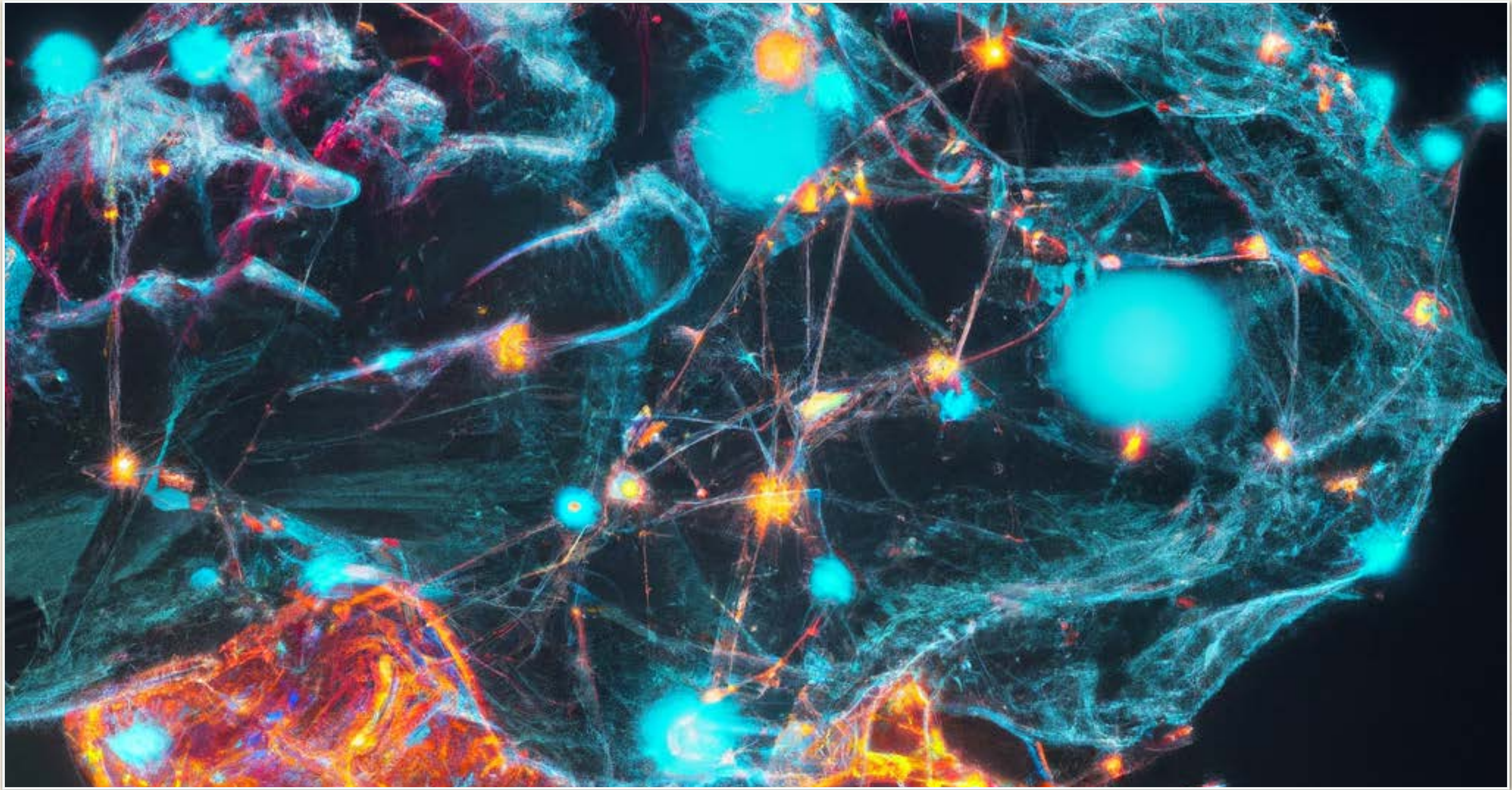
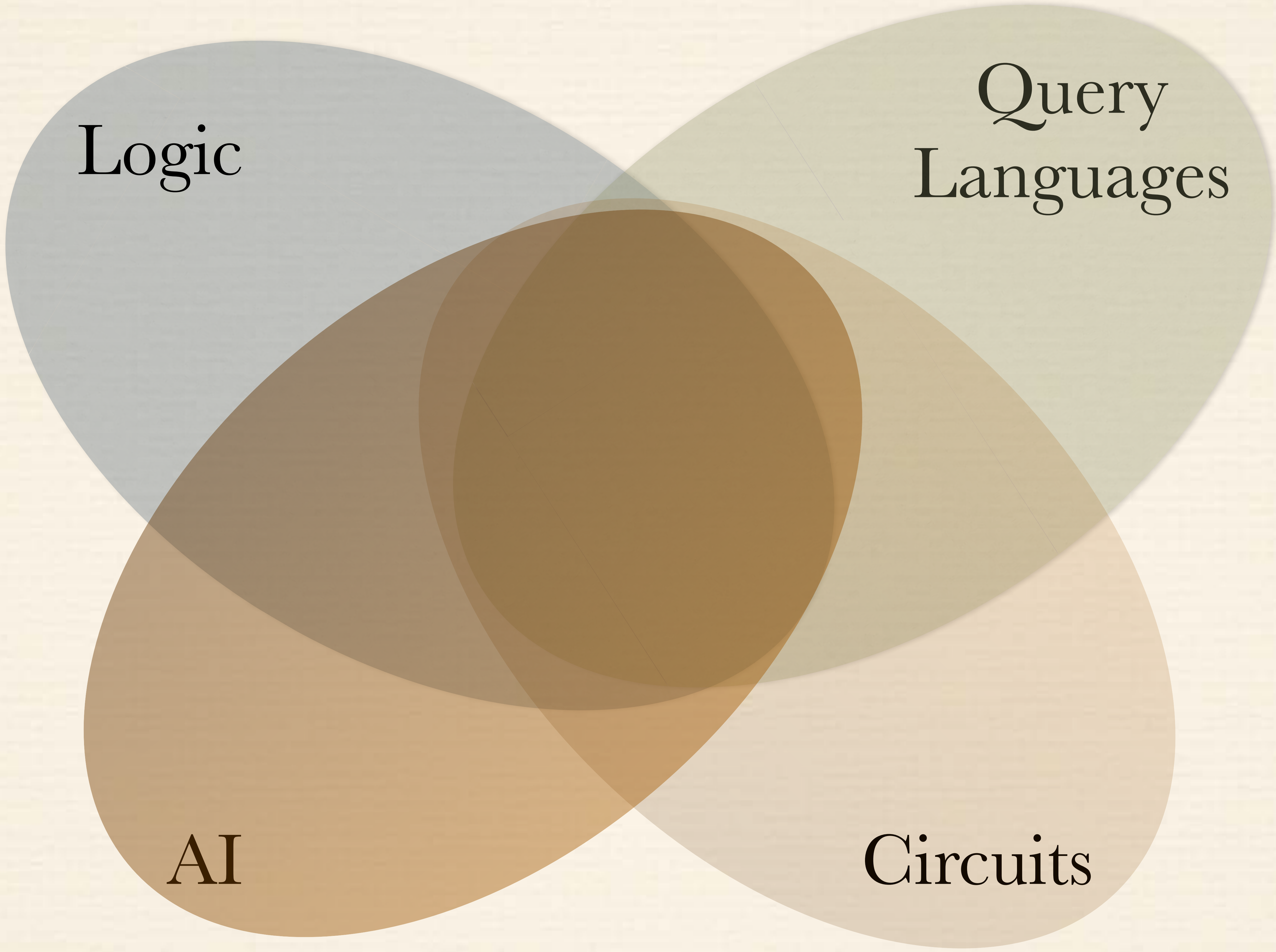


