, \mathbf{W}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(L)}, \mathbf{b}$

Two more examples of MPNNs

- Graph Isomorphism Networks (GIN)

$$\mathbf{F}^{(t)}_{v\bullet} := \mathsf{MLP}^{(t)}\left((1+\epsilon^{(t)})\mathbf{F}^{(t-1)}_{v\bullet} + \sum_{u \in N_G(v)} \mathbf{F}^{(t-1)}_{u\bullet}\right)$$

- Graph Convolution Network (GCN)

$$\mathbf{F}^{(t)}_{v\bullet} := \mathsf{MLP}^{(t)}\left(\frac{1}{\sqrt{|N_G(v)|+1}} \sum_{u \in N_G(v) \cup \{v\}} \frac{1}{\sqrt{|N_G(u)|+1}} \mathbf{F}^{(t-1)}_{u\bullet}\right)$$

GIN: Xu et al.: *How powerful are graph neural networks?* (2019)
GCN: Kipf and Welling: *Semi-supervised classification with graph convolutional networks* (2017)

MPNNs: Expressive power

What is ρ(MPNNs)?

Recall: All pairs of graphs (G, H) such that all MPNNs return same graph embedding on both graphs.

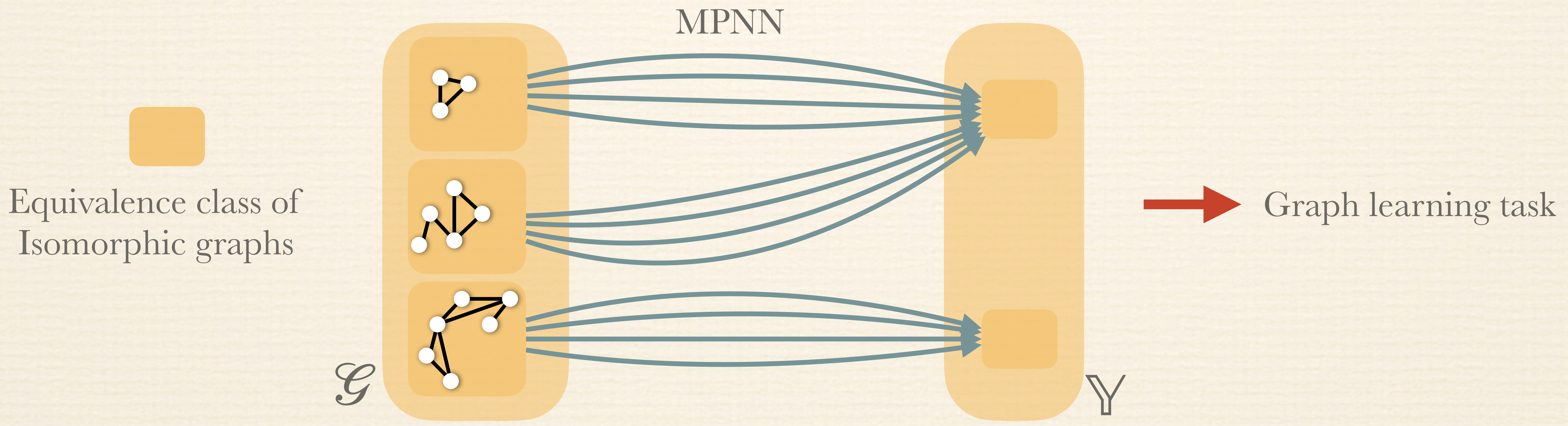
Understanding ρ(MPNNs) translates in understanding power of GNN 101, GCNs, GINs, ….

A short detour to graph isomorphism testing

MPNNs and isomorphic graphs

- Because of invariance: MPNNs embed isomorphic graphs in the same way. That is, if $G \cong H \Rightarrow (G, H) \in \rho(\text{MPNN})$

- Can MPNNs embed non-isomorphic graphs differently?



The graph isomorphism problem

Given two graph $G = (V_G, E_G, L_G)$ and $H = (V_H, E_H, L_H)$: are they isomorphic? Or is $G \cong H$?

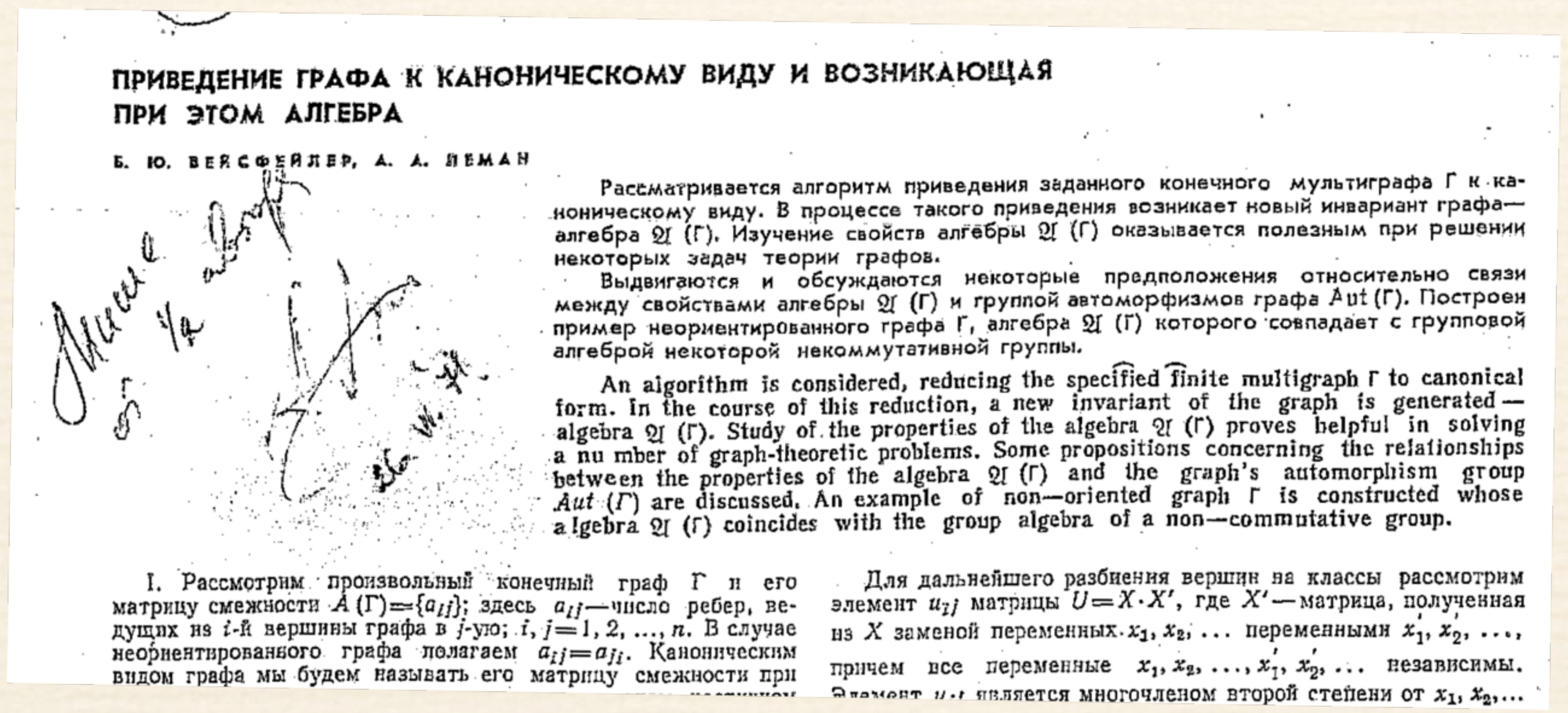
- Does there exist a graph isomorphism $\pi : V_G \to V_H$?
- Theory: computational complexity is open.
- Quasi-polynomial algoritm $n^{\log(n)^{\mathcal{O}(1)}}$ by László Babai (2016).
- Practice: very fast tests.

L. Babai: *Graph isomorphism in quasipolynomial time* (2016)

One-sided test: Colour refinement

Apply heuristic on G and H: If Heuristic say "no" then $G \ncong H$, otherwise we do not know.

- Common heuristic is colour refinement
- In a paper by Boris Weisfeiler and Andrei Leman (1968)



ПРИВЕДЕНИЕ ГРАФА К КАНОНИЧЕСКОМУ ВИДУ И ВОЗНИКАЮЩАЯ ПРИ ЭТОМ АЛГЕБРА

Б. Ю. ВЕЙСФЕЙЛЕР, А. А. ЛЕМАН

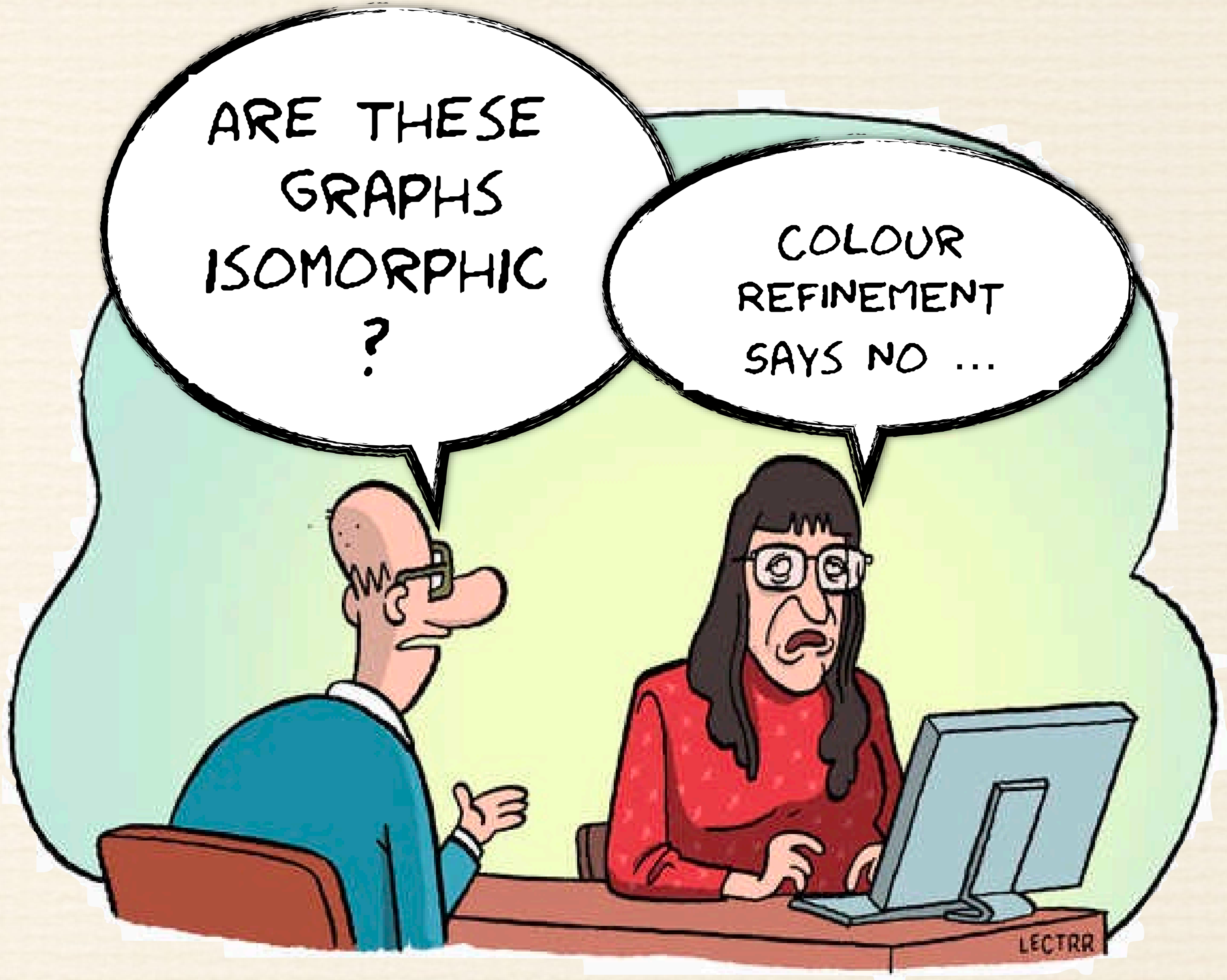
Рассматривается алгоритм приведения заданного конечного мультиграфа Γ к каноническому виду. В процессе такого приведения возникает новый инвариант графа—алгебра $\mathfrak{A}$ (Γ). Изучение свойств алгебры $\mathfrak{A}$ (Γ) оказывается полезным при решении некоторых задач теории графов.

Выдвигаются и обсуждаются некоторые предположения относительно связи между свойствами алгебры $\mathfrak{A}$ (Γ) и группой автоморфизмов графа Aut (Γ). Построен пример неориентированного графа Γ, алгебра $\mathfrak{A}$ (Γ) которого совпадает с групповой алгеброй некоторой некоммутативной группы.

An algorithm is considered, reducing the specified finite multigraph Γ to canonical form. In the course of this reduction, a new invariant of the graph is generated—algebra $\mathfrak{A}$ (Γ). Study of the properties of the algebra $\mathfrak{A}$ (Γ) proves helpful in solving a nu mber of graph-theoretic problems. Some propositions concerning the relationships between the properties of the algebra $\mathfrak{A}$ (Γ) and the graph's automorphism group *Aut* (Γ) are discussed. An example of non—oriented graph Γ is constructed whose a lgebra $\mathfrak{A}$ (Γ) coincides with the group algebra of a non—commutative group.

I. Рассмотрим произвольный конечный граф Γ и его матрицу смежности $A(\Gamma)=\{a_{ij}\}$; здесь a_{ij}—число ребер, ведущих из i-й вершины графа в j-ую; $i, j=1, 2, \ldots, n$. В случае неориентированного графа полагаем $a_{ij}=a_{ji}$. Каноническим видом графа мы будем называть его матрицу смежности при

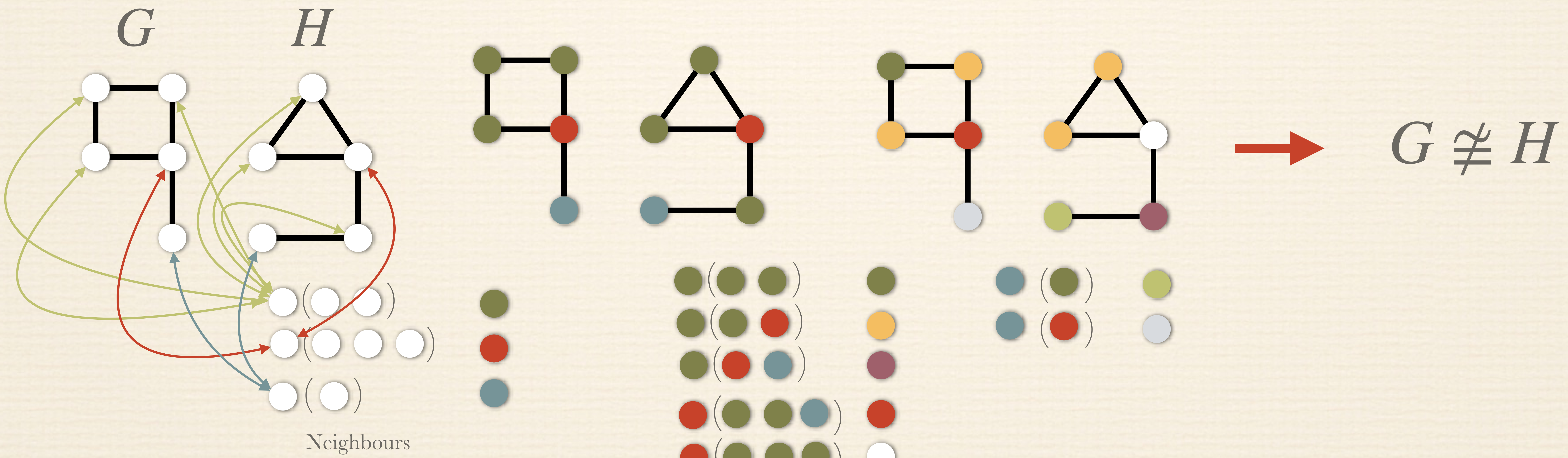
Для дальнейшего разбиения вершин на классы рассмотрим элемент u_{ij} матрицы $U=X\cdot X'$, где X'—матрица, полученная из X заменой переменных $x_1, x_2, \ldots$ переменными $x_1', x_2', \ldots$, причем все переменные $x_1, x_2, \ldots, x_1', x_2', \ldots$ независимы. Элемент u_{ij} является многочленом второй степени от $x_1, x_2, \ldots$



B. Weisfeiler and A. Leman. *The reduction of a graph to canonical form and the algebra which appears therein (1968)*

Colour refinement

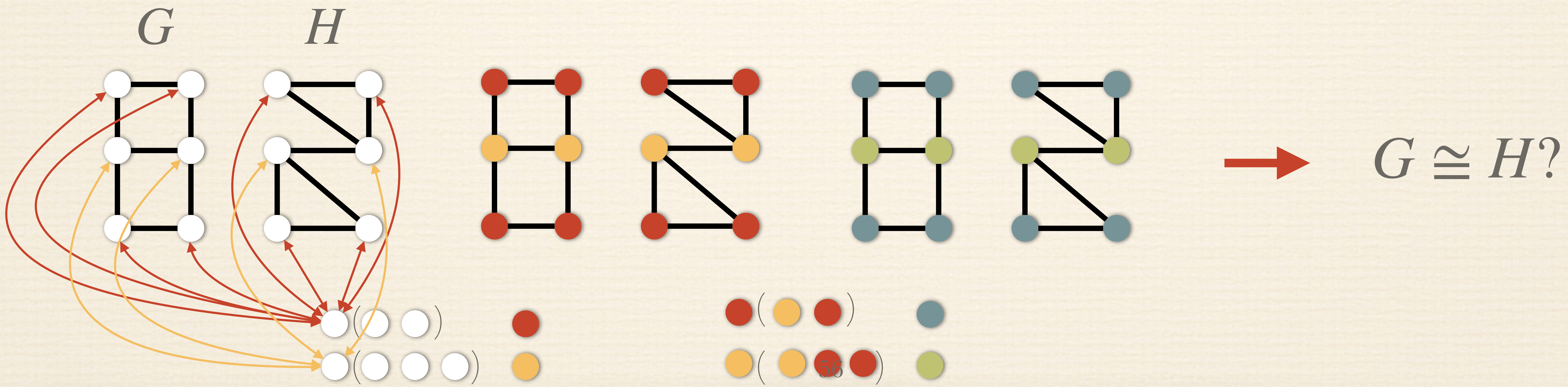
- Initial: All vertices have their original colour (label)
- Iteration: Separation of identically coloured vertices based on colour histograms of neighbours.
- Two graphs are non-isomorphic if they have different colour histograms.



Colour refinement

- Initial: All vertices have their original colour (label)
- Iteration: Separation of identically coloured vertices based on colour histograms of neighbours.
- Two graphs are non-isomorphic if they have different colour histograms.