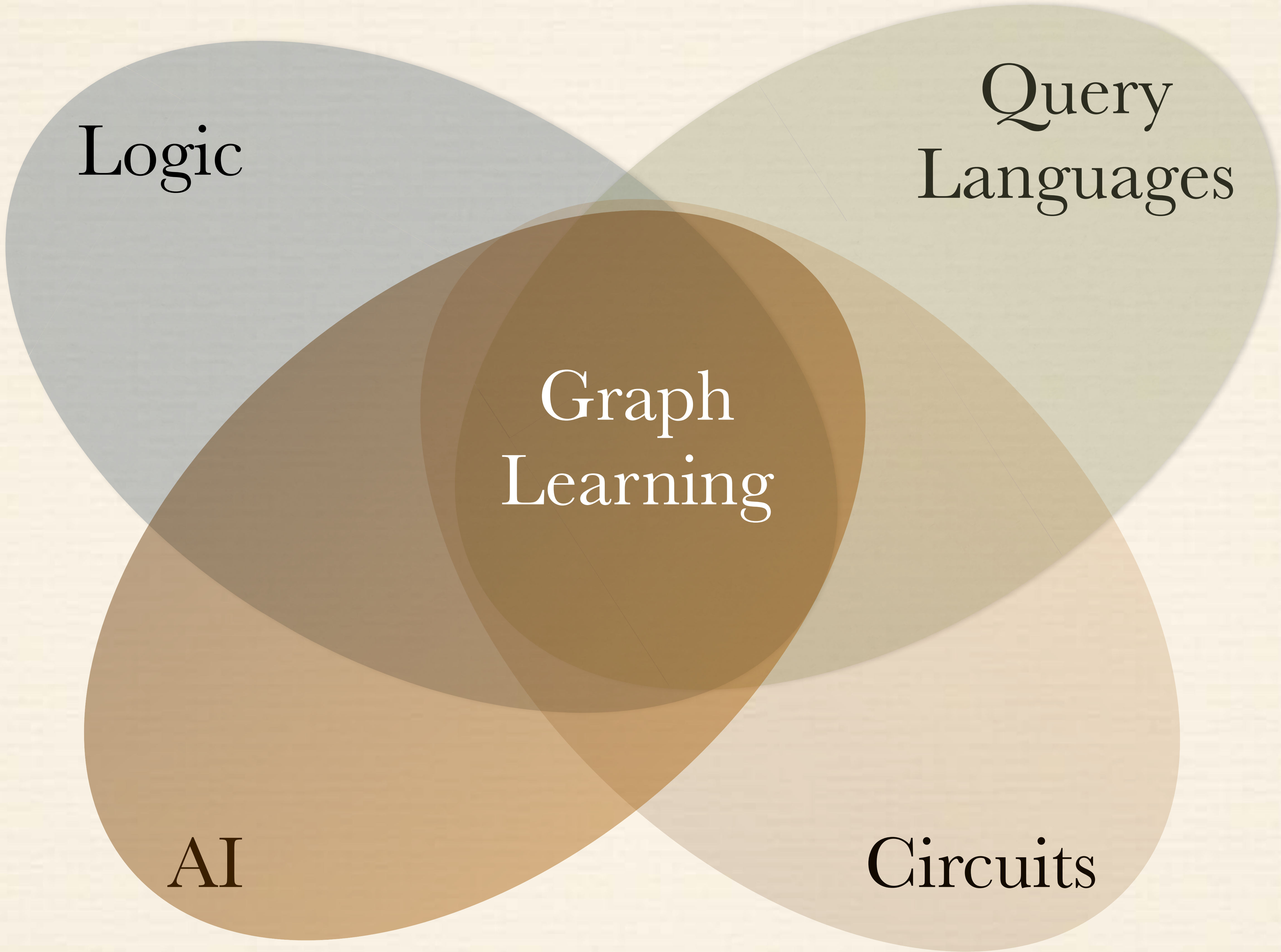
The power of graph learning

Floris Geerts (University of Antwerp, Belgium)

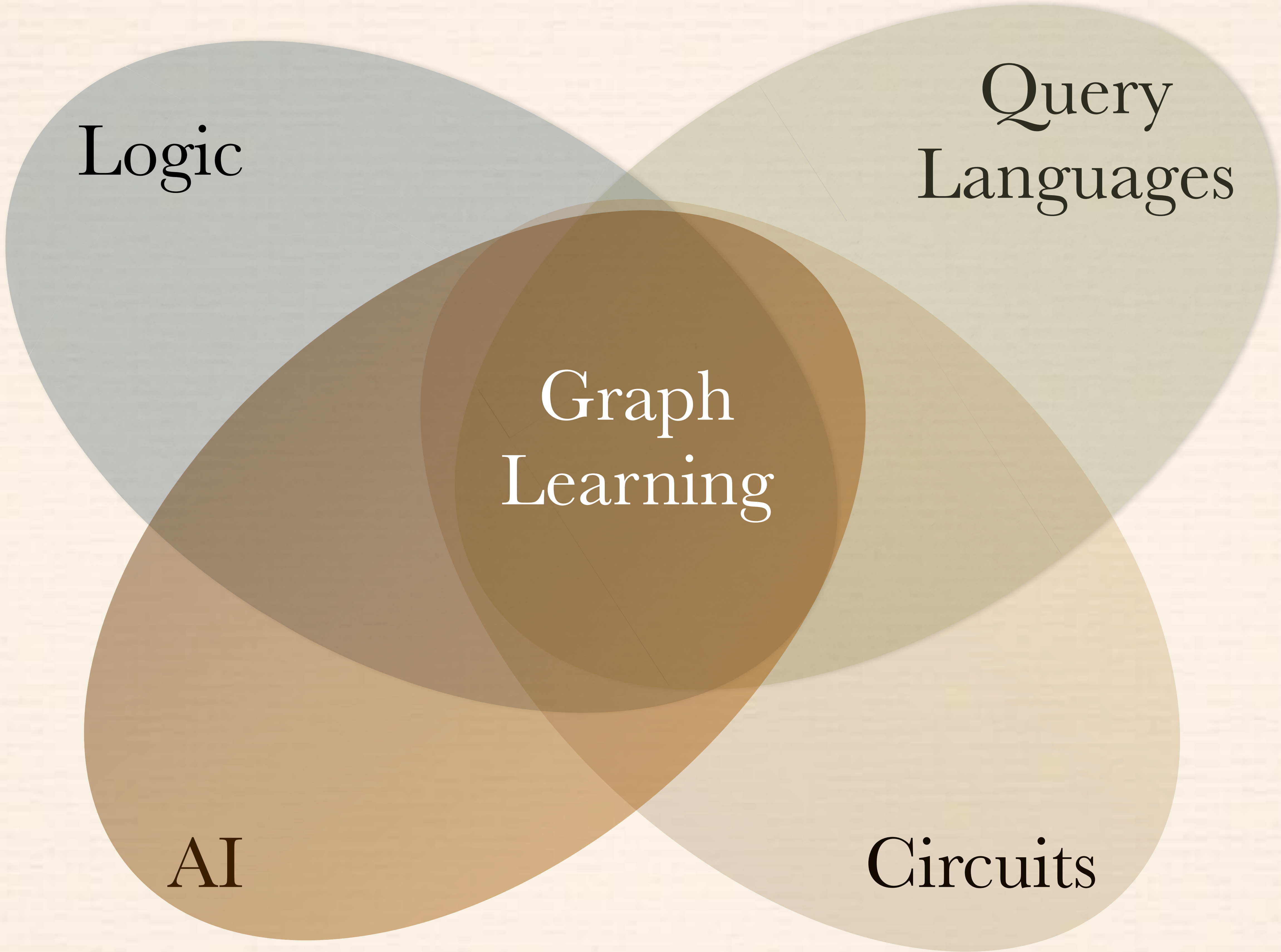
Logic and Algorithms in Database Theory and AI



Logic and Algorithms in Database Theory and AI



Logic and Algorithms in Database Theory and AI



So what is graph learning?

Graph learning

Prediction and classification problems on graphs

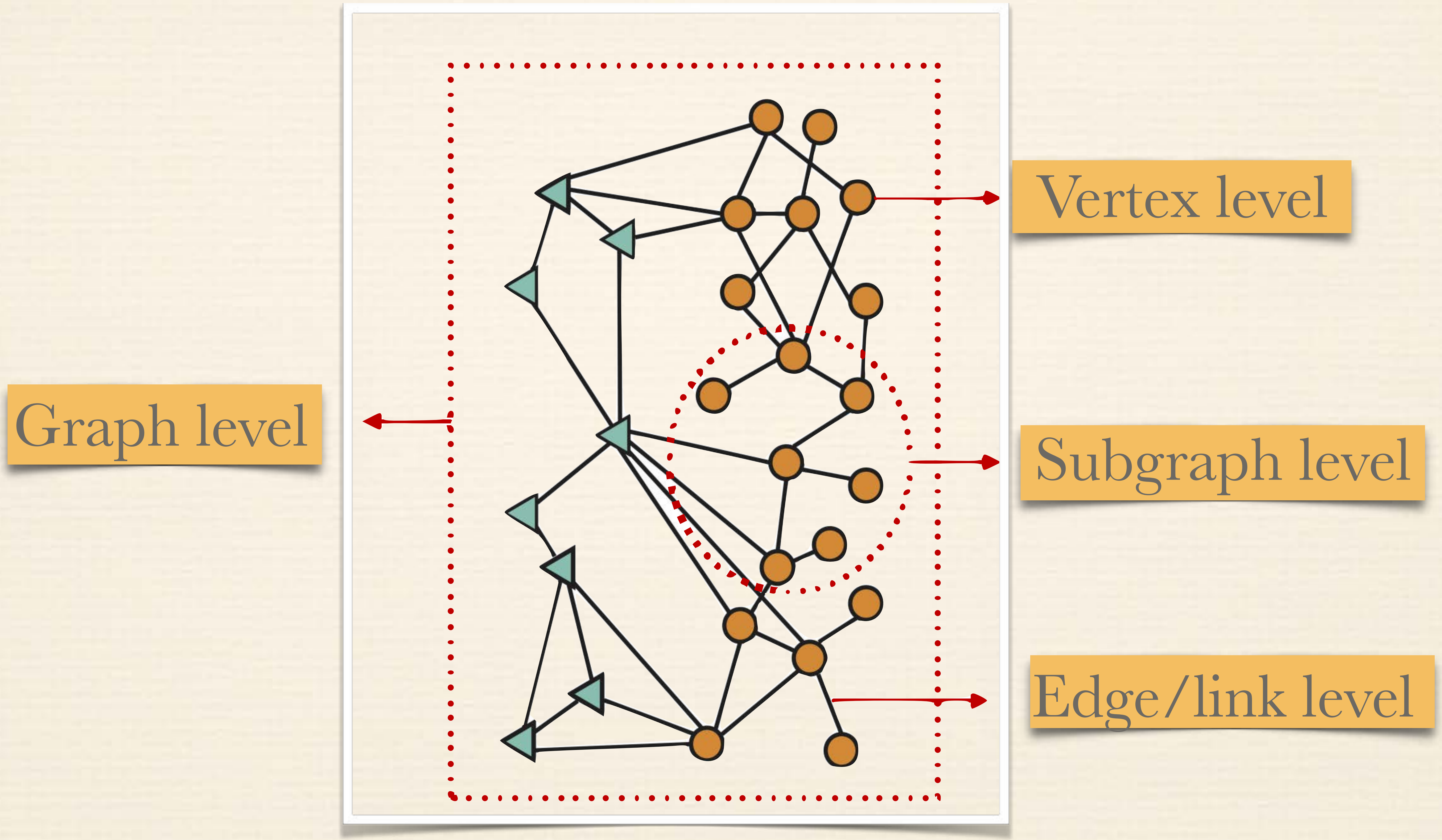
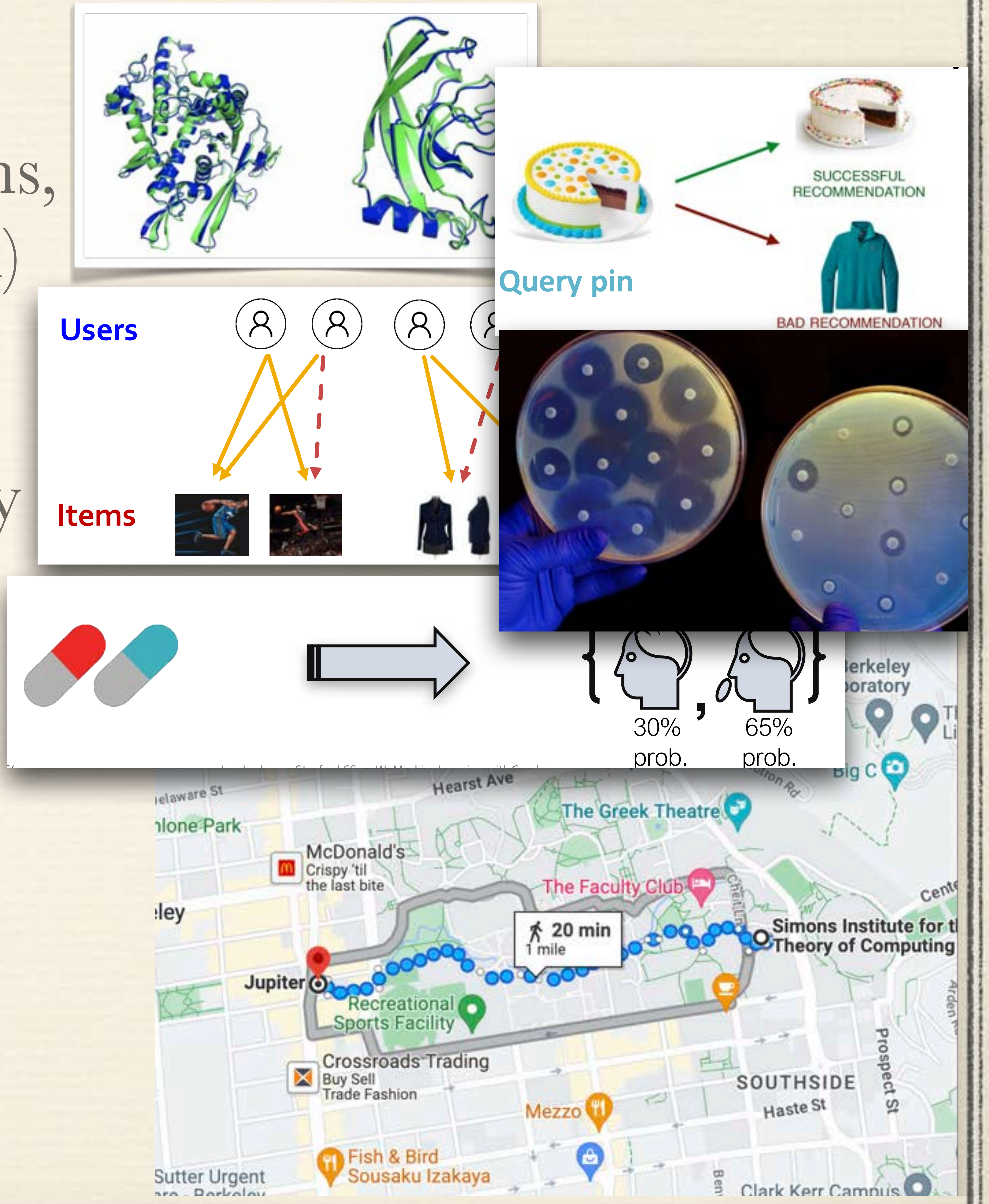


Image: Machine Learning on Graphs, Stanford course Jure Leskovec

Examples

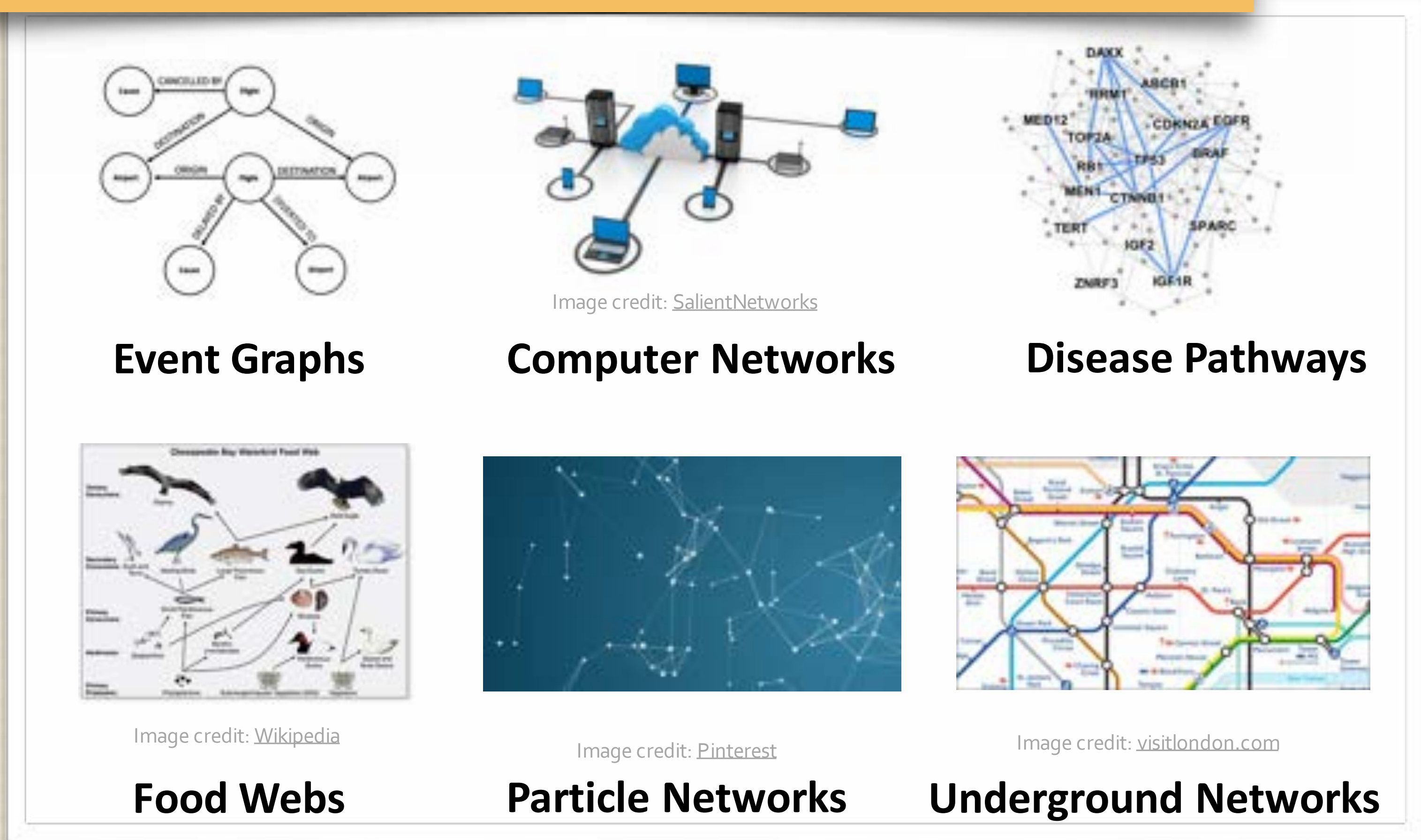
- Vertex classification: categorise online user/items, location amino acids (protein folding, alpha fold)
- Link prediction: knowledge graph completion, recommender systems, drug side effect discovery
- Graph classification: molecule property, drug discovery
- Subgraph tasks: traffic prediction