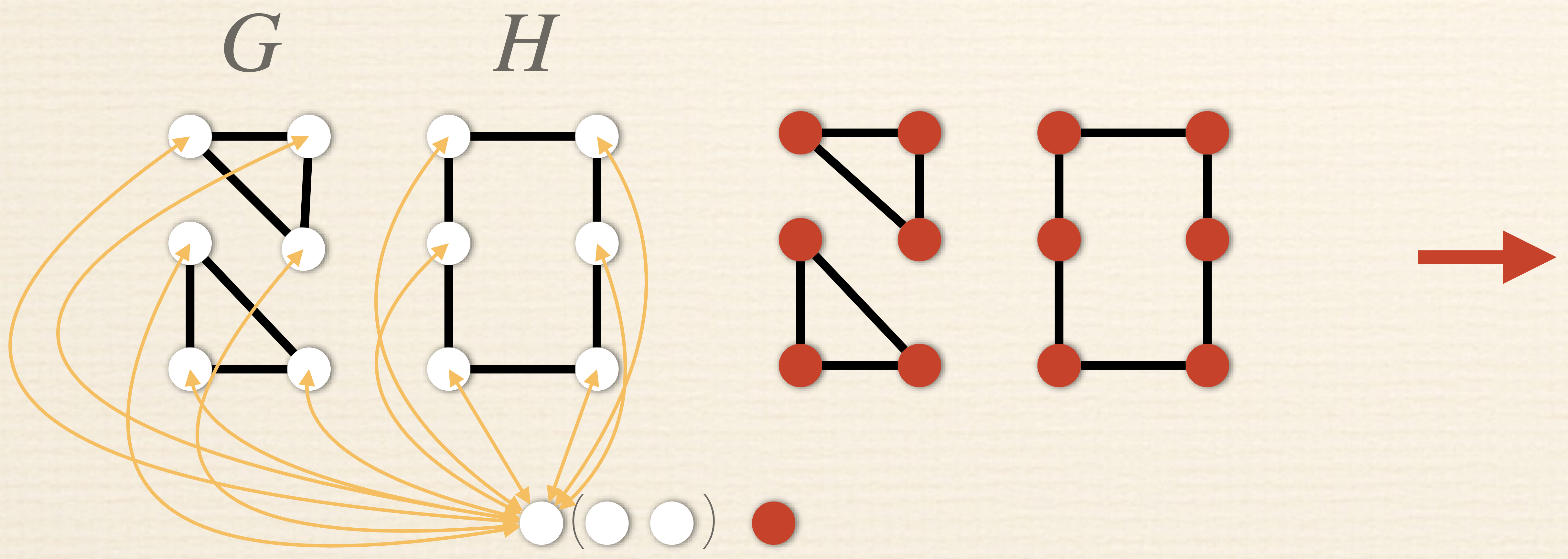
Stops when colour partition does not change (max n iterations)



Colour refinement

- Initial: All vertices have their original colour (label)
- Iteration: Separation of identically coloured vertices based on colour histograms of neighbours.
- Two graphs are non-isomorphic if they have different colour histograms.

Stops when colour partition does not change (max n iterations)



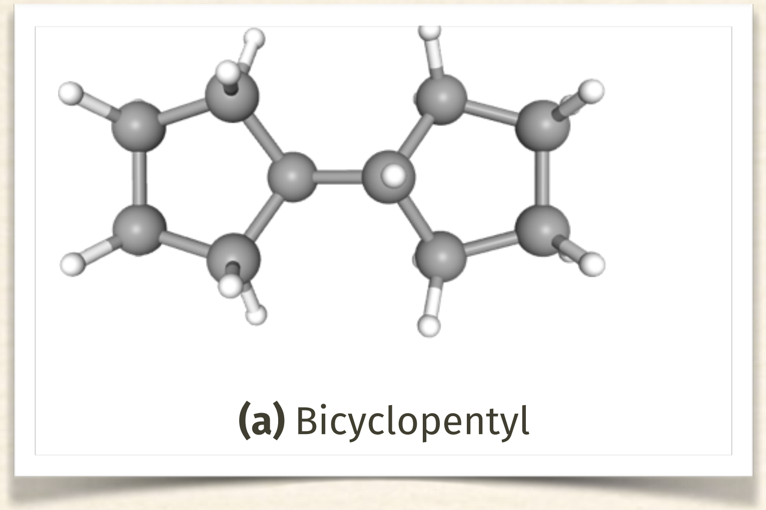
$G \cong H$?

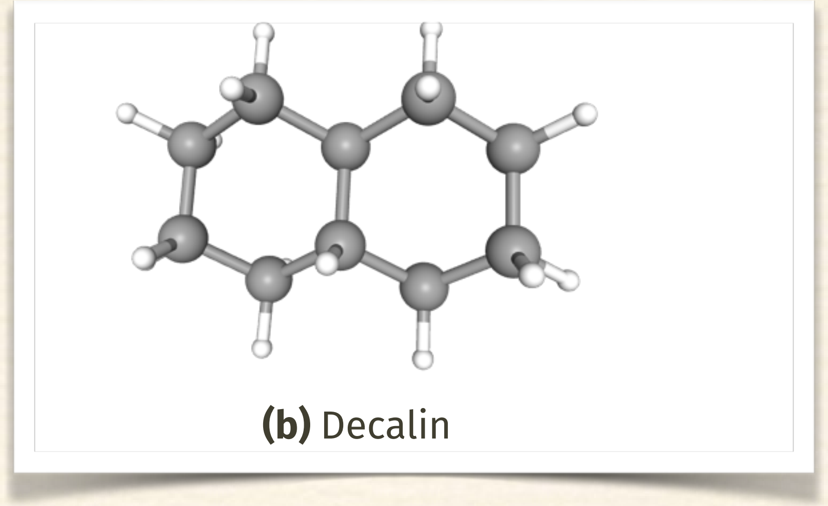
Colour refinement

- Extensively studied in the theoretical computer science community
- Many different characterisations of when two graphs have the same colour histograms (equivalent for colour refinement).
- Successful on random graphs with high probability
- Weak expressive (distinguishing) power

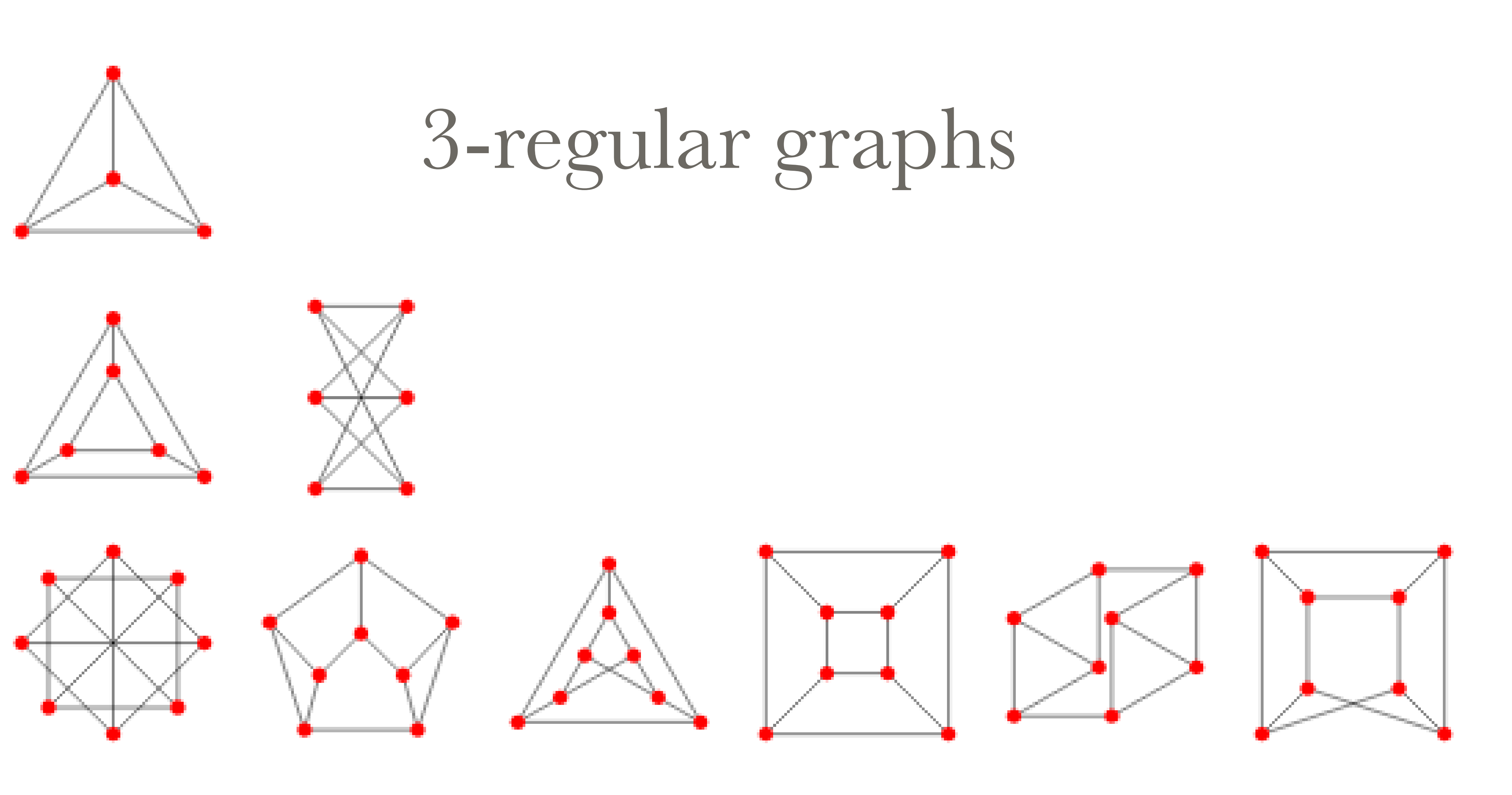
L. Babai and L. Kucera. *Canonical labelling of graphs in linear average time* (1979)
Cai et al.: *An optimal lower bound on the number of variables for graph identifications.* (1992)
Arvind et al.: *On the power of color refinement* (2015)
M. Grohe: *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory* (2017)
Arvind et al.: *On WL invariance: Subgraph Counts and related properties* (2019)
M. Grohe. *The logic of graph neural networks* (2021)

ρ(colour refinement)





- Cannot count cycles (triangles)
- Cannot distinguish d-regular graphs
- Only tree information



Back to MPNNs

Arvind et al.: *On the power of color refinement* (2015)
Images: Wolfram MathWorld, Christopher Morris

MPNNs & Colour refinement

Theorem (Morris et al. 2019, Xu et al. 2019)

If colour refinement cannot tell two graphs apart then neither can any MPNN!

MPNNs

$\xi^{(0)}(G, v) :=$ Hot-one encoding of label of vertex v

$\xi^{(t)}(G, v) := \mathsf{Upd}^{(t)}\left(\xi^{(t-1)}(G, v), \mathsf{Agg}^{(t)}\left(\{\{\xi^{(t-1)}(G, v), \xi^{(t-1)}(G, u) \mid u \in N_G(v)\}\}\right)\right)$

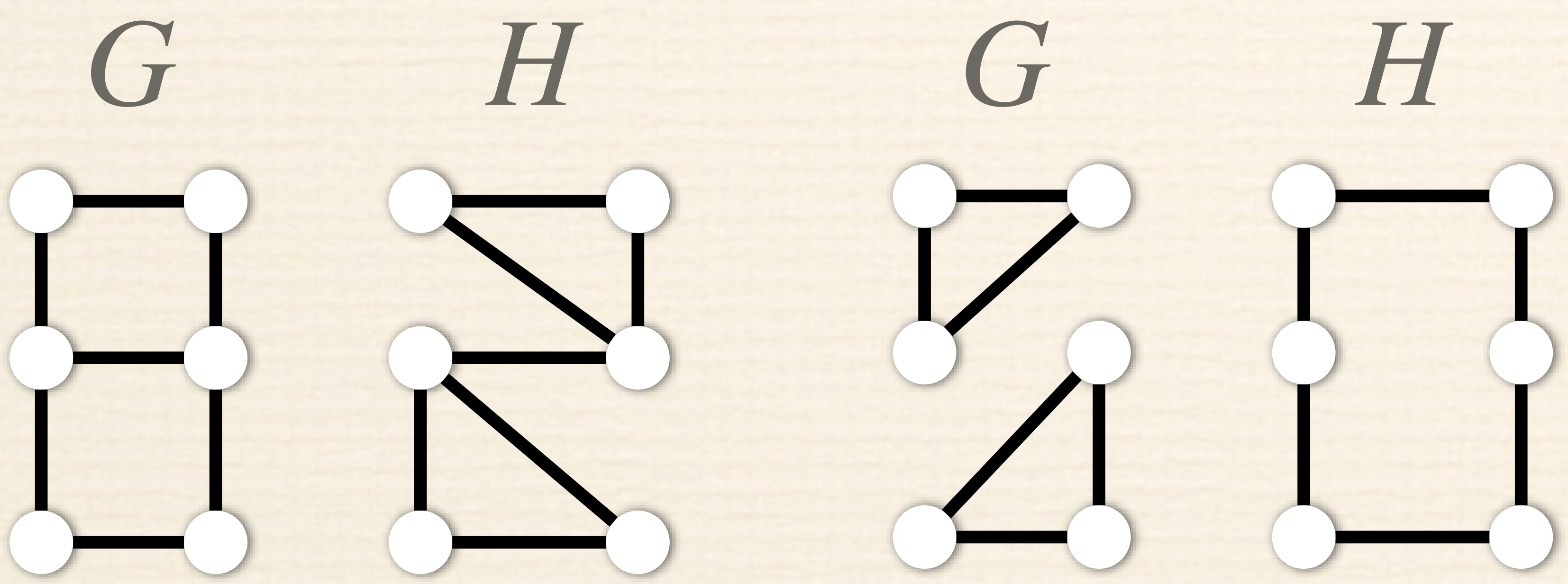
$\rho(G) := \mathsf{Readout}\left(\{\{\xi^{(L)}(G, v) \mid v \in V_G\}\}\right)$

Color refinement

$\mathsf{cr}^{(0)}(G, v) :=$ Initial label of v

$\mathsf{cr}^{(t)}(G, v) := \mathsf{Hash}\left(\mathsf{cr}^{(t-1)}(G, v), \{\{\mathsf{cr}^{(t-1)}(G, u) \mid u \in N_G(v)\}\}\right)$

$\rho(G) := \{\{\mathsf{cr}(G, v) \mid v \in V_G\}\}$

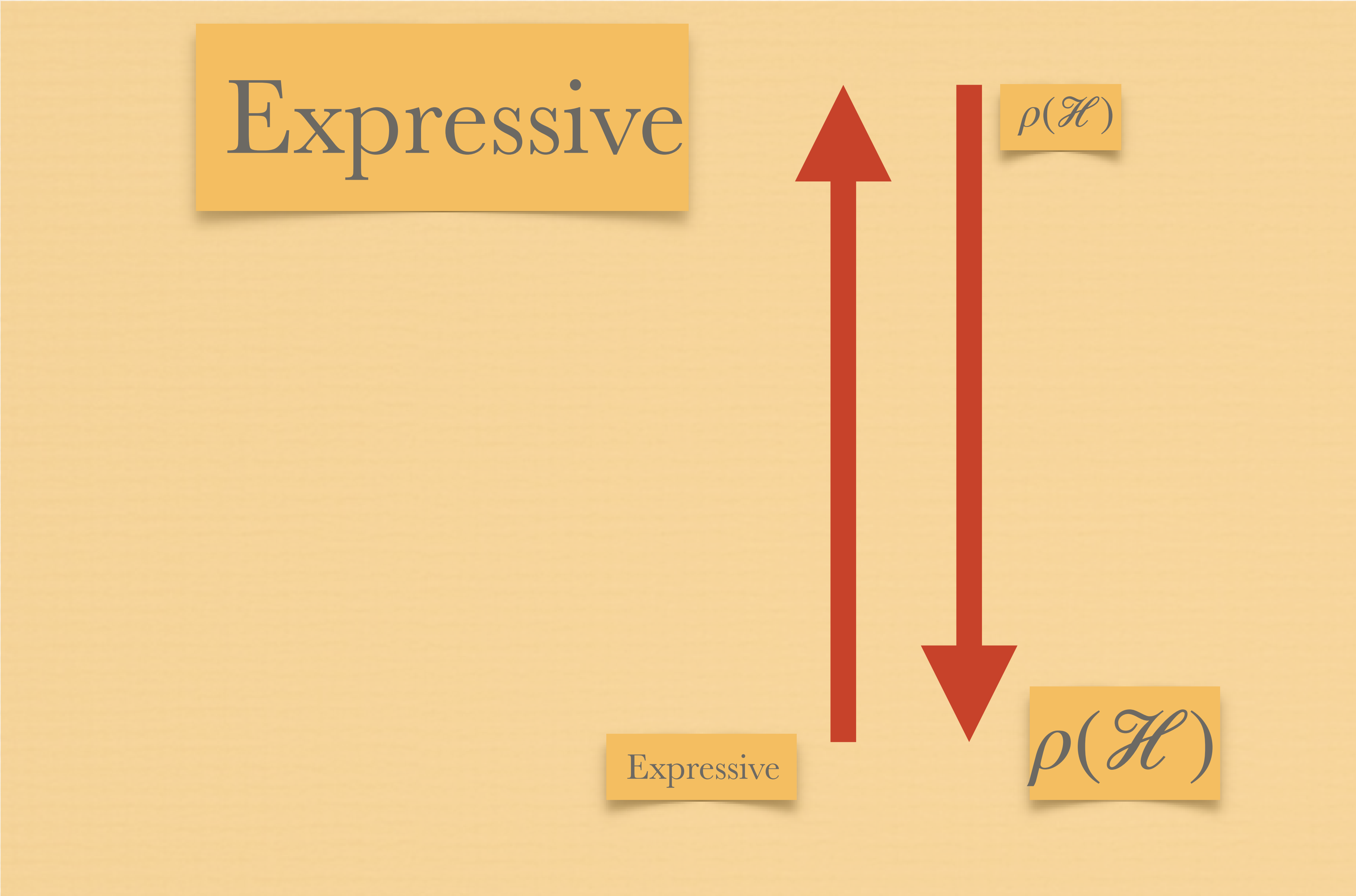


→ No MPNN can separate these graphs

Xu et al.: *How powerful are graph neural networks?* (2019)
Morris et al: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019)

MPNNs & Colour refinement

Recall:



We have just shown: $\rho(\text{colour refinement}) \subseteq \rho(\text{MPNNs})$

Expressive power of MPNNs is upper bounded by colour refinement

Lower bound?

- We have seen that MPNNs cannot separate more graphs than colour refinement.

- Can colour refinement separate more graphs than MPNNs? No!

Theorem (Morris et al. 2019)

There exists a GNN 101 which can embed G and H differently when colour refinement assigns them different colours

- The class of MPNNs is as powerful (or weak) as colour refinement

Morris et al: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019)

What else can we say?

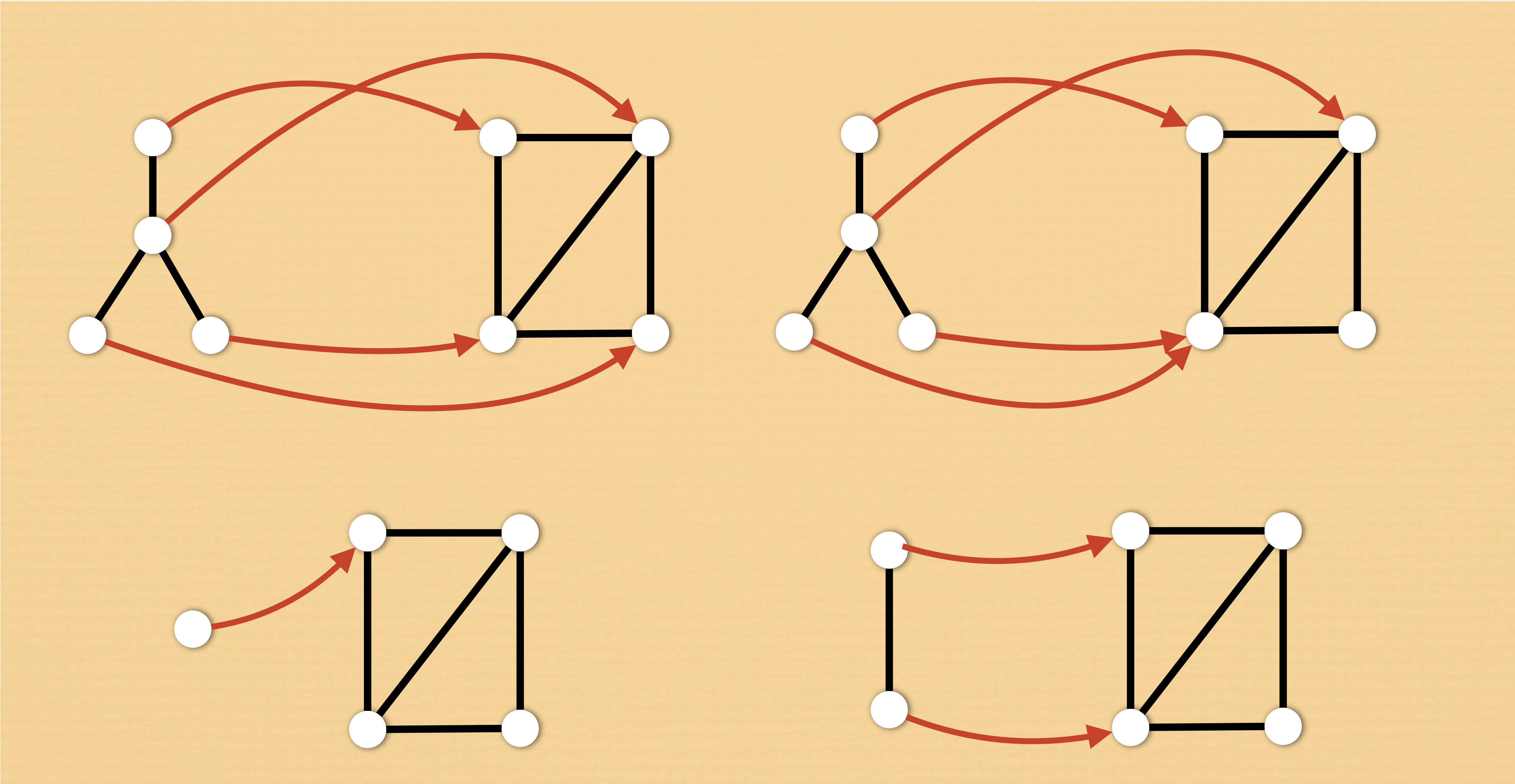
$\rho(\text{colour refinement}) = \rho(\text{MPNNs})$

Other - more insightful - characterisations?