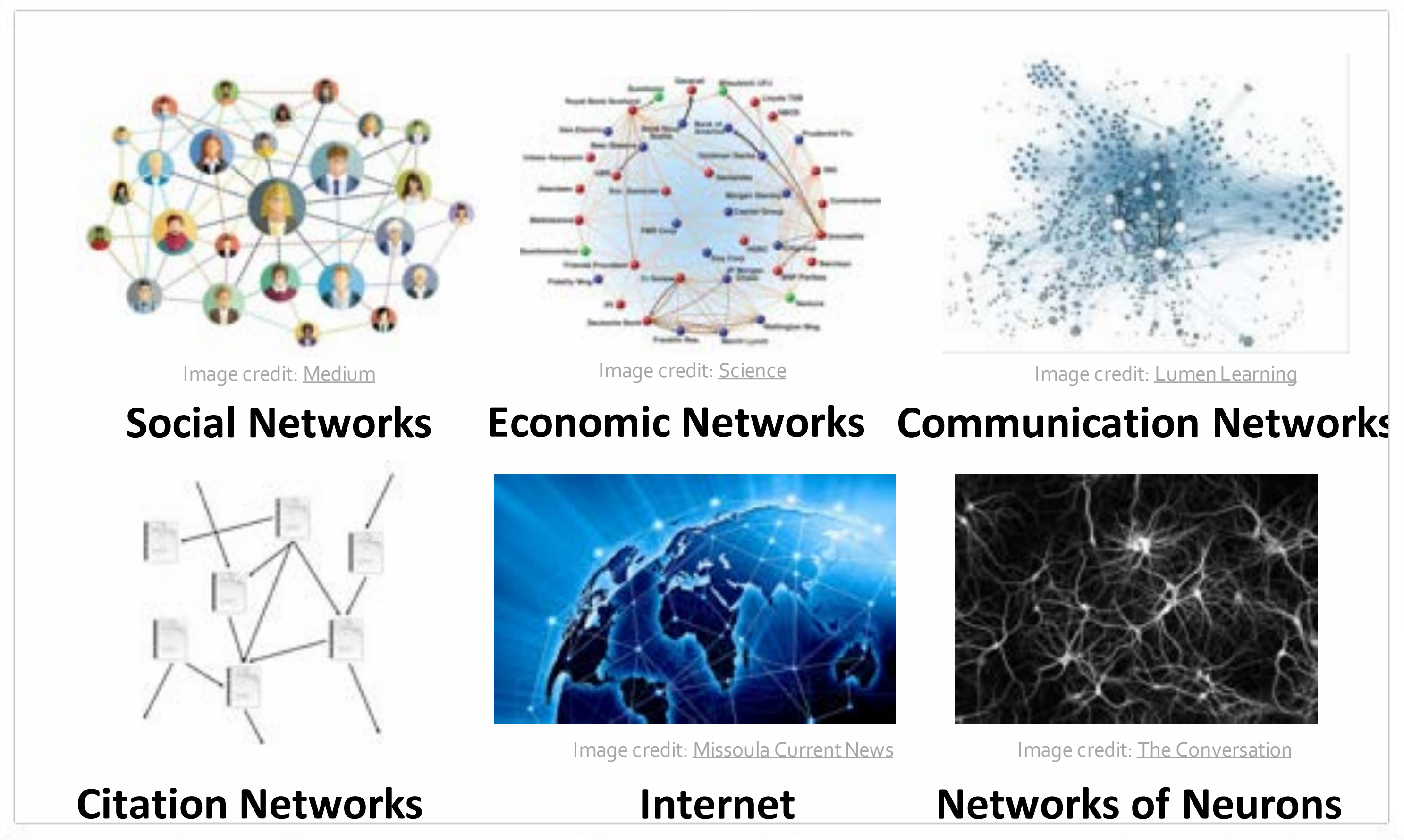


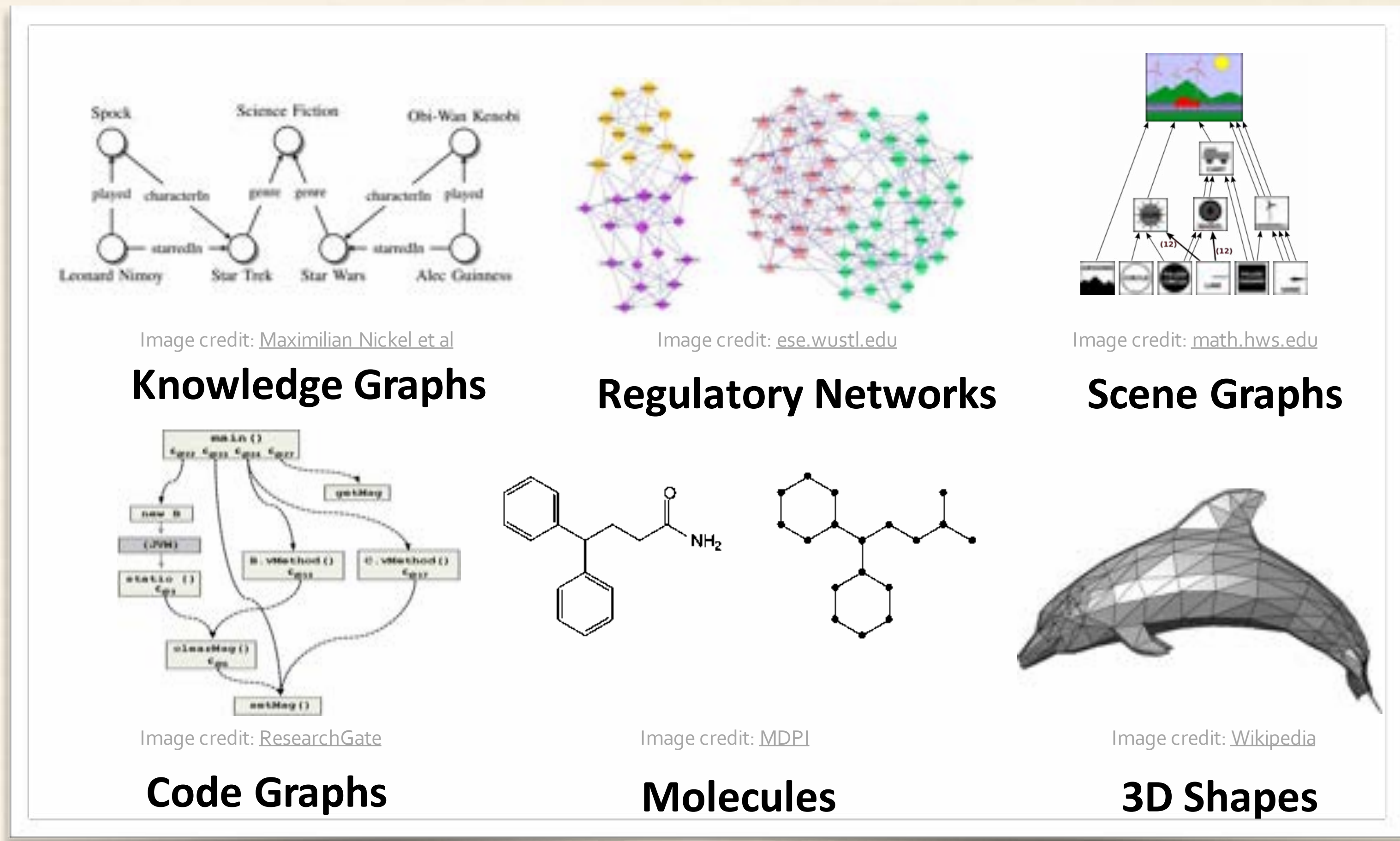
Images: Machine Learning on Graphs, Stanford course Jure Leskovec

Why learning on graphs?

Graphs are everywhere!