A detour to homomorphism counts

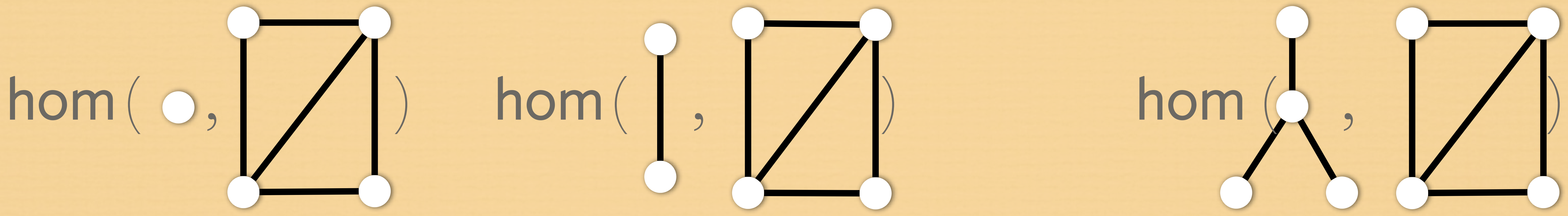
Homomorphisms

- Let $P = (V_P, E_P, L_P)$ and $G = (V_G, E_G, L_G)$ be graphs.
- A function $h : V_P \to V_G$ is a homomorphism if it is edge preserving $(v, w) \in E_p \Rightarrow (h(v), h(w)) \in E_G$ and label preserving.



Homomorphism counts

- Define $\mathsf{HOM}(P, G)$:= { all homomorphisms from P to G}
- Define $\mathsf{hom}(P, G) := |\mathsf{HOM}(P, G)|$.



$\#\text{vertices} = 4$ $\qquad 2\#\text{edges}=10$ $\qquad 70 = 2 \cdot 2^3 + 2 \cdot 3^3$

Homomorphisms

- Weaker notion than subgraph isomorphism (see later).
- Underlies semantics of many graph query languages.
- Algebra of homomorphism counts: A rich and active area of research.

Back to MPNNs

MPNNs and hom counts

Theorem (Dell et al. 2019, Dvorák 2010)

$\mathsf{hom}(T, G) = \mathsf{hom}(T, H)$ for all *trees* T
if and only if
colour refinement cannot distinguish G from H.

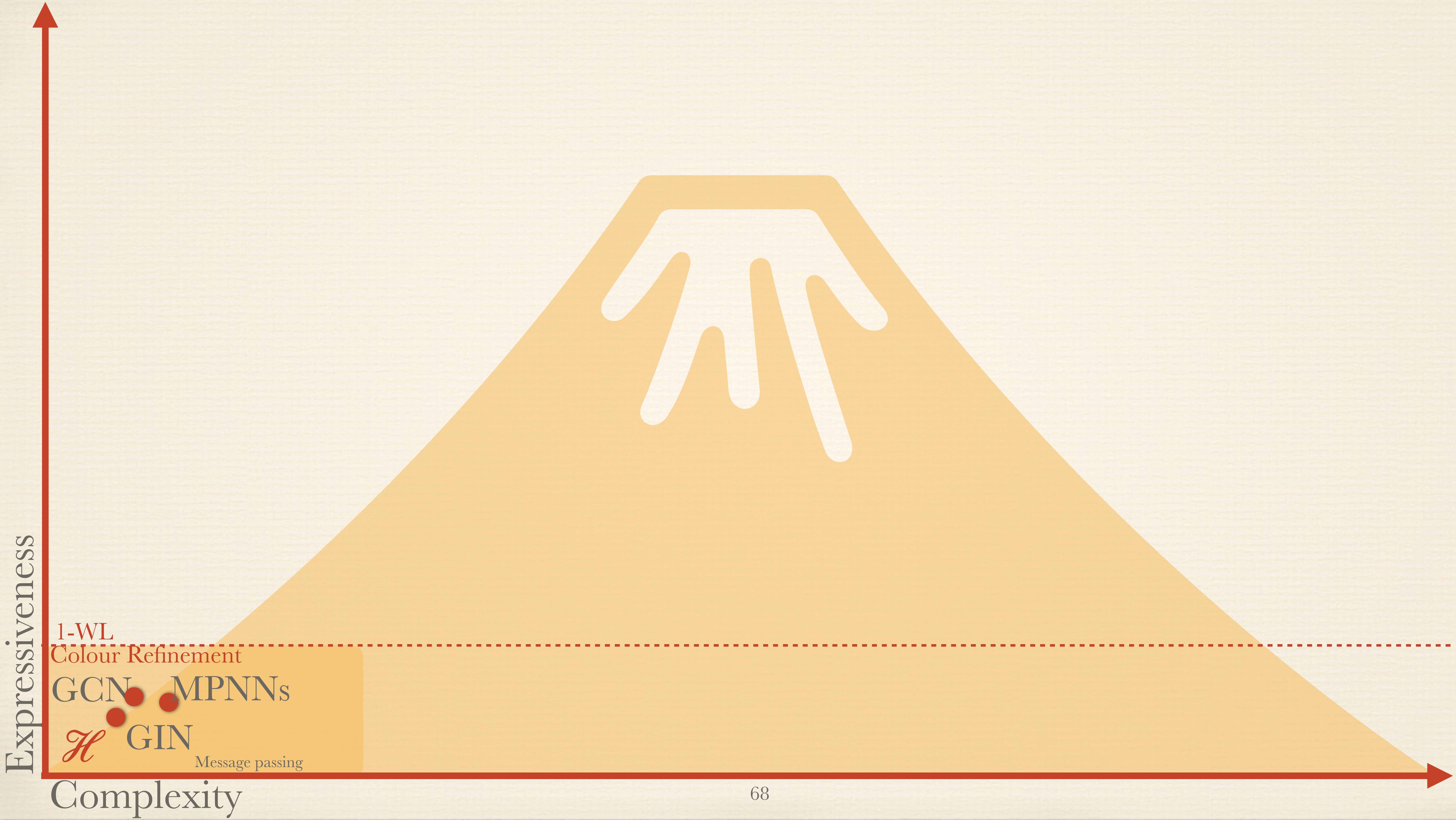
Corollary

$\mathsf{hom}(T, G) = \mathsf{hom}(T, H)$ for all *trees* T *if and only if* no MPNN can distinguish G from H.

Follows from $\rho(\text{cr}) = \rho(\text{MPNN})$

- MPNNs can only detect tree information from a graph!

Z. Dvoräk: *On recognizing graphs by numbers of homomorphisms* (2010)
Dell et al. *Lovász meets Weisfeiler and Leman* (2018)


Expressiveness
1-WL
Colour Refinement
GCN
MPNNs
𝓗
GIN
Message passing
Complexity

Beyond distinguishing power?

- Approximation properties (universality)
- Logical expressiveness
- Generalization

Approximation properties

- Equip set of graphs $\mathscr{G}$ with a topology and assume that $\mathscr{H}$ consists of continuous graph embeddings from $\mathscr{G}$ to $\mathbb{R}$.

- Let $\mathscr{C} \subseteq \mathscr{G}$ be a compact set of graphs.

Stone-Weierstrass

Theorem (Azizian & Lelarge 2021, G. and Reutter 2022)

If $\mathscr{H}$ is closed under linear combinations and products, then $\mathscr{H}$ can approximate any continuous function $\Xi : \mathscr{C} \to \mathbb{R}$ satisfying

$$\rho(\mathscr{H}) \subseteq \rho(\{\Xi\}) .$$

- Can be generalised to general embeddings with output space $\mathbb{R}^d$

W. Azizian and M. Lelarge: *Characterizing the expressive power of invariant and equivariant graph neural networks* (2021)
G. and J. Reutter: *Expressiveness and approximation properties of graph neural networks* (2022)

MPNNs: Approximation

Theorem (Azizian & Lelarge 2021, G. and Reutter 2022)

On compact set of graphs, MPNNs can approximate any continuous graph embedding $\Xi : \mathscr{C} \to \mathbb{R}$ satisfying
$\rho(\text{colour refinement}) \subseteq \rho(\{\Theta\})$

- We know $\rho(\text{MPNNs}) = \rho(\text{colour refinement})$
- Update functions can be used to approximate product and take linear combinations of MPNNs

- Intricate relation between distinguishing power and approximation properties

W. Azizian and M. Lelarge: *Characterizing the expressive power of invariant and equivariant graph neural networks* (2021)
G. and J. Reutter: *Expressiveness and approximation properties of graph neural networks* (2022)

Universality and graph isomorphism

Theorem (Chen et al. (2019)

In order for a class of methods to be able to approximate any (invariant) continuous functions, the class of methods should be able to distinguish any two non-isomorphic graphs.

Proof

Minimal size $\rho(\mathscr{H}) \subseteq \rho(\{\Xi\})$

↑

$(G, H) \in \rho(\mathscr{H}) \Leftrightarrow G \cong H$

Chen et al.: *On the equivalence between graph isomorphism testing and function approximation with GNNs* (2019)

Logical expressiveness

- Finite variable logics.
- Extension with Presburger quantifiers.

Colour refinement (again)

I mentioned that ρ(colour refinement) has many characterisations.

Of interest is also a logical one, in particular First-order logic with 2 variables and counting quantifiers (C_2).

$$\varphi(x) = \exists^{\leq 5} y \left(E(x,y) \wedge \exists^{\geq 2} x \left(E(y,x) \wedge L_a(x) \right) \right)$$

binary edge predicate   unary label predicate

Given graph G, vertex $v \in V_G$ satisfies φ: It has at most 5 neighbours
$(G, v) \vDash \varphi$ each with at least to neighbours labeled "a"

Cai et al.: *An optimal lower bound on the number of variables for graph identifications.* (1992)
M. Grohe. *The logic of graph neural networks* (2021)

Colour refinement and C_2

Theorem (Cai et al. 1992)

Two graph shave the same colour histogram after t iterations of colour refinement *if and only if* they satisfy the same C_2 sentences of quantifier depth t

$$\rho(\text{colour refinement}) = \rho(\text{MPNNs}) = \rho(\mathsf{C}_2)$$

What about vertices?

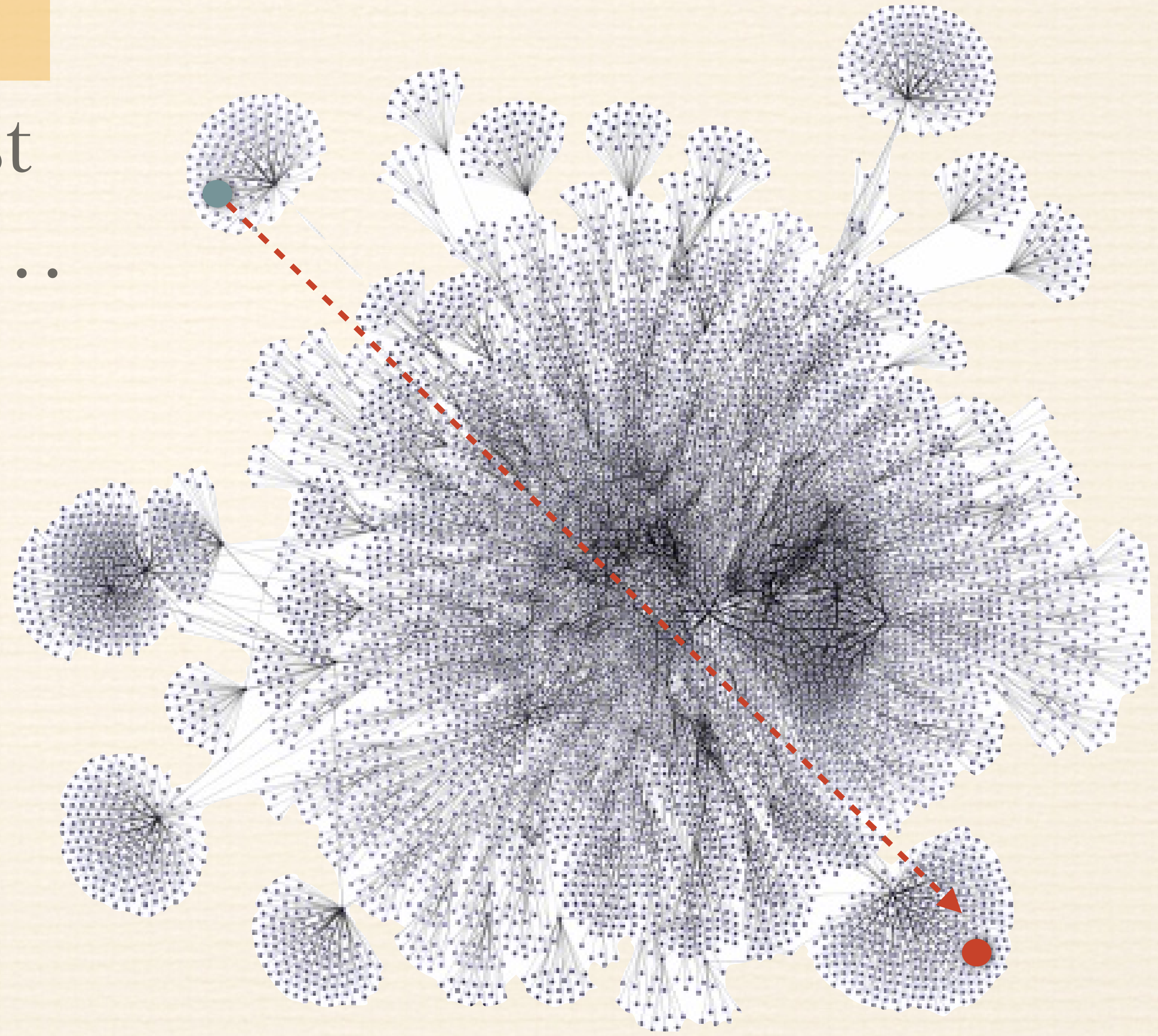
Cai et al.: *An optimal lower bound on the number of variables for graph identifications.* (1992)
M. Grohe. *The logic of graph neural networks* (2021)

Which unary C_2 formulas can MPNNs express?

$\mathscr{H}$ can $\mathscr{C}$-express Ξ if there exists a $\xi \in \mathscr{H}$ such that <u>for all</u> $G \in \mathscr{C}, \mathbf{v} \in V_G^p : \xi(G, \mathbf{v}) = \Xi(G, \mathbf{v})$

- <u>Not all:</u> $\varphi(x) := L_b(x) \wedge \exists y L_r(y)$

  I am blue and there exist a red vertex somewhere…



Cannot be reached by message passing!

Which unary C_2 formulas can MPNNs express?

- Not all: $\varphi(x) := L_b(x) \wedge \exists y L_r(y)$
- Graded modal logic: syntactical fragment of C_2 in which quantifiers are of the form $\exists^{\geq N}\big(E(x,y) \wedge \varphi'(y)\big)$

Theorem (Barceló et al. 2020)

Let $\varphi(x)$ be a unary FO formula. Then, $\varphi(x)$ is equivalent to a graded modal logic formula *if and only if* $\varphi(x)$ is expressible by the class of MPNNs.

$$\exists \xi \in \text{MPNNs} : \forall G \in \mathcal{G}, \forall v \in V_G : (G, v) \vDash \varphi \Leftrightarrow \xi(G, v) = 1$$

Barceló et al.: *The logical expressiveness of graph neural networks* (2020)

Role of activation functions

Theorem

Let $\varphi(x)$ be a unary FO formula. Then, $\varphi(x)$ is equivalent to a graded modal logic formula *if and only if* $\varphi(x)$ is expressible by the class of MPNNs.

- Proof relies on sign, ReLU, trReLU activation function.

Theorem (Sammy Khalife 2023)

There is a $\varphi(x)$ in graded modal logic that is not expressible by MPNNs using *polynomial activation functions.*

Sammy Khalife. *GNNs with polynomial activation functions have limited expressivity (202*

MPNN+: Extended MPNNs

- Can we extend MPNNs such that all C_2 formulas (including $\varphi(x) := L_b(x) \wedge \exists y L_r(y)$) can be expressed?

$$\xi^{(t)}(G,v) := \mathsf{Upd}^{(t)}\Big(\xi^{(t-1)}(G,v), \mathsf{Agg}^{(t)}\big(\{\{\xi^{(t-1)}(G,v), \xi^{(t-1)}(G,u) \mid u \in N_G(v)\}\}\big)\Big)$$

Add global aggregation in every layer

$$\xi^{(t)}(G,v) := \mathsf{Upd}^{(t)}\Big(\xi^{(t-1)}(G,v), \mathsf{Agg}^{(t)}\big(\{\{\xi^{(t-1)}(G,v), \xi^{(t-1)}(G,u) \mid u \in N_G(v)\}\}\big), \mathsf{Global}^{(t)}\big(\{\{\xi^{(t-1)}(G,u) \mid u \in V_G\}\}\big)\Big)$$

Barceló et al.: *The logical expressiveness of graph neural networks* (2020)

MPNNs+

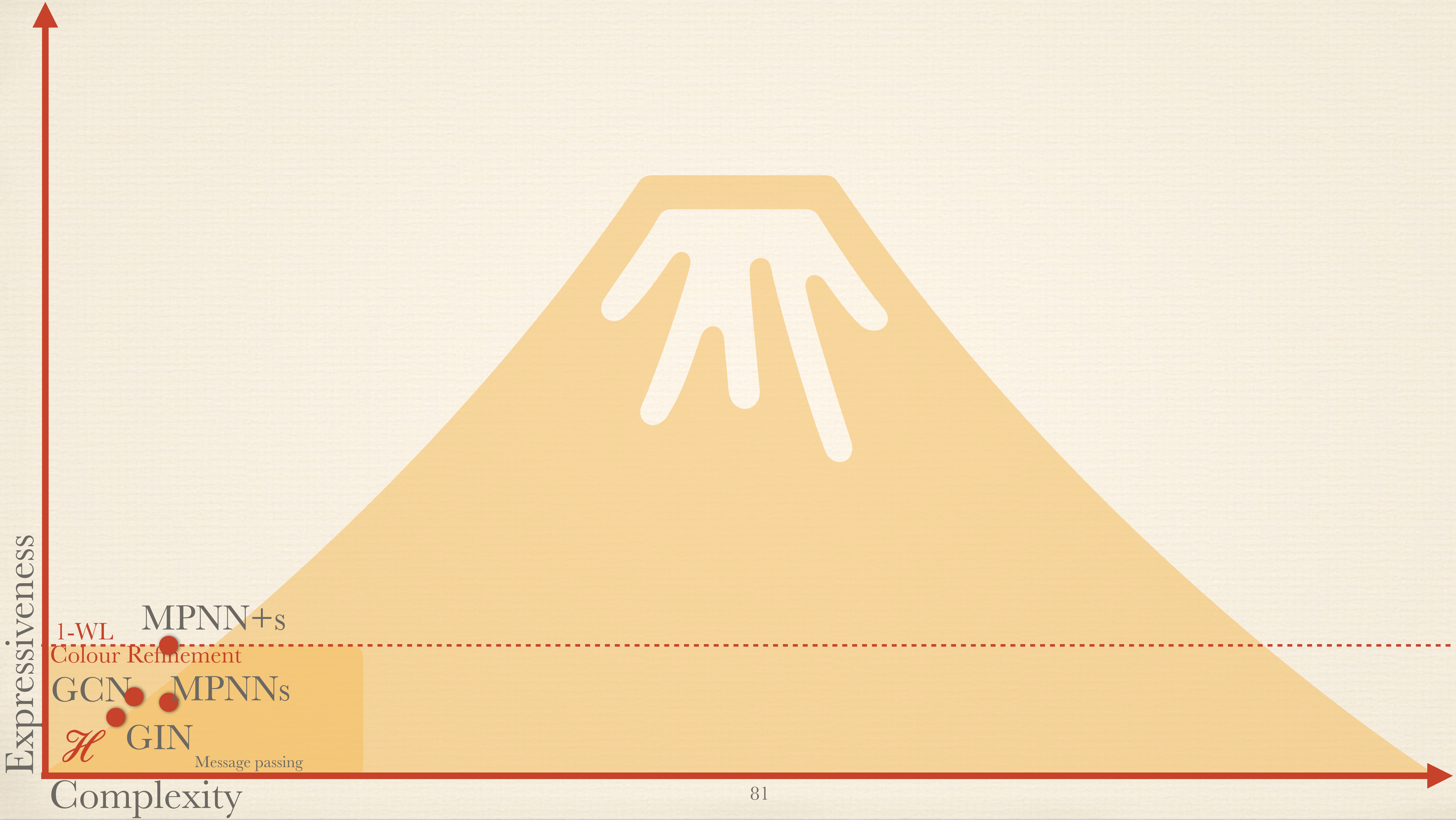
Theorem (Barceló et al. 2020)

Every unary C_2 formula $\varphi(x)$ is expressible by the class of MPNNs+

- The corresponding colour refinement version is known as the one-dimensional Weisfeiler-Leman algorithm or 1-WL on vertices.

$$\rho(\text{1-WL}) = \rho(\text{MPNNs+})$$

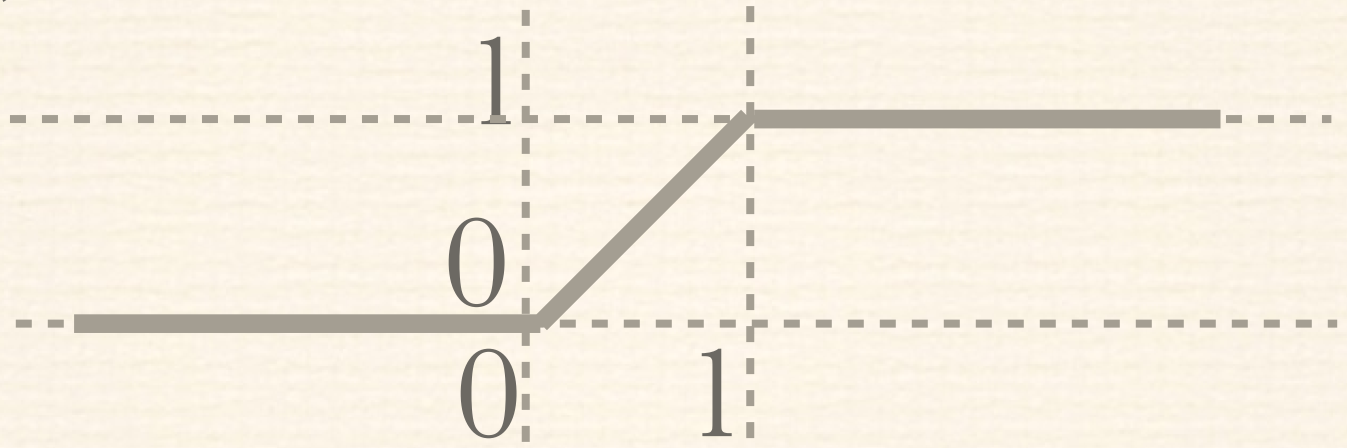
Barceló et al.: *The logical expressiveness of graph neural networks* (2020)



Expressiveness
MPNN+s
1-WL
Colour Refinement
GCN
MPNNs
$\mathcal{H}$
GIN
Message passing
Complexity

Wait a moment! MPNNs go easily beyond FO!

$$\text{trRelU}\Big(\sum_{u\in N_G(v)} P_r(u) - \sum_{u\in N_G(v)} P_b(u)\Big) = \begin{cases} 1 & v \text{ had more red than blue neighbors} \\ 0 & \text{otherwise} \end{cases}$$



This property is known not to be expressible as an FO formula $\varphi(x)$

Proof

By means of *locality* of FO
or
By playing so-called *Erhenfeucht-Fraisse* game.

L. Libkin. *Elements of Finite Model Theory (2004)*

Solution? Add more complex quantifiers!

$$\varphi(x) = \exists^{\geq 2} y \left(E(x,y) \wedge L_r(y) \right)$$ → $$\varphi(x) = (\#_y[E(x,y) \wedge L_r(y)] \geq 2)$$

true when x has more than two red neighbours

$$\varphi(x) = (\#_y[E(x,y) \wedge L_r(y)] - \#_y[E(x,y) \wedge L_b(y)] \geq 0)$$

true when x has more red than blue neighbours

$$\varphi(x) = \left(\sum_{i=1}^{k} a_i \#_y[E(x,y) \wedge \psi_i(y)] \leq \delta \right)$$

true when the neighbours of x satisfy the linear inequality

Presburger quantifier

Michael Benedikt, Chia-Hsuan Lu, Boris Motik, and Tony Tan. Decidability of graph neural networks via logical characterizations (2024)

Local and Global

$\varphi(x) = (\#_y[E(x,y) \wedge L_r(y)] \geq 2)$

true when x has more than two red neighbours

$\varphi(x) = (\#_y[L_r(y)] - \#_y[E(x,y) \wedge L_r(y)] = 0)$

true when x has all red nodes in graph as neighbours

global counting — local counting

- Extending Two-variable FO with Presburger quantifiers:
- New logics: MP (global) and L-MP (local)

Uniform characterisation of sum-GNNs

Theorem (Benedikt et al. 2024)

- The language L-MP is equivalent to sum-GNNs using eventually constant activation functions
- Allowing global aggregation in sum-GNNs, equivalence to MP.

a vertex has more red than blue neighbours

↓

L-MP expressible → sum-GNN expressible

$$\text{trRelU}\Big(\sum_{u \in N_G(v)} P_r(u) - \sum_{u \in N_G(v)} P_b(u)\Big)$$

$$\varphi(x) = (\#_y[E(x,y) \wedge L_r(y)] - \#_y[E(x,y) \wedge L_b(y)] \geq 0)$$

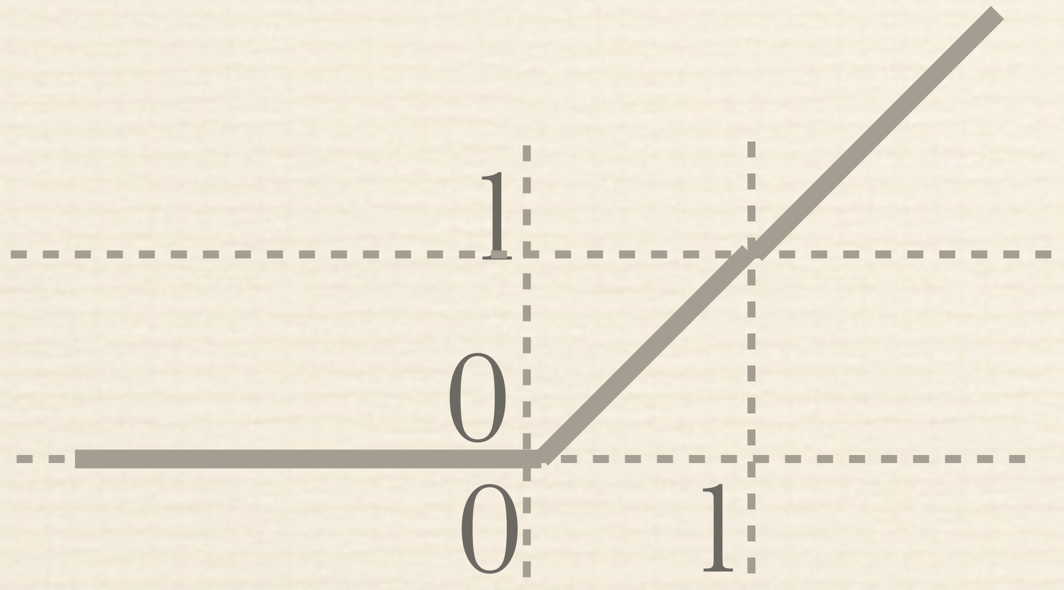
Michael Benedikt, Chia-Hsuan Lu, Boris Motik, and Tony Tan. Decidability of graph neural networks via logical characterizations (2024)

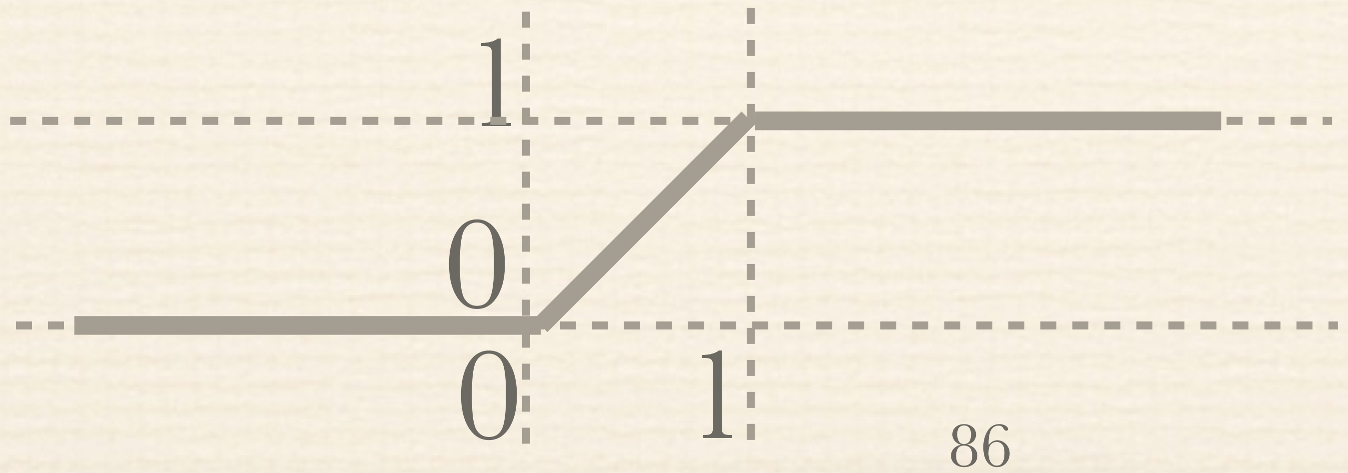
Uniform characterisation of sum-GNNs

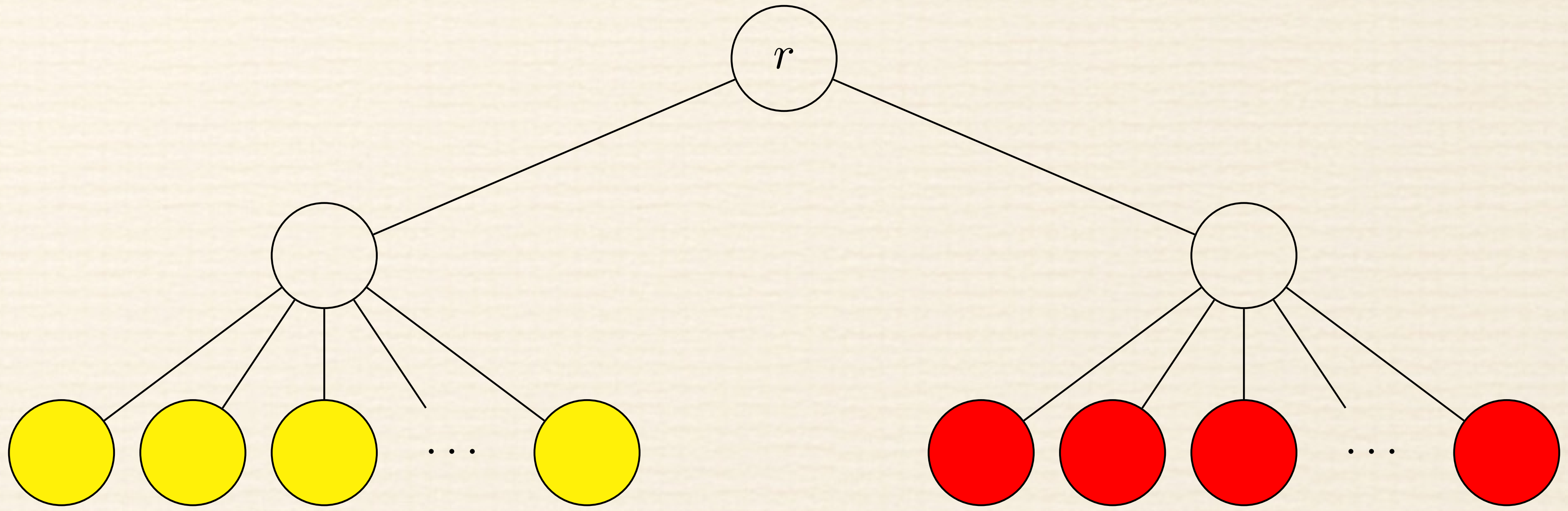
Theorem (Benedikt et al. 2024)

- The language L-MP is equivalent to sum-GNNs using eventually constant activation functions
- Allowing global aggregation in sum-GNNs, equivalence to MP.

- What about ReLU?? This activation is not eventually constant. Theorem fails.
- What about max, avg as aggregation functions?







Generalisation and expressivity

- We want the generalisation error $L_{\mathscr{D}}(\xi) - L_{\mathscr{T}}(\xi)$ to be small.

Theorem (Vapnik and Chervonenkis 1964)

- For $\delta > 0$, with probability $1 - \delta$ (in our selection of training data $\mathscr{T}$ of size m) for all $\xi \in \mathscr{H}$:

$$L_{\mathscr{D}}(\xi) - L_{\mathscr{T}}(\xi) \leq \sqrt{\frac{2d \log(\frac{em}{d})}{m}} + \sqrt{\frac{\log(\frac{1}{\delta})}{2m}}$$

- where d is the VC dimension of $\mathscr{H}$

Vladimir Vapnik and Alexey Chervonenkis: The uniform convergence of frequencies of the appearance of events to their probabilities (1964)

VC dimension

- A set of graphs $G_1, \ldots, G_d$ is shattered by an embedding class $\mathscr{H}$

  if, for any labeling $y_1, \ldots, y_d \in \{0,1\}^d$

  we can find an embedding $\xi \in \mathscr{H}$ (which may depend on the labeling)

  such that $\xi(G_1) = y_1, \ldots, \xi(G_d) = y_d$

- VC dimension= maximal number of graphs that can be shattered.

VC dimension

- VC dimension= maximal number of graphs that can be shattered.

Let us assume we consider graphs up to size n: $\mathcal{G}_n$

Theorem

The VC dimension of MPNNs on $\mathcal{G}_n$ is bounded by the number of graphs in $\mathcal{G}_n$ that can be distinguished by MPNNs.

We can also show matching lower bound.

Corollary

The VC dimension of MPNNs on all graphs is unbounded.

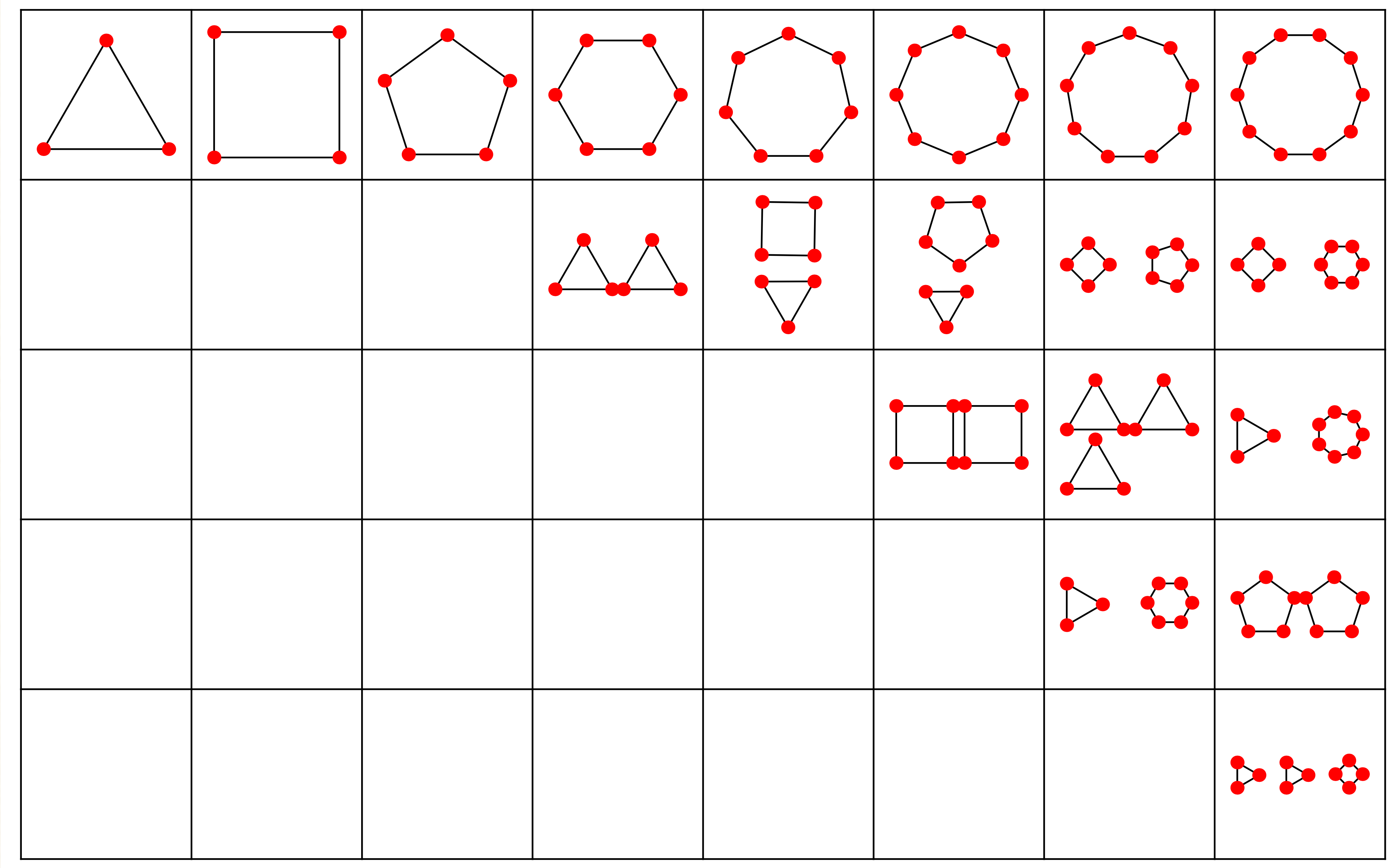
Christopher Morris, F. Geerts, J. Tonshoff, Martin Grohe: *WL meet VC (2023)*

Colour complexity

If color refinement does not need many colours for a graph: low colour complexity

We can get smaller bounds on number of distinguished graphs.

VC dimension is small → Less training data needed



on 2-regular graphs only one colour is needed.