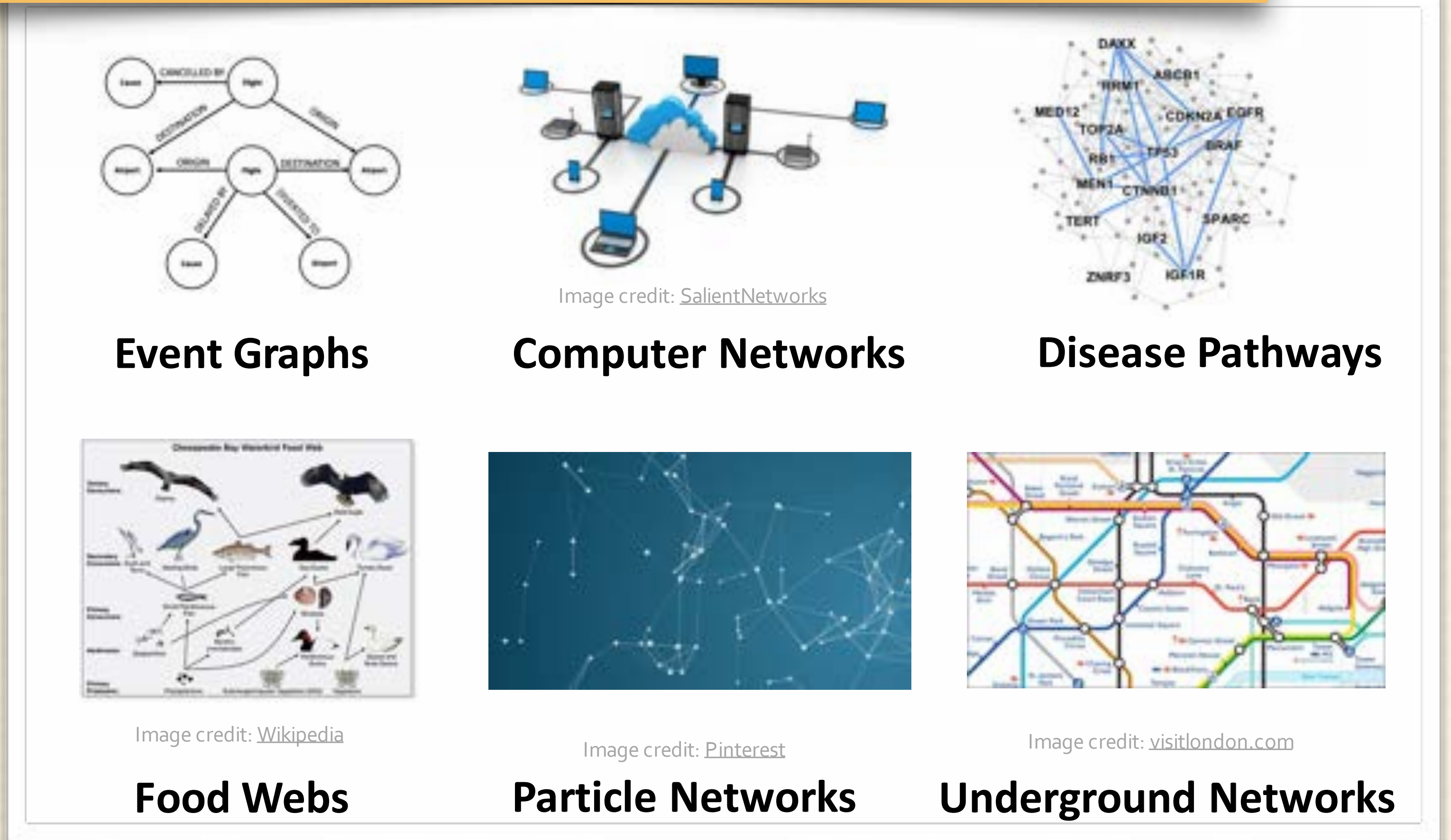


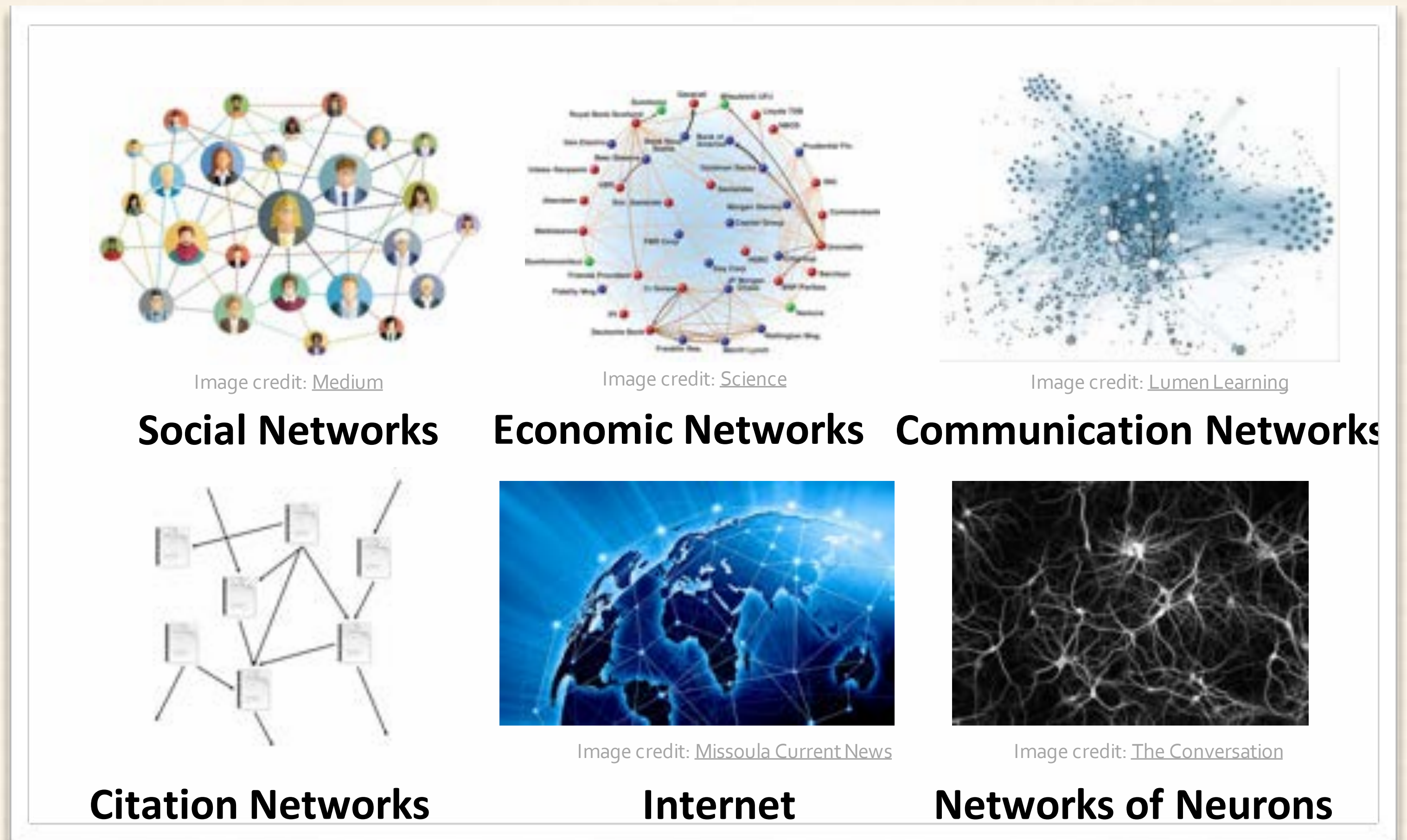
Graph learning methods are thus widely applicable

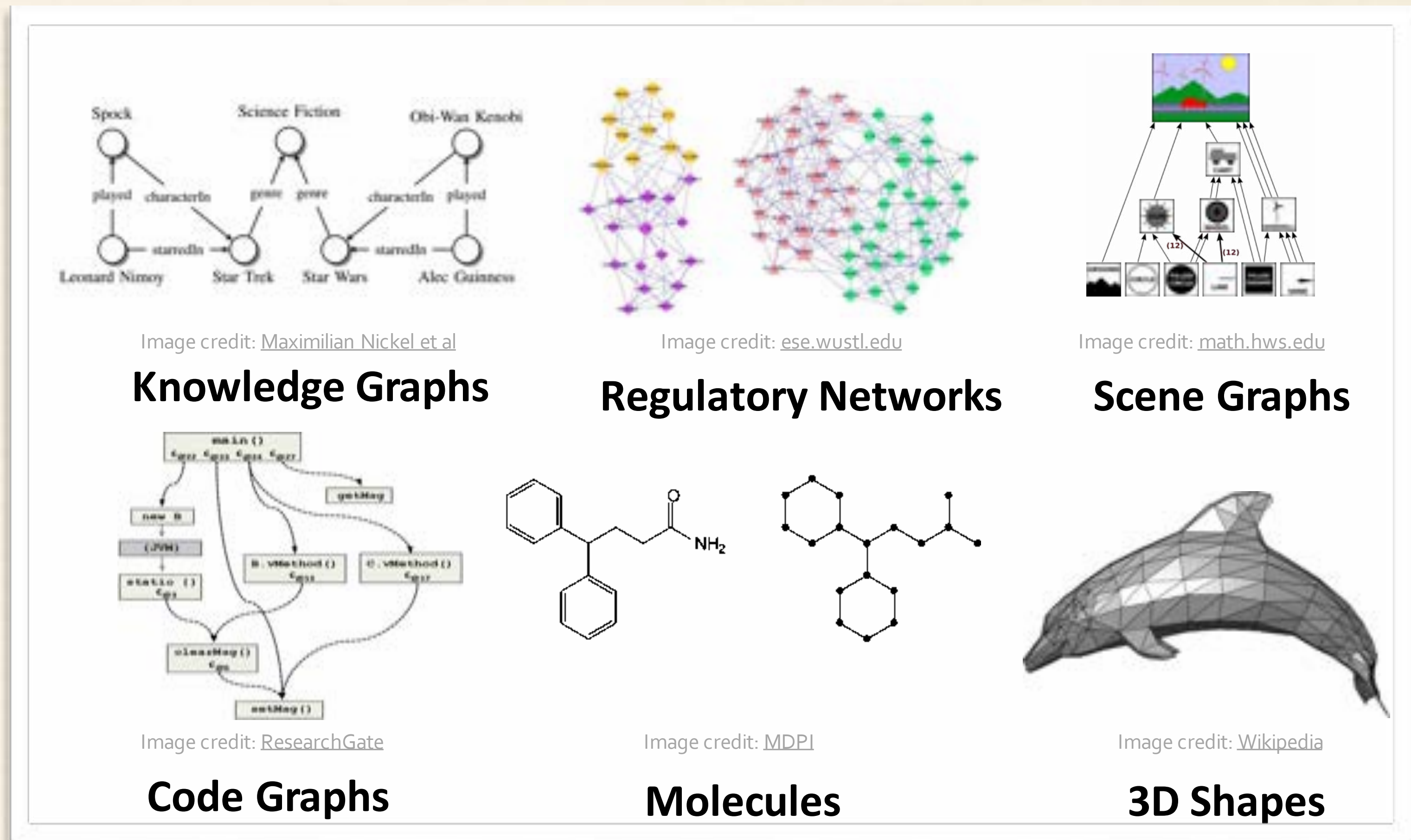
Images: Machine Learning on Graphs, Stanford course Jure Leskovec

Why learning on graphs?

Graphs are everywhere!







Graph learning methods are thus widely applicable

How is learning typical done?