MPNNs only distinguish based on size

Christopher Morris, F. Geerts, J. Tonshoff, Martin Grohe: *WL meet VC (2023)*

Generalisation and expressivity

Only tip of iceberg

Continuity and covering numbers

Distance measures

graphons

graph neural tangent kernels

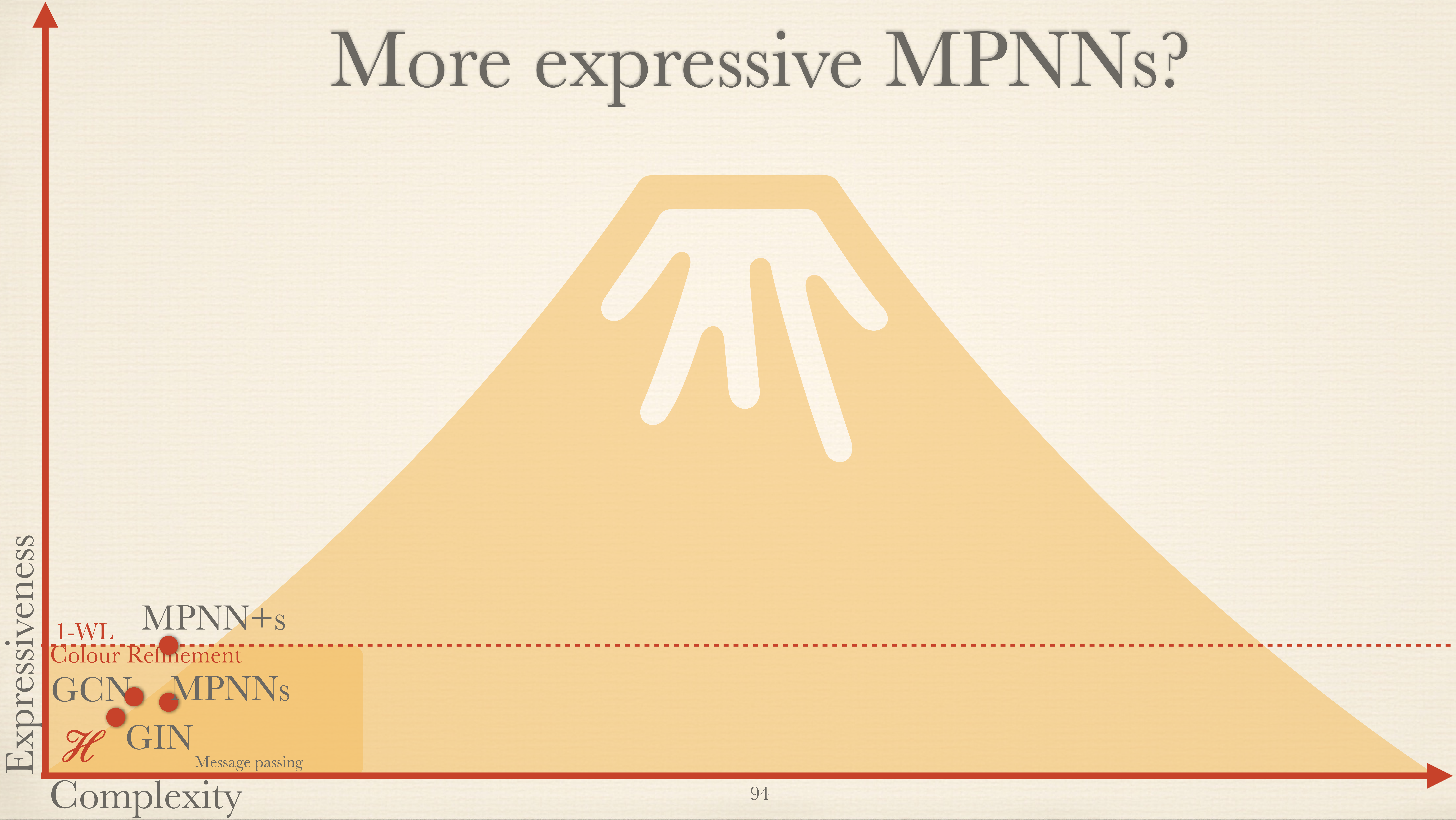
Understanding precise impact of expressiveness on generalization, not well understood yet

Questions?

More powerful methods

Boosting expressive power

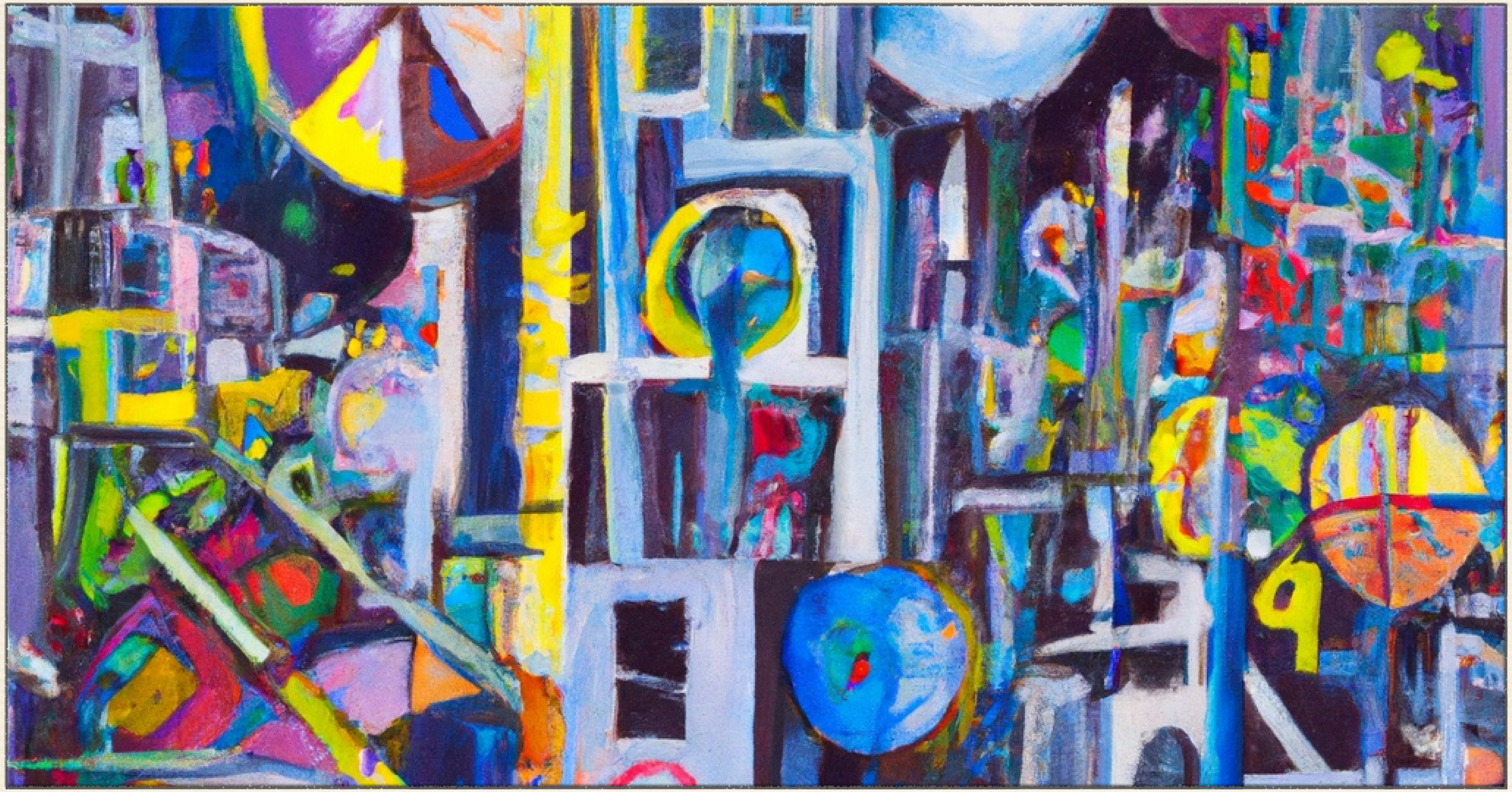
More expressive MPNNs?



How to beyond MPNNs?

Theoretical research guides architecture design!

- Feature augmentation
- subgraph GNNs.
- Higher-order MPNNs

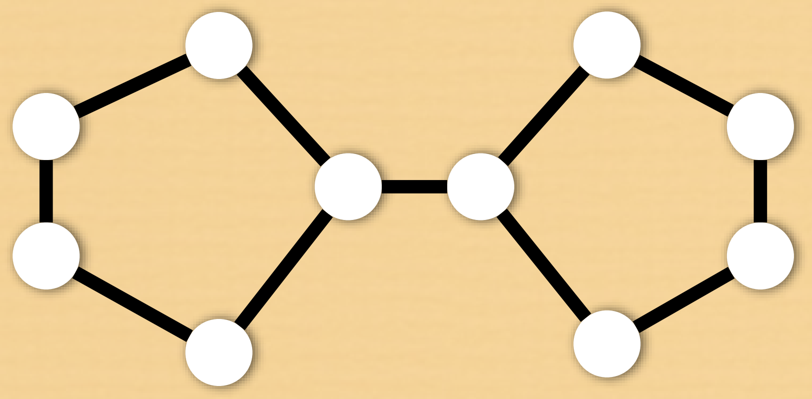


Feature Augmentation

Boost the expressive power by adding information

Feature engineering

- Deep learning and MPNNs have replaced "old school" feature engineering approach.



→

- Number of edges
- Number of cycles of length 5
- Centrality measures

→ $\mathbb{R}^d$ → SVM

- MPNNs were supposed to learn such features automatically …

Idea #1: Adding expressive features

Recall:

Theorem

$\mathsf{hom}(T, G) = \mathsf{hom}(T, H)$ for all *trees* T if and only if no MPNN can distinguish G from H.

- What if we add subgraph information before doing message-passing?

More than trees

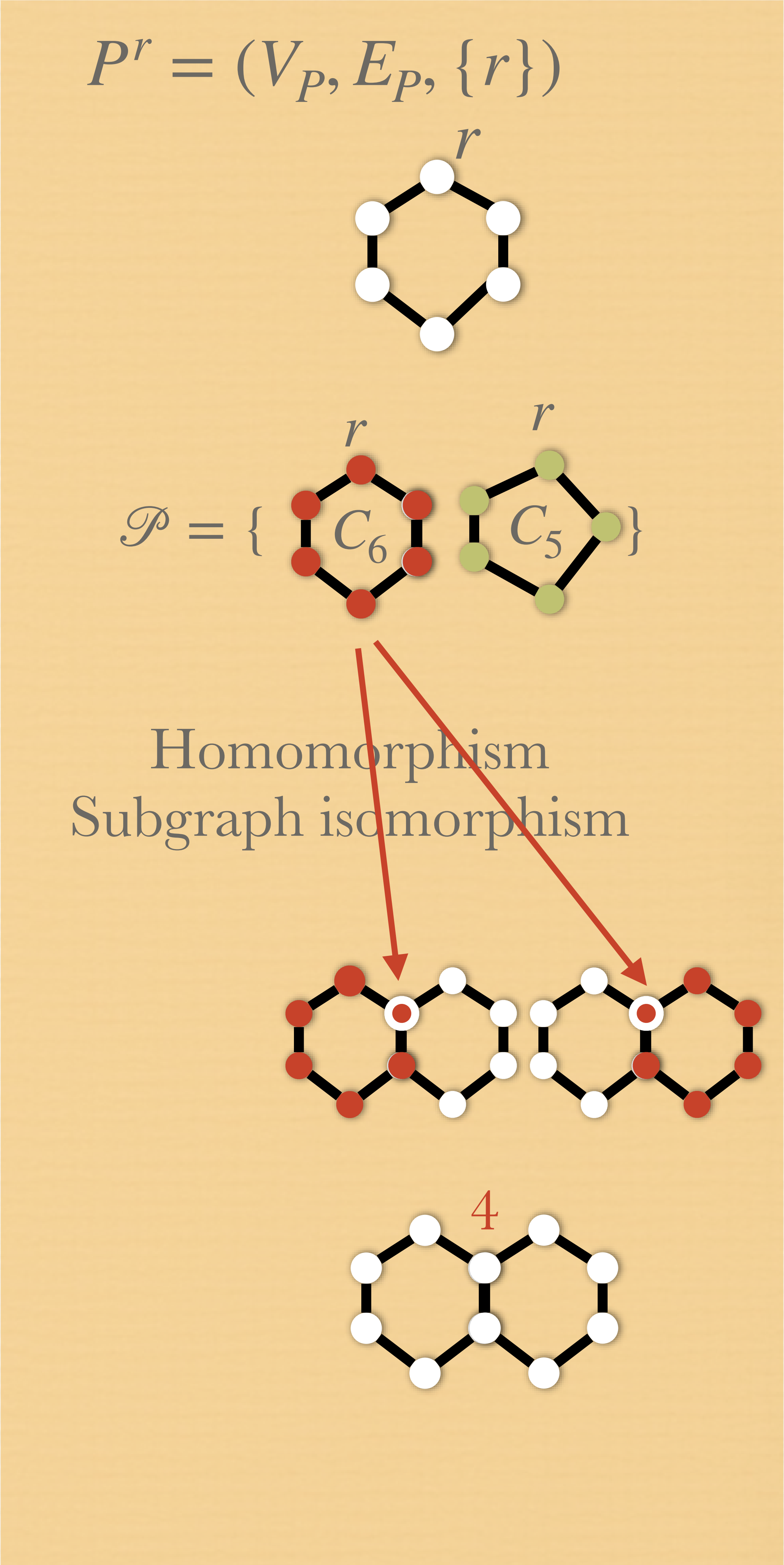
Structural encodings

1. Choose collection of rooted graph patterns/motifs

$$\mathscr{P} := \{P_1^r, \ldots, P_\ell^r\}$$

2. Choose how to match subgraphs in $\mathscr{P}$ with data graph G

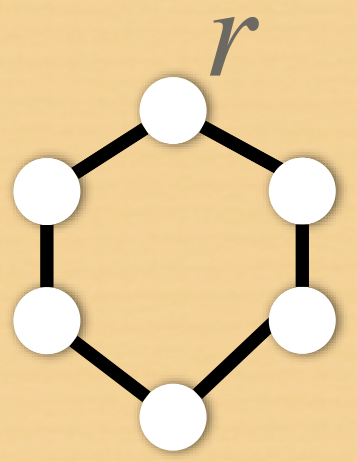
3. Add count of matches to vertices as extended features.



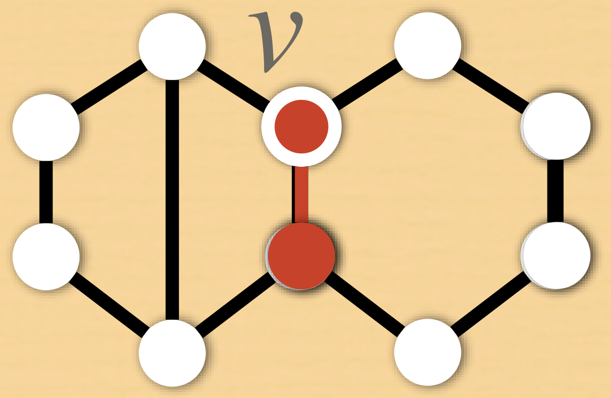
Matches

- Homomorphism: edge preserving
- Subgraph isomorphism: bijection , edge preserving
- Induced subgraph isomorphism: bijection, edge preserving (both ways)

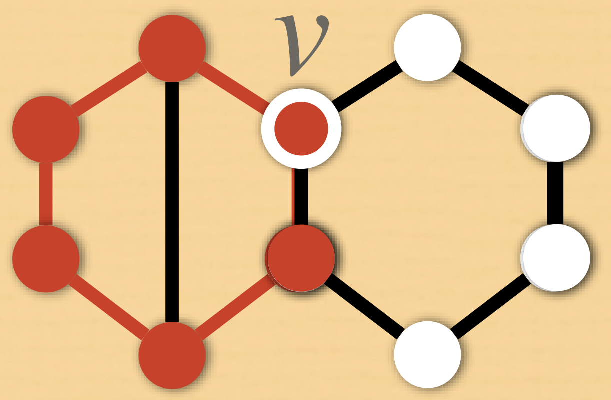
$P^r = (V_P, E_P, \{r\})$



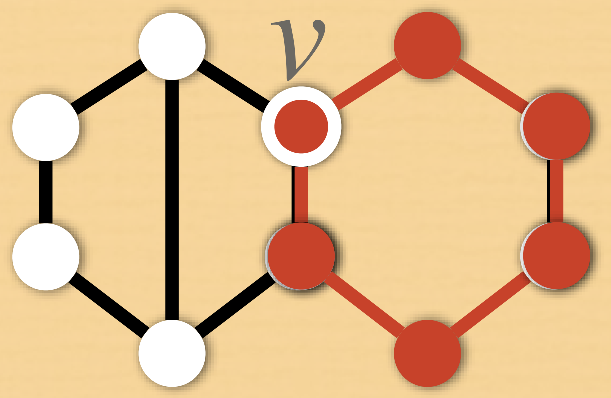
$\pi : V_P \to V_S \subseteq V_G$ containing v



hom(P^r, G^v)



subiso(P^r, G^v)



indsubiso(P^r, G^v)

Counts

$\mathscr{P}$-MPNNs

- Add structural encoding as vertex features and run MPNN

$$\mathscr{P} := \{P_1^r, \ldots, P_\ell^r\}$$

$\mathscr{P}$-MPNNs

$$\xi^{(0)}(G, v) := \text{Hot-one encoding of label of vertex } v + \mathsf{hom}(P_1^r, G^v), \ldots, \mathsf{hom}(P_\ell^r, G^v)$$

$$\xi^{(t)}(G, v) := \mathsf{Upd}^{(t)}\Big(\xi^{(t-1)}(G, v), \mathsf{Agg}^{(t)}\big(\{\{\xi^{(t-1)}(G, v), \xi^{(t)}(G, u), \mathsf{hom}(P_1^r, G^u), \ldots, \mathsf{hom}(P_\ell^r, G^u) \mid u \in N_G(v)\}\}\big)\Big)$$

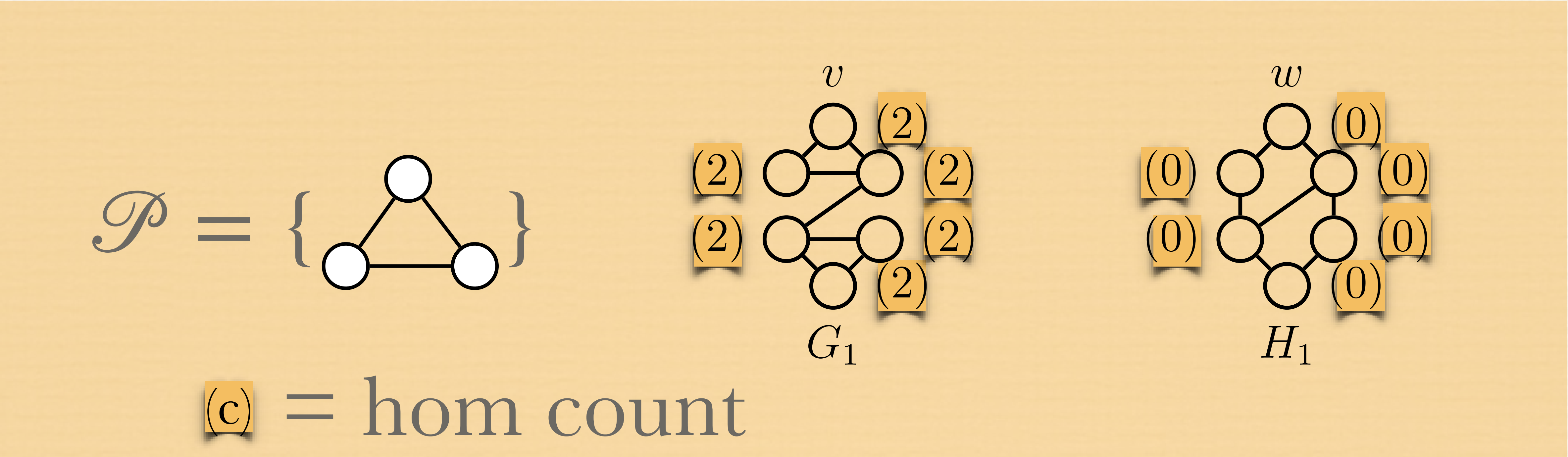
$$\rho(G) := \mathsf{Readout}\Big(\{\{\xi^{(L)}(G, v) \mid v \in V_G\}\}\Big)$$

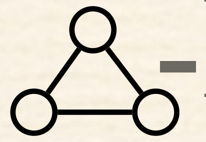
hom counts of patterns

- Did we increase expressive power?

Barceló et al.: *Graph neural networks with local graph parameters.* (2021)

$\mathscr{P}$-MPNNs

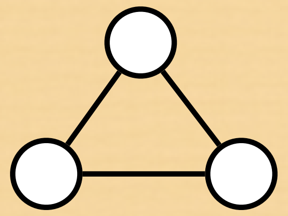


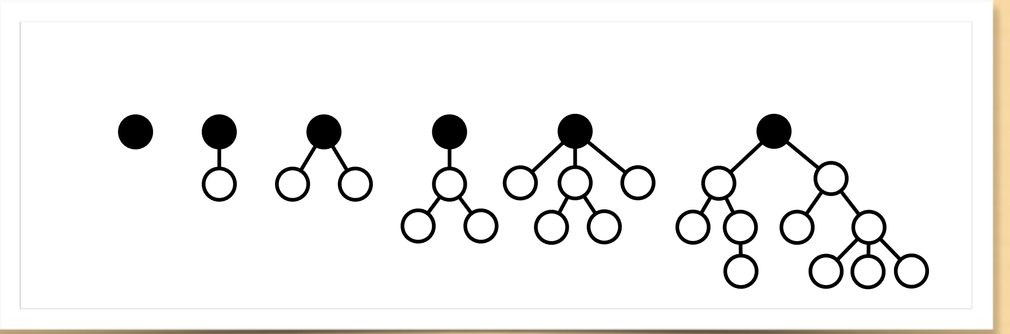
- We have seen that these graphs equivalent for colour refinement but clearly not for -MPNNs.
- So, increase in power!
- What is their precise expressive power?

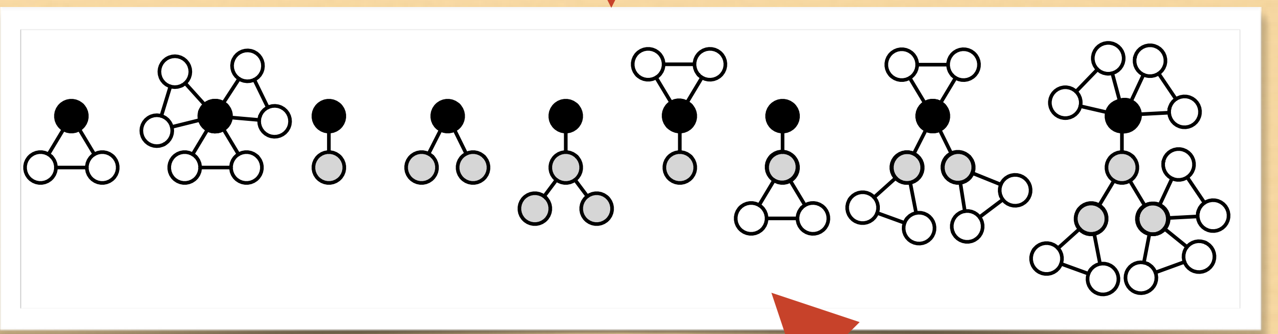
$\mathscr{P}$-MPNNs: Expressive power

Theorem

$\mathsf{hom}(T, G) = \mathsf{hom}(T, H)$ for all $\mathscr{P}$–*pattern trees* T *if and only if* no P-MPNN can distinguish G from H.

$\mathscr{P} = \{$  $\}$





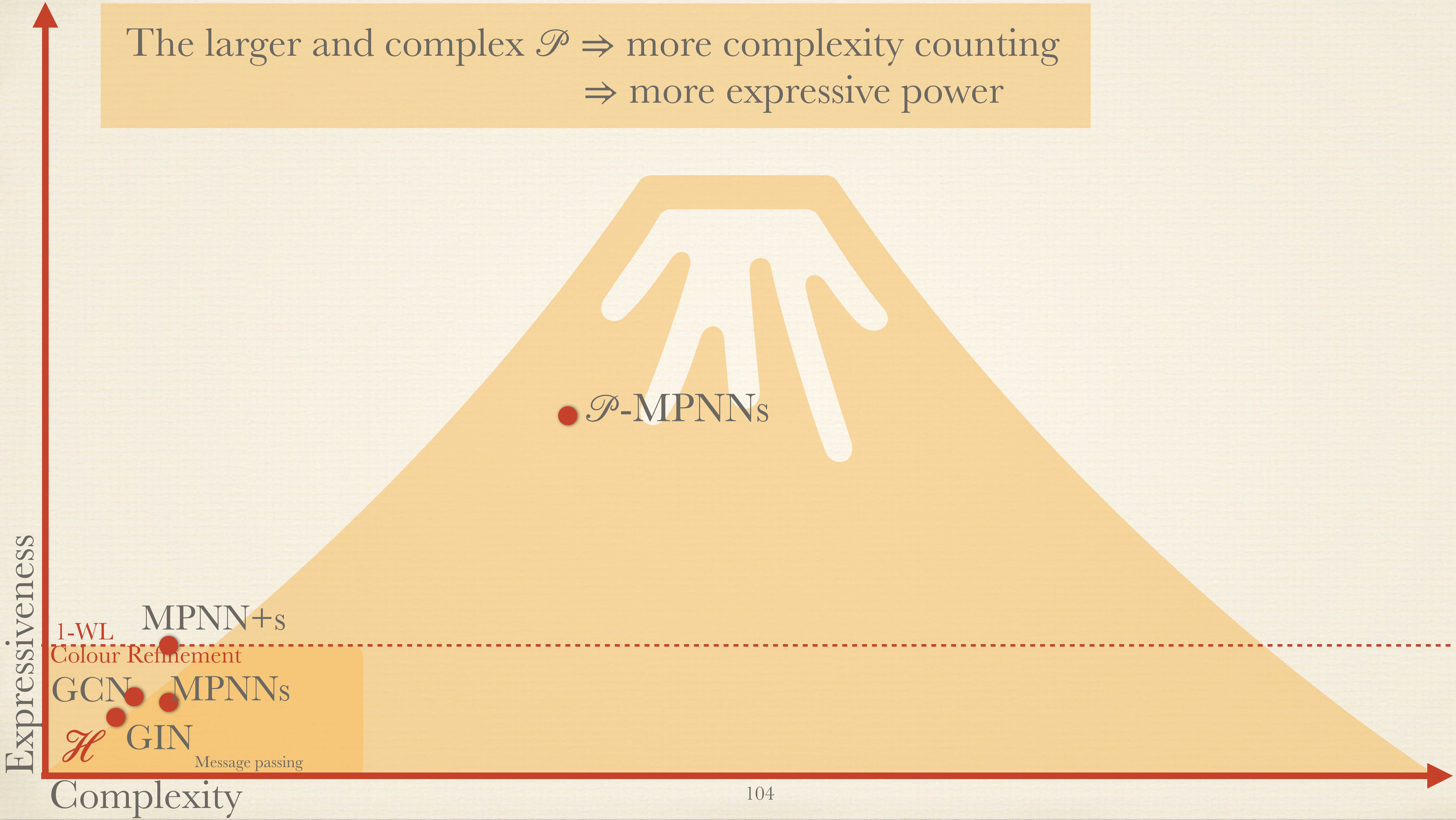
Take tree: add in each tree vertex copies of rooted patterns

Zinc dataset

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**

Barceló et al.: *Graph neural networks with local graph parameters.* (2021)

The larger and complex $\mathscr{P} \Rightarrow$ more complexity counting
$\Rightarrow$ more expressive power



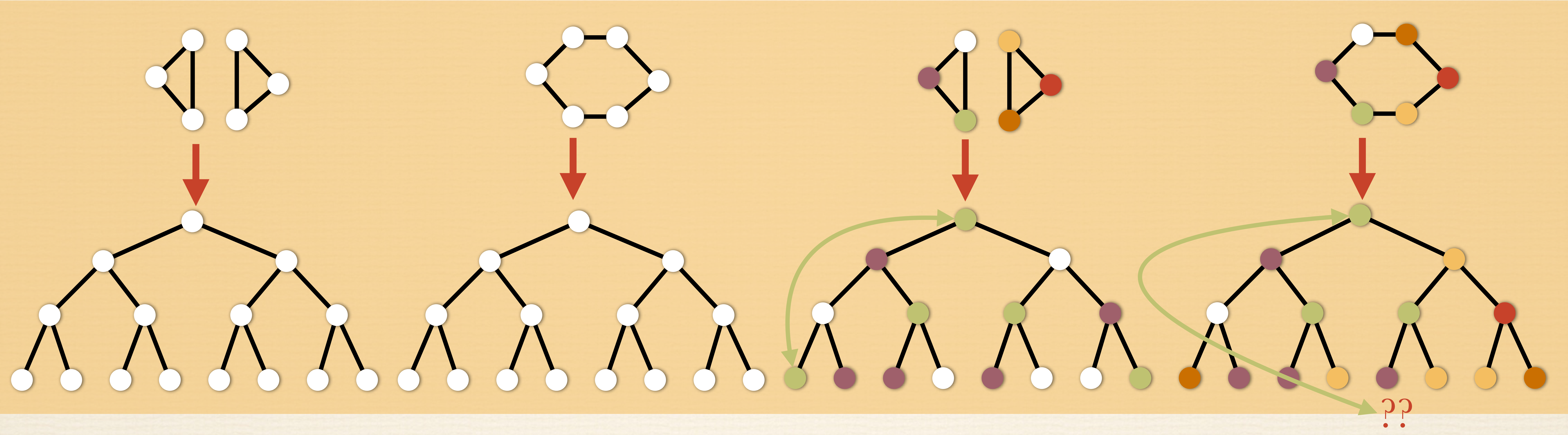
Idea #2: (Random) Vertex identifiers

- Message-Passing is only based on vertex features and adjacency information.
- Two different vertices with the same vertex features will be treated the same (if they have the same colour in colour refinement).

What if we add vertex identifiers?

Vertex identifiers

Self identification: useful for cycle detection



In terms of colour refinement: every vertex has a unique colour

rMPNNs

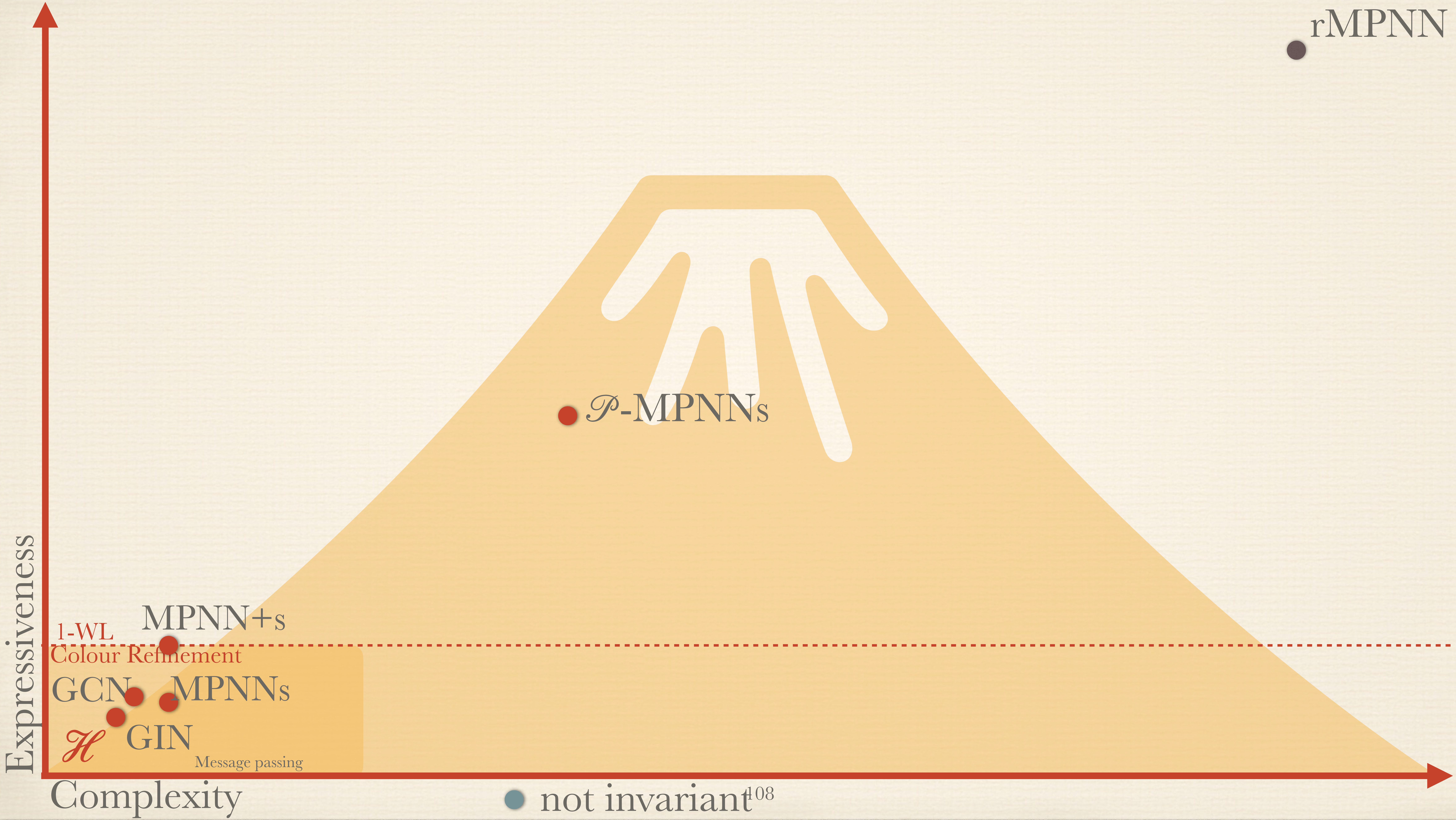
- How to choose identifiers? Common choice is at random!
- With high probability random features are vertex identifiers

Theorem

rMPNNs approximate any invariant graph/vertex embedding with high probability

- Invariance of computed embedding only in expectation!

Dasoulas et al.: *Coloring graph neural networks for node disambiguation* (2020)
Abboud et al. :*The surprising power of graph neural networks with random node initialization.* (2021)
Sato et al.: *Random features strengthen graph neural networks* (2021).


rMPNN
𝒫-MPNNs
MPNN+s
1-WL
Colour Refinement
GCN
MPNNs
ℋ
GIN
Message passing
Expressiveness
Complexity
not invariant

Idea #3: Use global information

- Extract global graph information and use it as positional encodings of vertices
  - Spectral information
  - Shortest paths (distance information)
  - Biconnectivity (connectivity information)

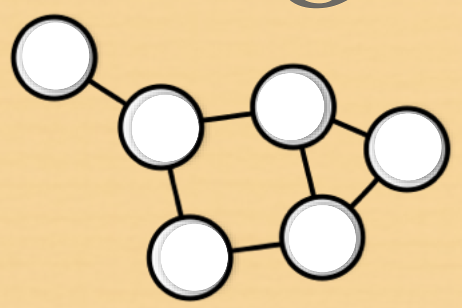
Kreuzer et al.: *Rethinking graph transformers by spectral attention* (2021)
Ying et al.: *Do transformers really perform bad for graph representation* (2021)
Lim et al.: *Sign and Basis Invariant Networks for Spectral Graph Representation Learning* (2022)
Zhang et al.: *Rethinking the expressive power of gnns via graph biconnectivity* (2023)] [2]

Spectral graph theory

- Eigenvalues/vector: $\mathbf{M} \cdot \mathbf{v} = \lambda \mathbf{v}$

- For adjacency matrices: Eigenvalues and eigenvectors of Laplacian $\mathbf{L}_G = \mathbf{D}_G - \mathbf{A}_G$

G



$$\mathbf{L}_G = \mathbf{D}_G - \mathbf{A}_G$$

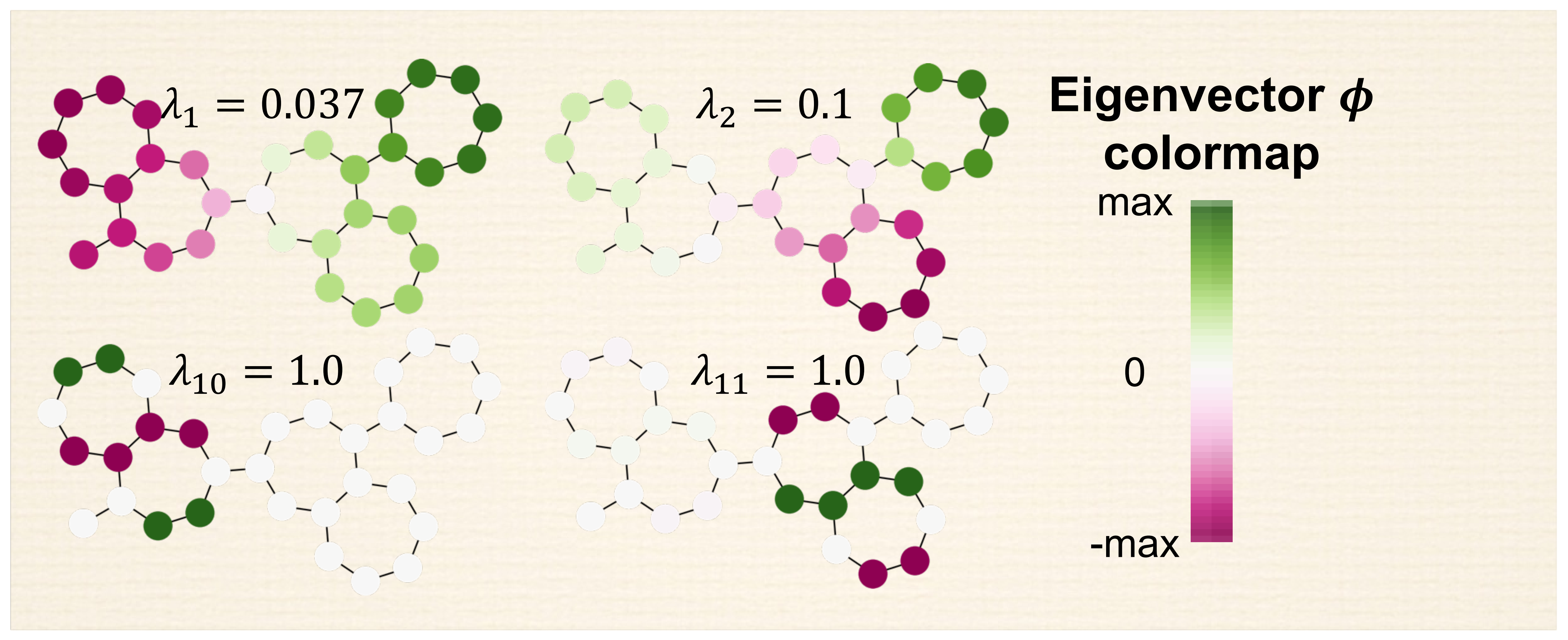
$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

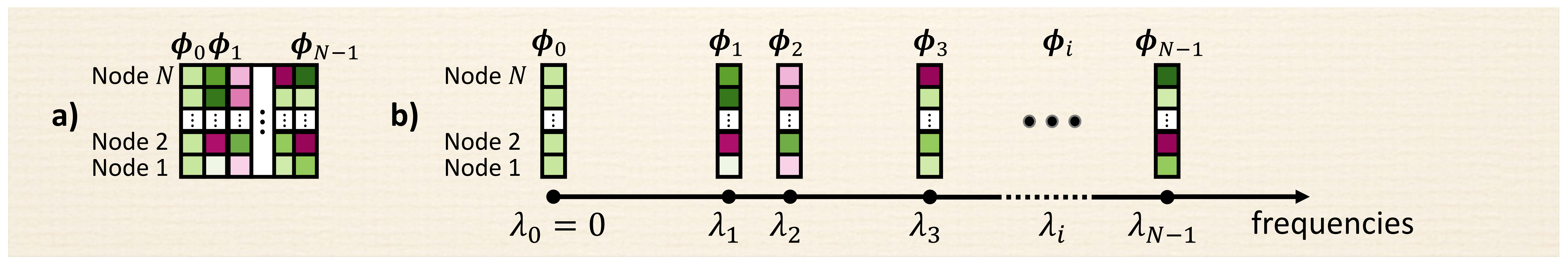
- Laplacian eigenvalues and vectors contain connectivity information
    - multiplicity 1st eigenvalue $\sim$ connected components.