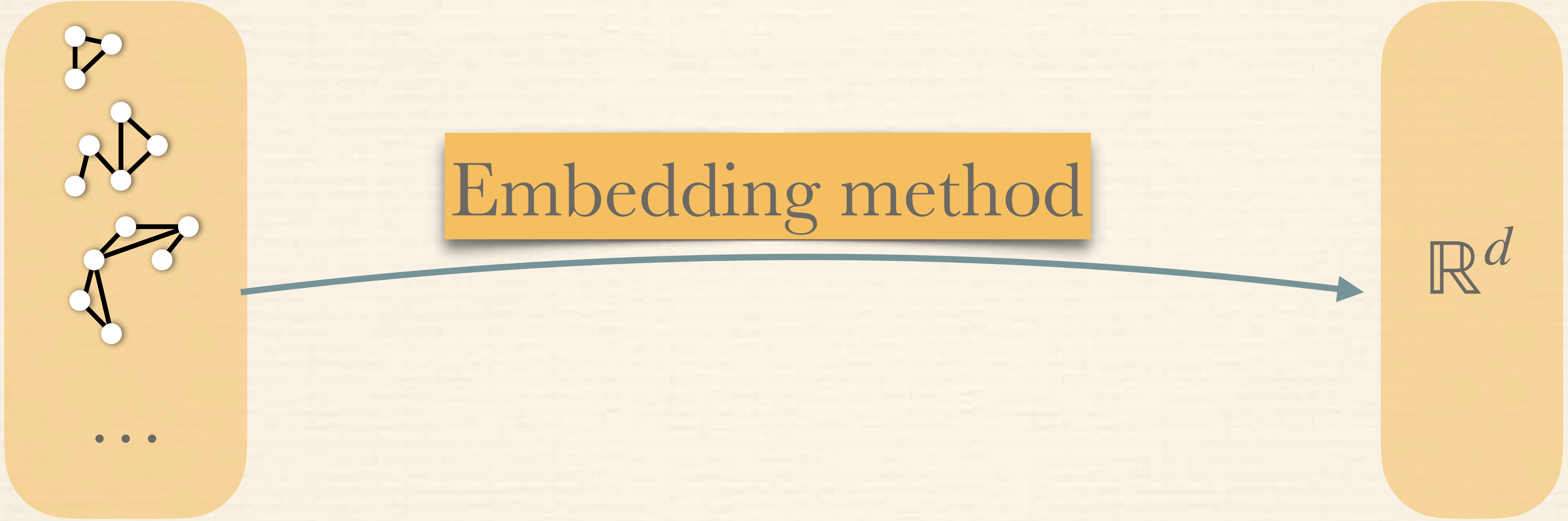
Images: Machine Learning on Graphs, Stanford course Jure Leskovec

Embedding-based graph learning

$\mathcal{G} =$ all graphs

$\mathbb{Y} =$ output space

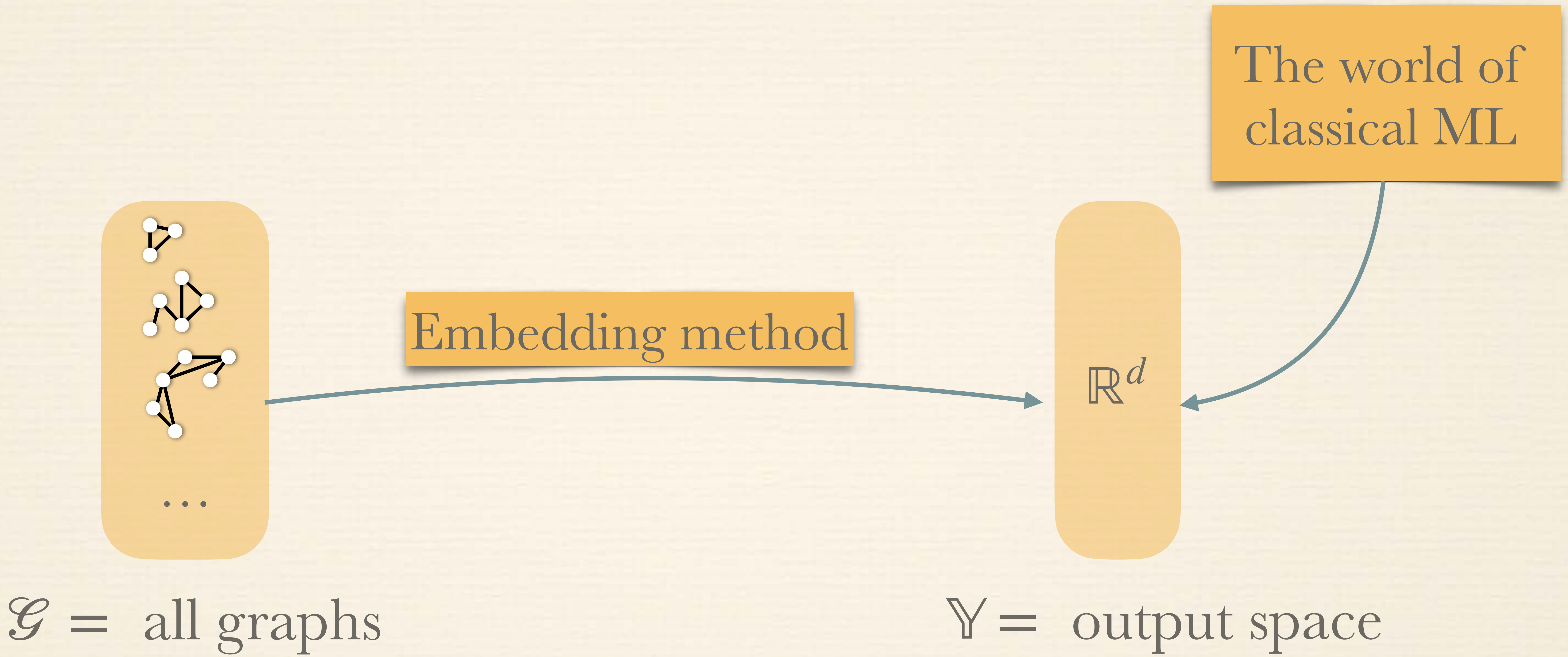
Embedding-based graph learning



$\mathscr{G} =$ all graphs

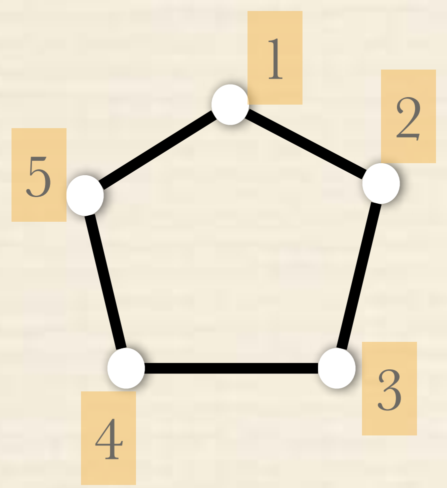
$\mathbb{Y} =$ output space

Embedding-based graph learning



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

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

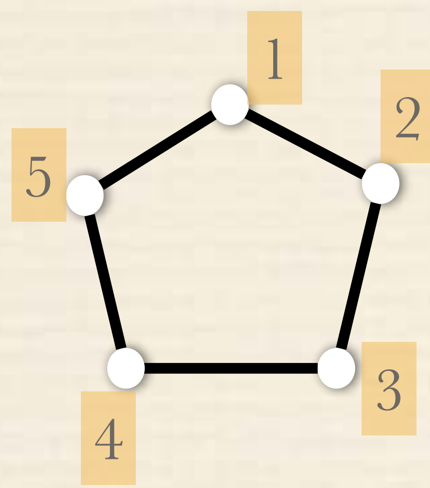
$\longrightarrow$ (0,1,0,0,1,1,0,1,0,0,0,1,0,1,0,0,0,1,0,1,1,0,0,1,0) $\longrightarrow$

- Deep neural network
- Support vector machines
- …

Images: flaticon.com, Noun project

What's new?

$\mathcal{G}$

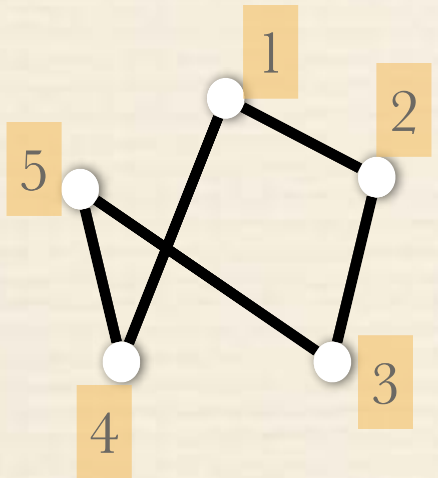


$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

flatten

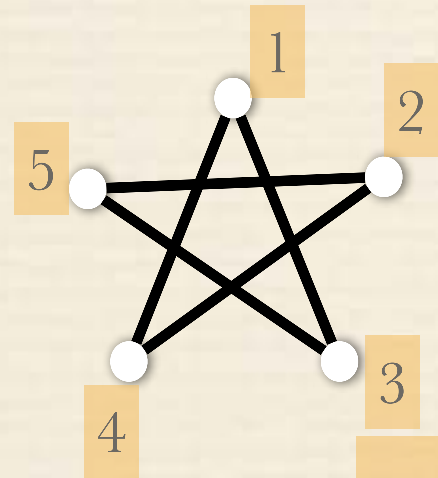
$\mathbb{Y} = \mathbb{R}^{10}$

(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)



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

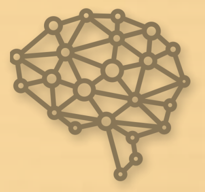
(0,1,0,1,0,1,0,1,0,0,0,1,0,0,1,1,0,0,0,1,0,0,1,1,0)



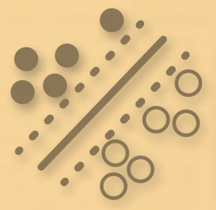
$$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