Images: Wikipedia

Spectral MPNNs

Add eigenvectors as vertex features





Kreuzer et al.: *Rethinking graph transformers by spectral attention* (2021)

Spectral invariant

$$\mathbf{A} = \sum_{\lambda} \lambda \mathbf{P}_{\lambda} \qquad \mathbf{P}_{\lambda} = \begin{pmatrix} p^{\lambda}_{11} & p^{\lambda}_{12} & \cdots & p^{\lambda}_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ p^{\lambda}_{n1} & p^{\lambda}_{n2} & \cdots & p^{\lambda}_{nn} \end{pmatrix}$$

Multiset

Spectral invariant

$$v \mapsto \mathsf{specinv}(v) := (\lambda, p^{\lambda}_{vv}, \{\{p^{\lambda}_{vu} \mid u \in V_G\}\})_{\lambda \in \Lambda}$$

Graph properties

Number of length 3, 4, or 5 cycles, whether a graph is connected and the number of length k closed walks from any vertex to itself

Beyond 1-WL/Colour Refinement

Cvetković et al.: *Eigenspaces of graphs* (1997)
M. Fürer: *On the power of combinatorial and spectral invariants* (2010)

SpecMPNN

Spectral invariant

$$v \mapsto \mathsf{specinv}(v) := (\lambda, p^{\lambda}_{vv}, \{\{p^{\lambda}_{vu} \mid u \in V_G\}\})_{\lambda \in \Lambda}$$

Variation used in Signet and BasisNet → 2-WL bound

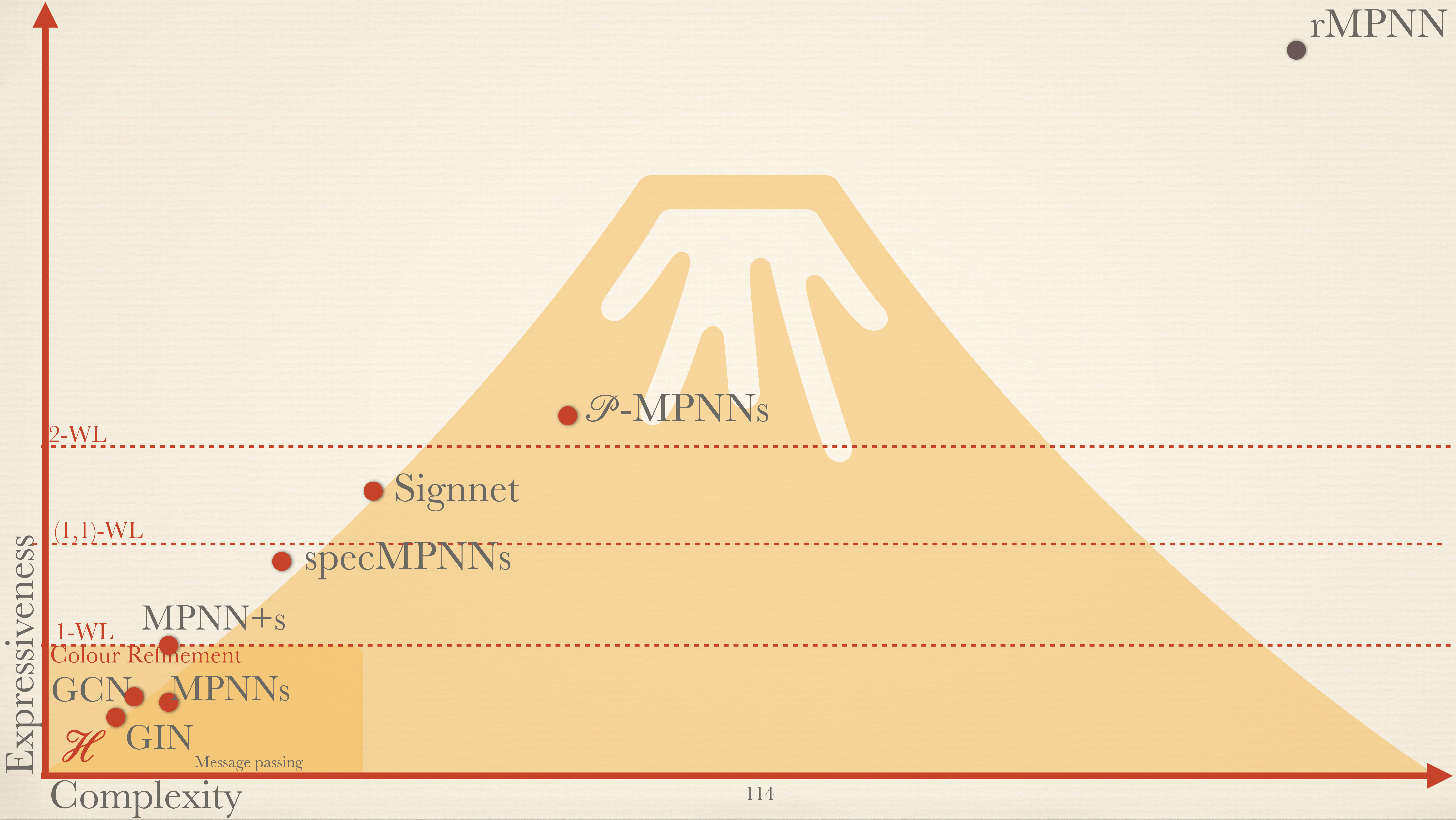
Can be using combination with any MPNN

Theorem (Seppelt and Rattan (2023)

specMPNN bounded in power by (1,1)-WL and strictly lower than 2-WL

We discuss these WL's later

On the Expressive Power of Spectral Invariant Graph Neural Networks" by Bohang Zhang, Lingxiao Zhao, Haggai Maron (2024)
G. Rattan and T. Seppelt: *Weisfeiler-Leman and Graph Spectra* (2023)



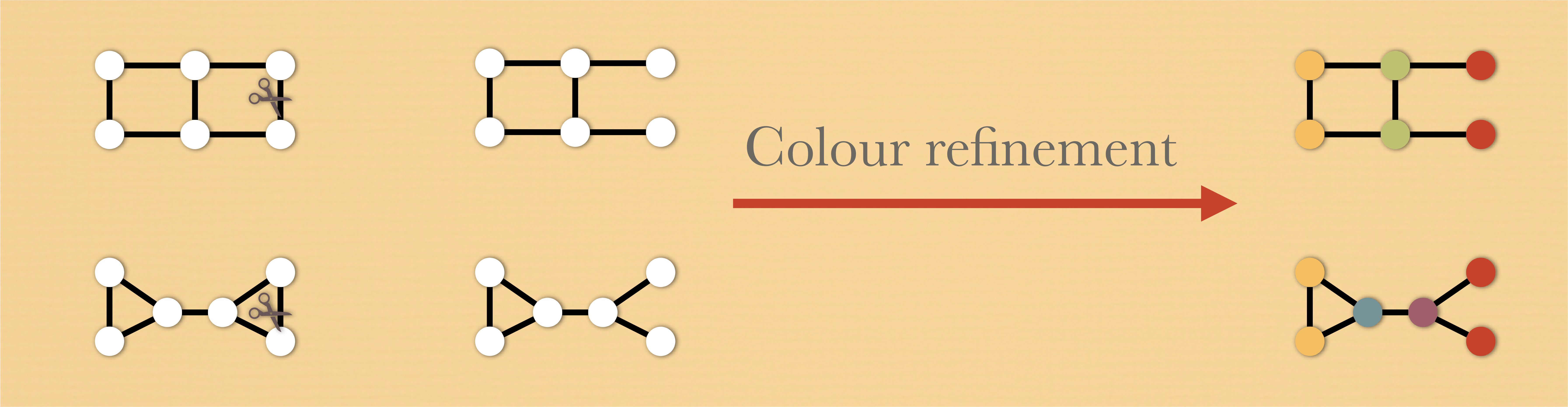
rMPNN
𝒫-MPNNs
2-WL
Signnet
(1,1)-WL
specMPNNs
MPNN+s
1-WL
Colour Refinement
GCN
MPNNs
ℋ
GIN
Message passing
Expressiveness
Complexity

Subgraph GNNs

Turning one graph into many

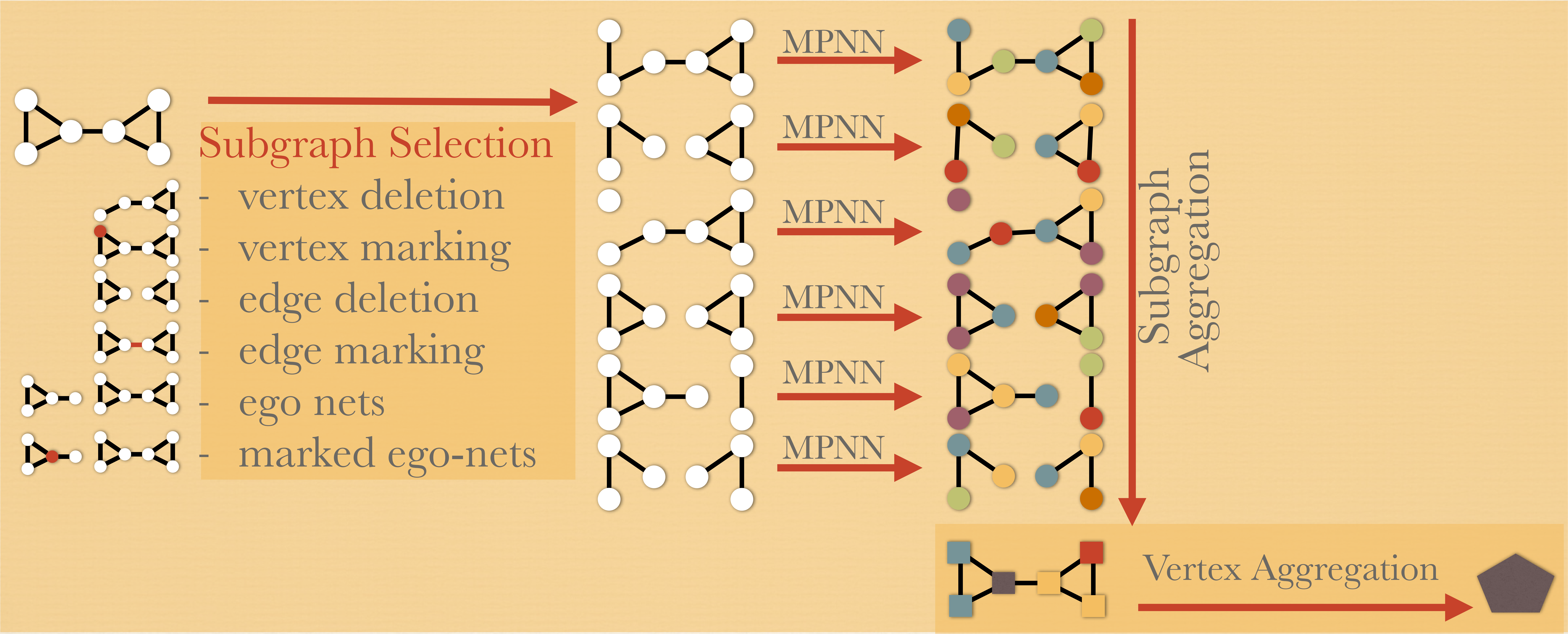
General idea

- Colour refinement equivalent graphs may contain colour refinement inequivalent subgraphs.

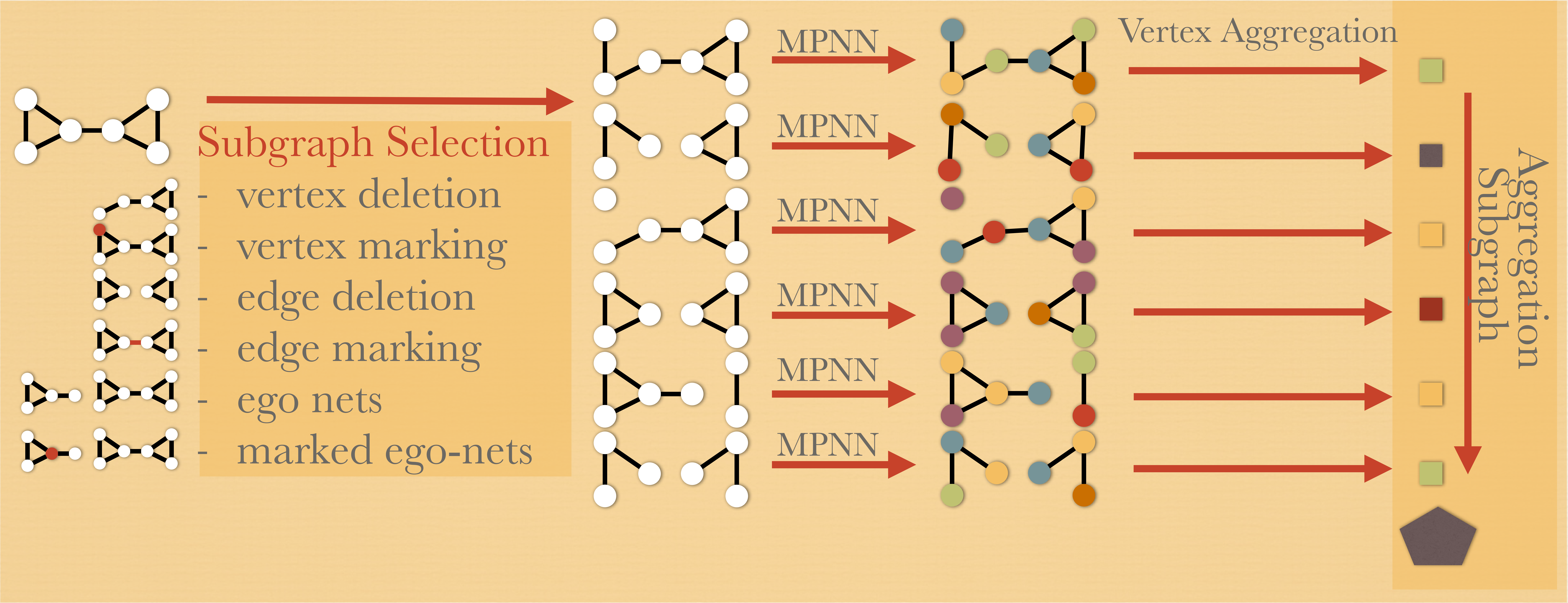


- View graphs as a collection of subgraphs then run MPNN

Subgraph↦Vertex Aggregation

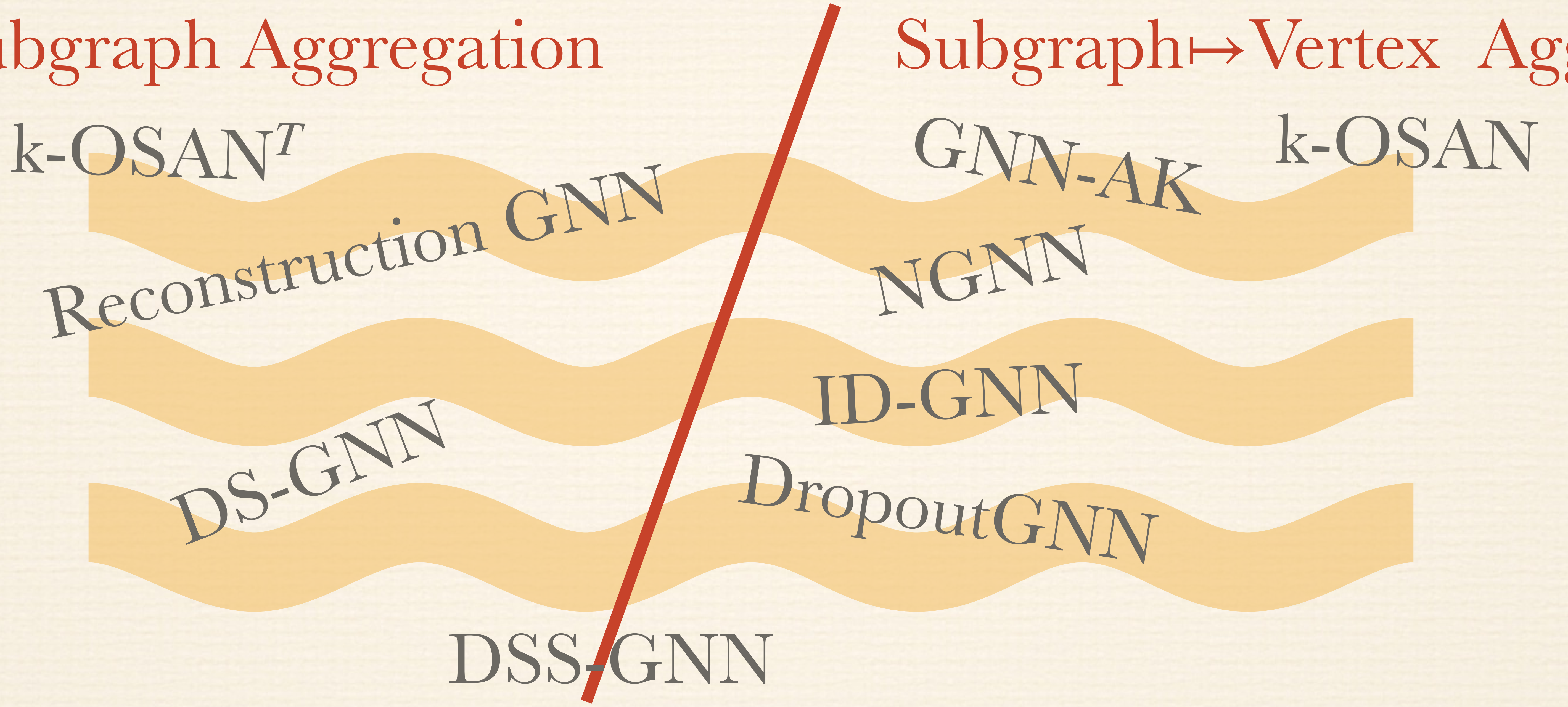


Vertex↦Subgraph Aggregation



The subgraph GNN "wave"

Vertex↦Subgraph Aggregation | Subgraph↦Vertex Aggregation



All provably more expressive than MPNNs*

Bevilacqua et al: *Equivariant subgraph aggregation network* (2022)
Cotta et al.: *Reconstruction for powerful graph representations* (2021)
Bevilacqua et al.: *Understanding and extending subgraph GNNs by rethinking their symmetries* (2022)
Huang et al.: *Boosting the cycle counting power of graph neural networks with I2-GNNs* (2022)
Papp et al.: *DropGNN: Random dropouts increase the expressiveness of graph neural networks*. (2021)
Qian et al.: *Ordered subgraph aggregation networks.* (2022)
You et al.: *Identity-aware graph neural networks*. (2021)
Zhang and P. Li. *Nested graph neural networks* (2021)
Zhao et al.: *From stars to subgraphs: Uplifting any GNN with local structure awareness* (2022)

Selection policies

DS-GNN
- vertex deletion
- edge deletion
- ego nets
- marked ego-nets

ID-GNNs
- marked ego-nets

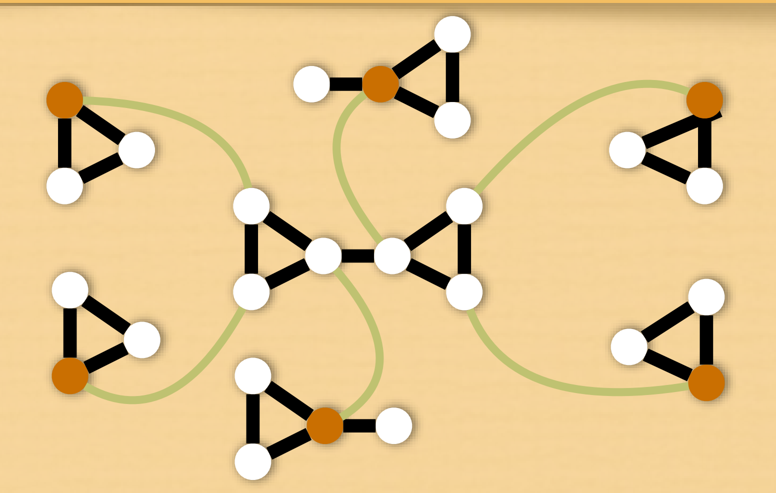
GNNs-AK
- ego-nets

k-OSAN
- size k subgraph marking

Rec-GNN
- k-vertex deletion

NGNN
- ego-nets

Popular/effective: ego-nets



k-OSAN

Theorem (Qian et al. 2022)

- k-OSANs and k-OSANs^t encompass *almost all* subgraph methods with selection policy involving k vertices.
- *Strictly bounded* in expressive power by *(k+1)-WL*
- *Incomparable* to k-WL.