(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



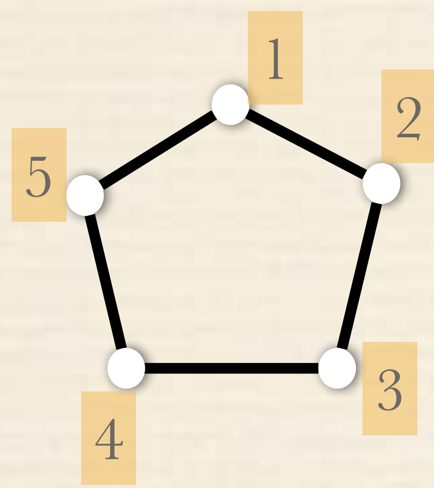
Support vector machines

…

Images: flaticon.com, Noun project

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

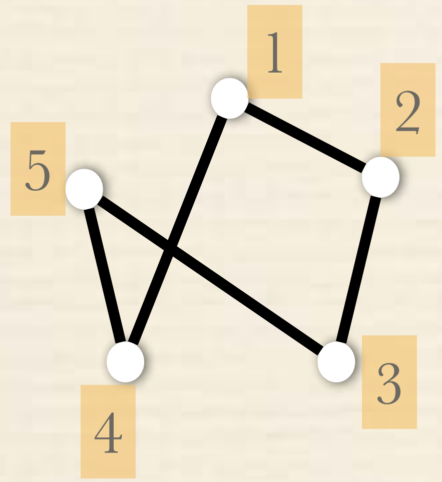
$\mathbb{Y} = \mathbb{R}^{10}$

(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)

Deep neural network

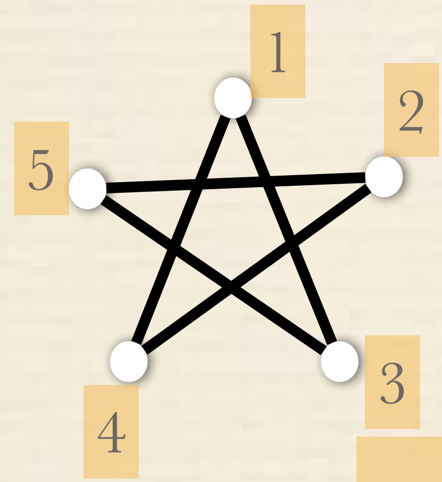
Support vector machines

…



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

(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

Different representation ⇒ different result

Images: flaticon.com, Noun project

Invariant embeddings

- We need embeddings to be graph invariants
- Isomorphic inputs should give the same result

genericity in query languages

Invariant embeddings

- We need embeddings to be graph invariants
- Isomorphic inputs should give the same result

genericity in query languages

p-vertex embedding $\xi : \mathscr{G} \to (\mathscr{V}^p \to \mathbb{Y}) : (G, \mathbf{v}) \mapsto \xi(G, \mathbf{v})$ is invariant if for all G, all isomorphisms π, and $\mathbf{v} \in V_G^p : \xi(G, \mathbf{v}) = \xi(\pi(G), \pi(\mathbf{v}))$

- Invariance is achieved by composing invariant building blocks to build embeddings

Graph learning (semi-supervised)

- Given training set $\mathcal{T}$ and hypothesis class $\mathcal{H}$ of invariant embedding methods $\mathcal{T} := \{(G_1, \mathbf{v}_1, y_1), \ldots, (G_\ell, \mathbf{v}_\ell, y_\ell)\} \subseteq \mathcal{G} \times \mathcal{V}^p \times \mathbb{Y}$
- Empirical risk minimisation: Find embedding ξ in $\mathcal{H}$ which minimises empirical loss on training set $\mathcal{T}$:

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

- Solved using backpropagation/gradient descent like optimisation algorithms

Graph learning (semi-supervised)

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

We know now what graph learning is but what are these hypothesis classes?

Hypothesis classes?

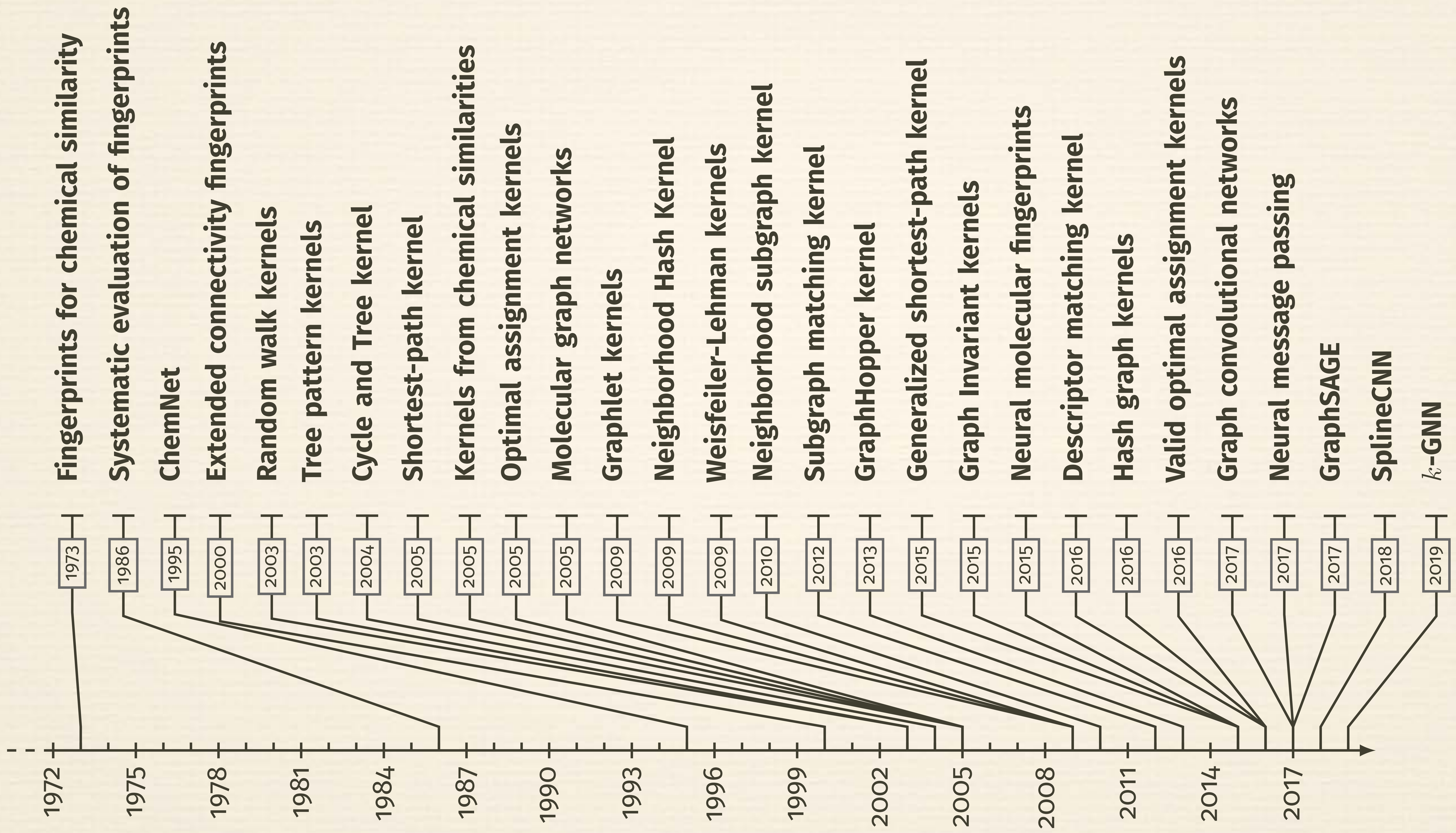


Image: Christopher Morris

Hypothesis classes?