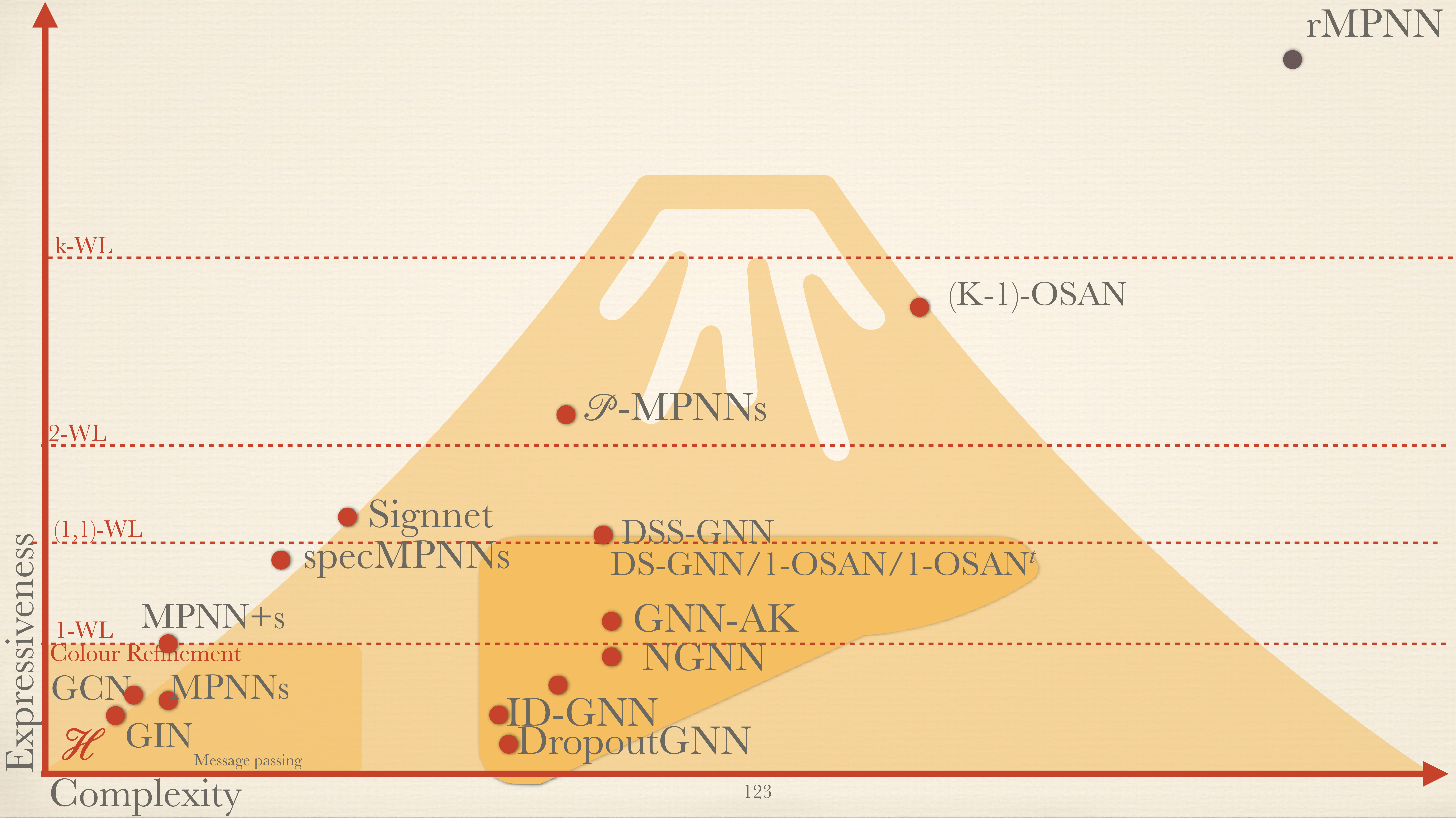
k=2

- if 2-WL cannot distinguish graphs, then neither can 1-OSANs
- 2-WL can distinguish more graphs than 1-OSANs
- There exists graphs than can be distinguished by 1-OSANs but not by MPNNs, and vice versa, there exists graphs that can be distinguished by MPNNs but not by 1-OSANs

Qian et al.: *Ordered subgraph aggregation networks.* (2022)

Subgraph GNNs

- Can always ensure to be strictly more expressive than MPNNs by including original graph in batch.
- Tractability only when easy subgraph policies are used, i.e., leading to a small number (linear) of subgraphs.
- Seems a good balance between complexity and expressiveness


rMPNN
k-WL
(K-1)-OSAN
𝒫-MPNNs
2-WL
Signnet
(1,1)-WL
DSS-GNN
specMPNNs
DS-GNN/1-OSAN/1-OSAN'
MPNN+s
GNN-AK
1-WL
Colour Refinement
NGNN
GCN
MPNNs
ID-GNN
ℋ
GIN
Message passing
DropoutGNN
Expressiveness
Complexity

K-dimensional Weisfeiler-Leman

Boosting expressive power by higher-order message-passing

More powerful heuristic

Apply heuristic on G and H: If Heuristic say "no" then $G \ncong H$, otherwise we do not know.

$G \cong H$?

Colour refinement → No → $G \ncong H$

↓

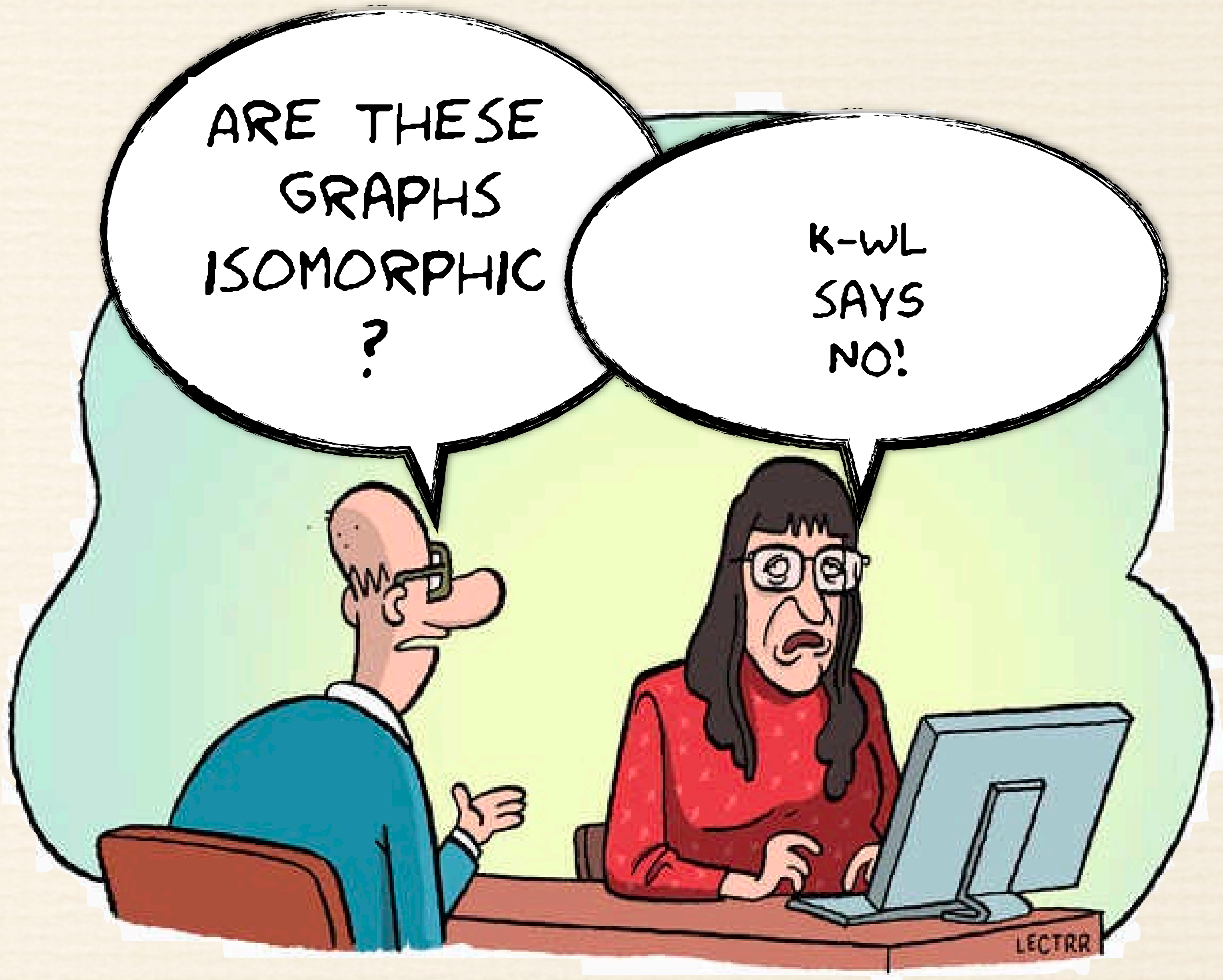
1-WL → No → $G \ncong H$

↓

2-WL → No → $G \ncong H$

↓

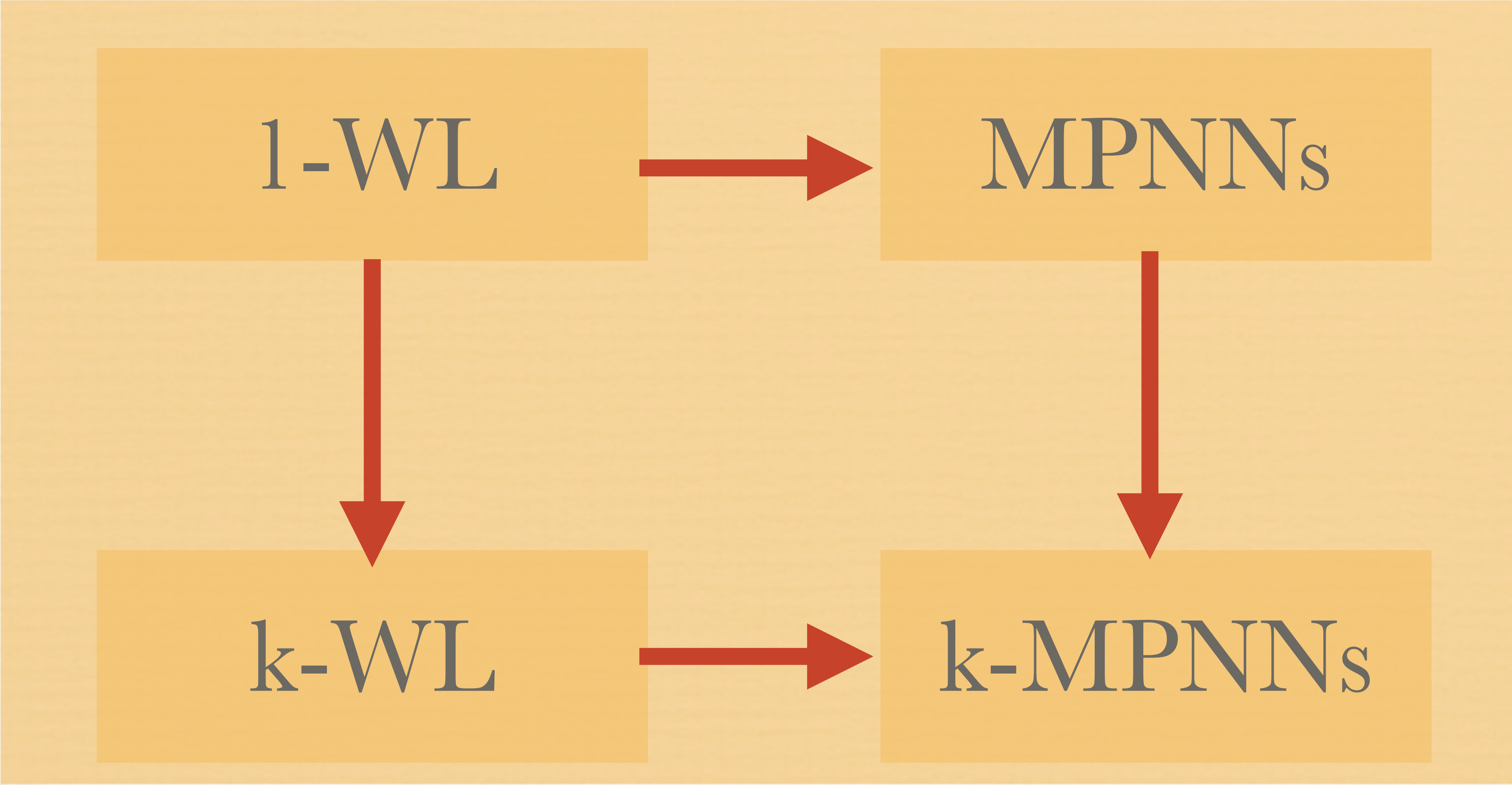
⋰⋱



Idea: higher-order GNNs

Theorem (Dell et al. 2018, …)

$\mathsf{hom}(T, G) = \mathsf{hom}(T, H)$ for all graphs T *of tree width* k
if and only if
k-WL cannot tell apart G from H



k-MPNNs will detect more graph information than MPNNs

Z. Dvorák: *On recognizing graphs by numbers of homomorphisms* (2010)
Dell et al. *Lovász meets Weisfeiler and Leman* (2018)

k-Folklore GNNs (k-FGNs)

$$\xi^{(t)}(G, v_1, \ldots, v_k) := \mathsf{MLP}_1^{(t)}\Big(\sum_{u \in V_G} \prod_{i=1}^{k} \mathsf{MLP}_2^{(t)}(\xi^{(t-1)}(G, v_1, \ldots, v_{i-1}, u, v_{i+1}, \ldots, v_k))\Big)$$

k-vertex embedding

Global aggregation

Uses multiplication

Expressive power?

Theorem (Maron et al. 2019), Azizian and Lelarge 2021)

$$\rho(k-\text{FGNN}) = \rho(k\text{-WL})$$

Maron et al.: *Provably powerful graph networks* (2019)
W. Azizian and M. Lelarge. *Characterizing the expressive power of invariant and equivariant graph neural networks* (2021)

k-GNNs

A simpler architecture:

$$\xi^{(t)}(G, v_1, \ldots, v_k) := \sigma\Big(\xi^{(t-1)}(G, v_1, \ldots, v_k)\mathbf{W}_1^{(t)} + \big(\sum_{i=1}^{k} \sum_{u \in V_G} \xi^{(t)}(G, v_1, \ldots, v_{i-1}, u, v_{i+1}, \ldots, v_k)\big)\mathbf{W}_2^{(t)} \Big)$$

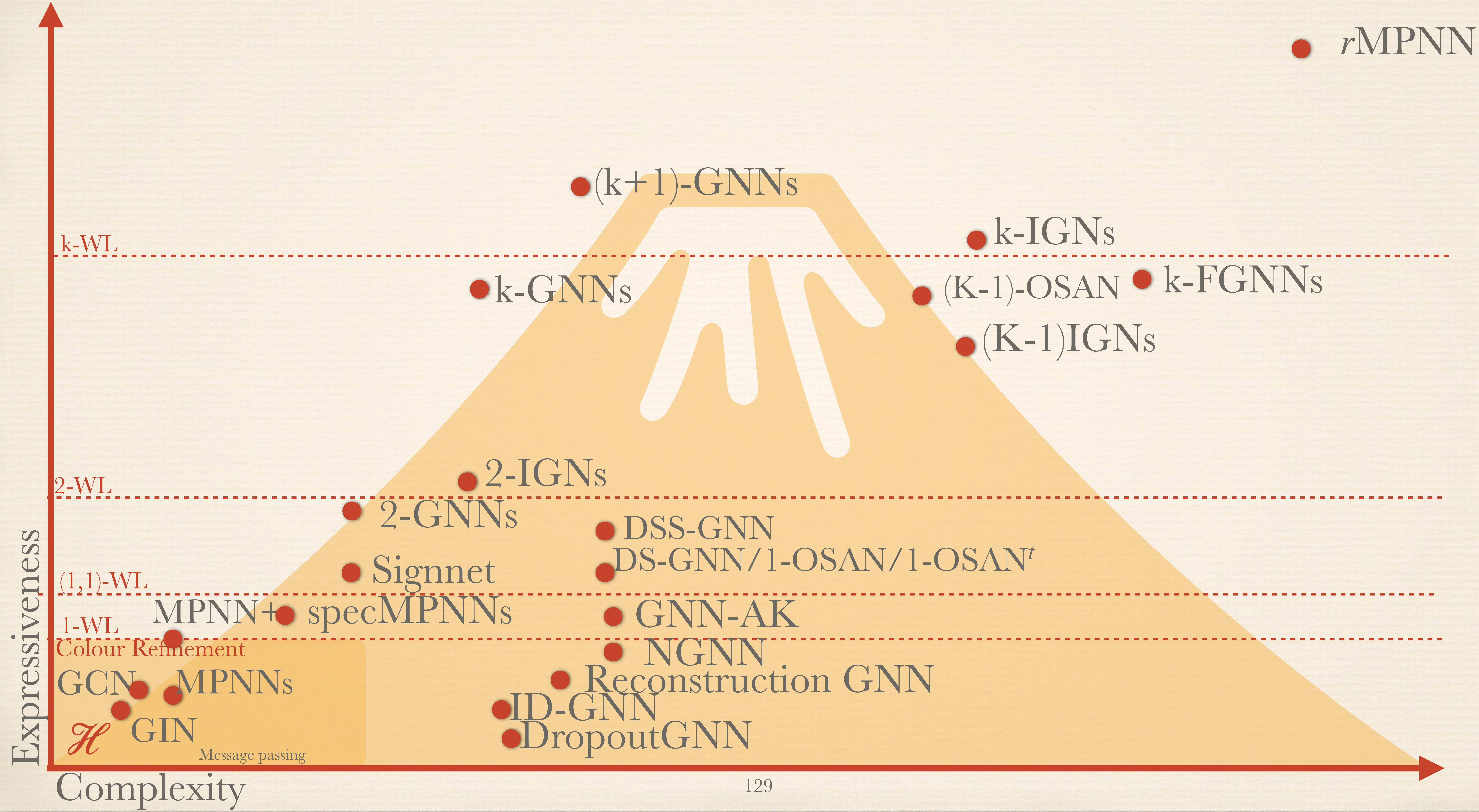
Global aggregation

Expressive power?

Theorem (Morris et al. 2019)

$$\rho(k - \text{GNN}) = \rho(k\text{-WL})$$

Morris et al.: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019)


rMPNN
(k+1)-GNNs
k-IGNs
k-WL
k-GNNs
(K-1)-OSAN
k-FGNNs
(K-1)IGNs
2-IGNs
2-WL
2-GNNs
DSS-GNN
DS-GNN/1-OSAN/1-OSANt
Signnet
(1,1)-WL
MPNN+
specMPNNs
GNN-AK
1-WL
Colour Refinement
NGNN
Reconstruction GNN
GCN
MPNNs
ID-GNN
GIN
DropoutGNN
Message passing
Expressiveness
Complexity

Questions?

Conclusions

And look ahead

What to use?

Subgraph	Feature Augmentation	Higher-order
❖ Small graphs ❖ Good compromise in general	❖ Large training datasets ❖ Invariance not important ❖ Preprocessing ok	❖ Graphs are small ❖ Efficiency not essential ❖ Expressivity guarantee needed

Road ahead

Expressiveness

- A lot of recent progress
- WL hierarchy needs better reconciliation with practice
- Hom count characterisations
- Relational

Connection with Learning??

- Optimisation and training unexplored

- Generalisation properties
- Sample efficiency?

Gary et al. :*Generalization and Representational Limits of Graph Neural Networks* (2020)
Morris et al: *WL meet VC* (2023)

Conclusion

- Study of expressive is beautiful area of research for ML researchers
- Combines theory and practice in an elegant way
- Many unresolved questions …