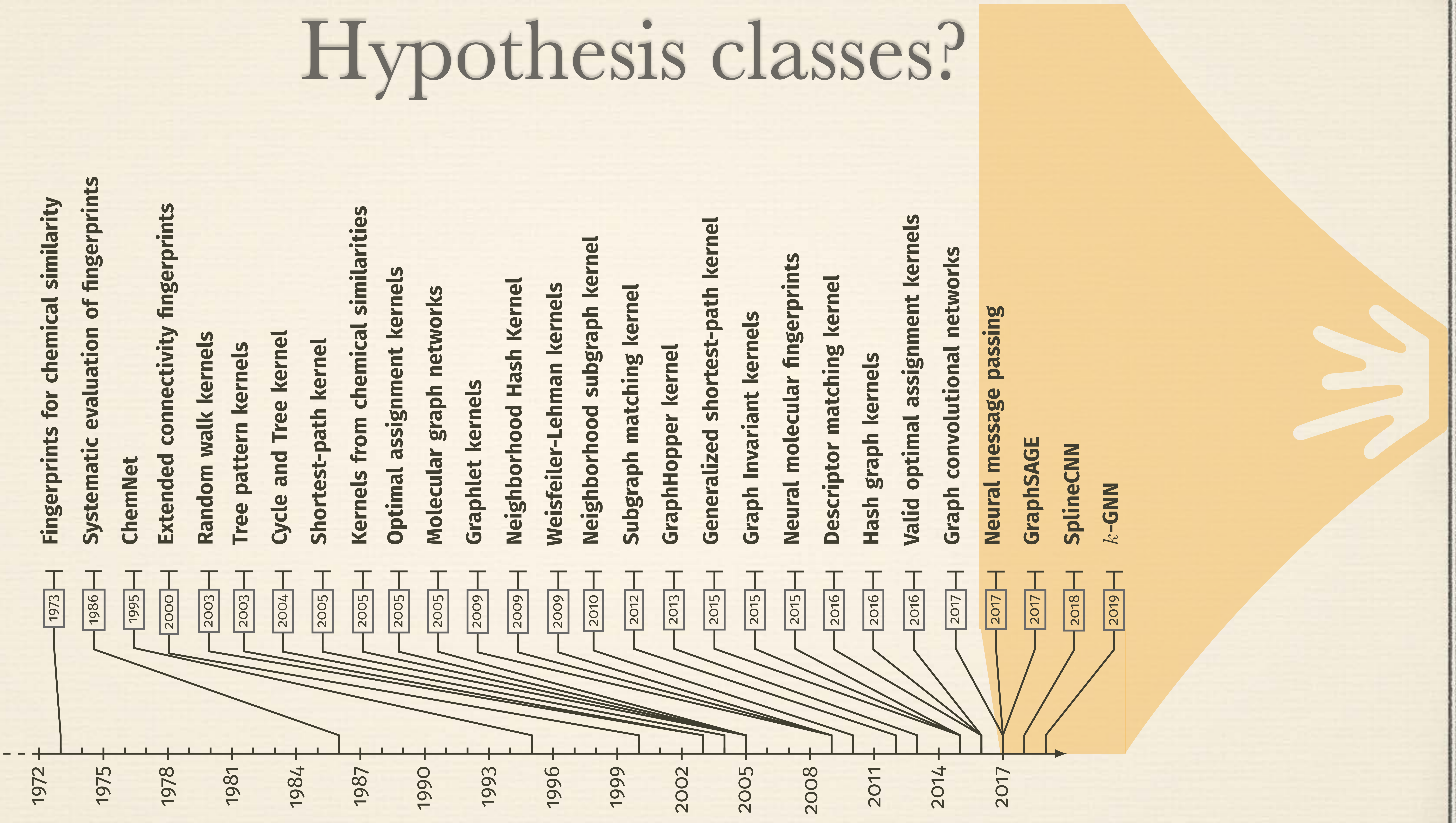


Image: Christopher Morris

Hypothesis classes

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query language
2. Analyse expressive power of query language
3. Transfer understanding back to graph learning world

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

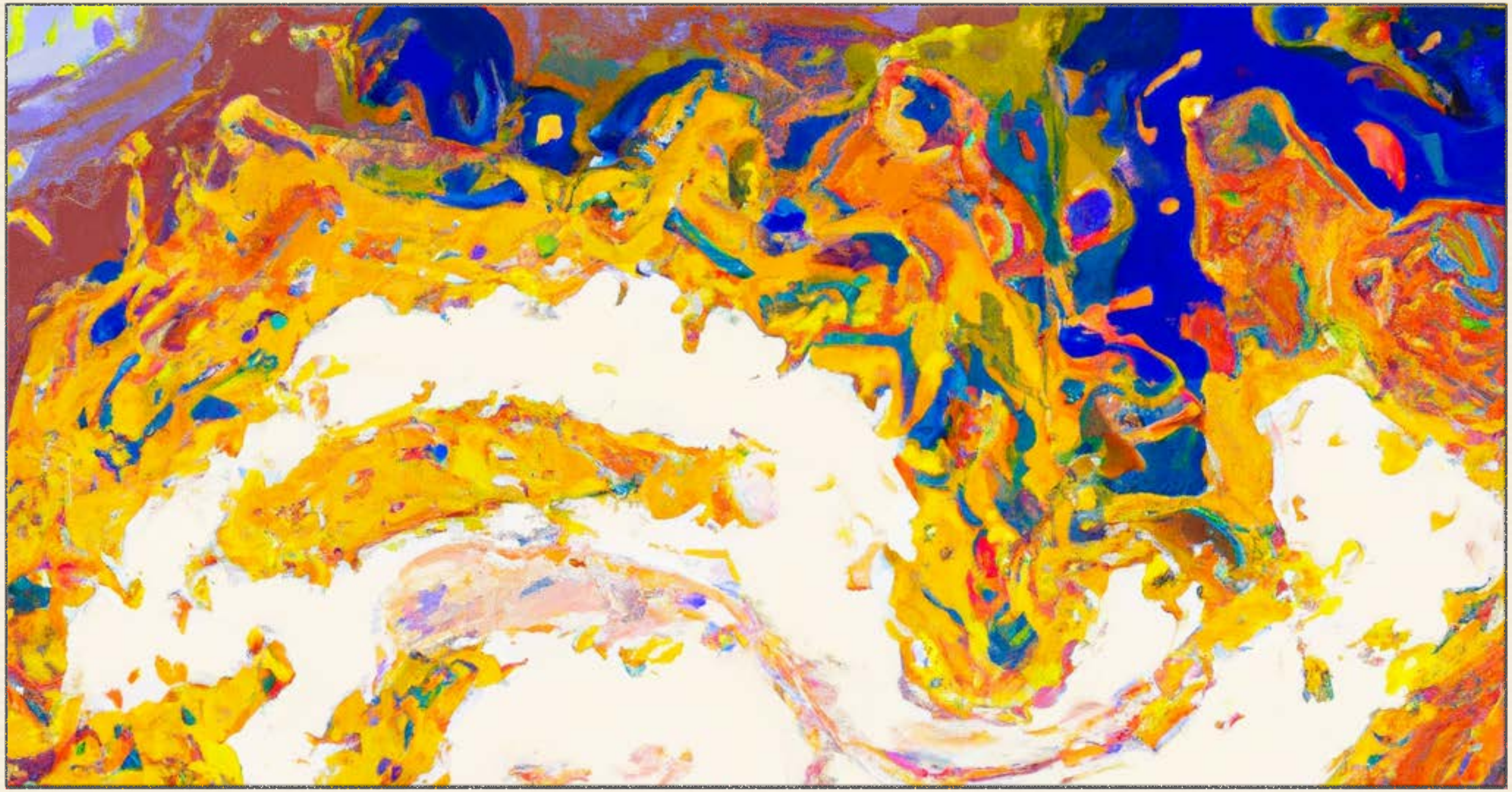
What kind of language?

1. See graph embedding methods as queries in some query language

Expressive power?

2. Analyse expressive power of query language

3. Transfer understanding back to graph learning world



Graph Embedding Language

Graph Embedding Language (GEL)

- Most methods are specified in terms of linear algebra computations interleaved with non-linear function applications
- Crucial component is multiplication with adjacency matrix which corresponds to neighbourhood aggregation

Desired language needs function application and aggregation

Graph Embedding Language (GEL)

- Most methods are specified in terms of linear algebra computations interleaved with non-linear function applications
- Crucial component is multiplication with adjacency matrix which corresponds to neighbourhood aggregation

Desired language needs function application and aggregation

Let us see first see an example of an embedding class $\mathscr{H}$

Graph Neural Networks 101

- Non-linear activation function σ (ReLU, sign, sigmoid, …)
- $\mathbf{F}_G^{(t)} \in \mathbb{R}^{n \times d}$ denotes embedding of vertices in graph G

$$\begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} \begin{pmatrix} 0.1 & 31 & 8 & 4.03 \\ 5 & 0.03 & 9.7 & -1 \\ -3 & 118 & -63 & 0.204 \end{pmatrix}$$

- Weight matrices $\mathbf{W}_1^{(t)} \in \mathbb{R}^{d \times d}$ and $\mathbf{W}_2^{(t)} \in \mathbb{R}^{d \times d}$ and bias vector $\mathbf{b} \in \mathbb{R}^{1 \times d}$

$\mathbf{F}_G^{(0)}$ ← Initial hot-one embedding of vertex labels

Matrix form $$\mathbf{F}_G^{(t)} := \sigma\left(\mathbf{F}_G^{(t-1)}\mathbf{W}_1^{(t)} + \boxed{\mathbf{A}_G}\mathbf{F}_G^{(t-1)}\mathbf{W}_2^{(t)} + \mathbf{B}^{(t)}\right) \in \mathbb{R}^{n \times d}$$

Adjacency matrix

Aggregation over neighbours

GNN 101: Graph embedding

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

$$\mathbf{F}_G := \sigma\left(\sum_{v \in V_G} \mathbf{F}_G^{(L)} \mathbf{W} + \mathbf{b}\right) \in \mathbb{R}^{1 \times d}$$

Aggregation over all vertices

- Hypothesis class $\mathscr{H}$ consists of $\xi_\omega : G \mapsto \mathbf{F}_G$ parametrised by weights

- Empirical Risk Minimisation: Find best parameters

$$\mathbf{W}_1^{(1)}, \ldots, \mathbf{W}_1^{(L)}, \mathbf{W}_2^{(1)} \ldots, \mathbf{W}_2^{((L)}, \mathbf{W}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(L)}, \mathbf{b}$$

Graph Embedding Language (GEL)

GEL expression

Syntax

$\varphi(\mathbf{x})$ of dimension d and free variables $\mathbf{x} = \{x_1, \ldots, x_\ell\}$

Higher order embedding

Semantics

$$\xi_\varphi : \mathscr{G} \to (\mathscr{V}^\ell \to \mathbb{R}^d)$$

It is really just going to be a simple version of a query languages with aggregates studied in database theory and it resembles Datalog°

Hella, Libkin, Nurmonen, Wong: *Logics with Aggregates.* (2001)
Abo Khamis, Ngo, Pichler, Suciu, Wang: *Convergence of Datalog over (Pre-) Semiring*. (2022)
G. and Reutter: *Expressiveness and approximation properties of graph neural networks.* (2022)

Atomic GEL expressions

Atomic expressions

Label: $\varphi(x_i) := \mathsf{Lab}_j(x_i)$ of dim 1 and free var x_i

Edge: $\varphi(x_i, x_j) := E(x_i, x_j)$ of dim 1, free vars x_i, x_j

Equality: $\varphi(x_i, x_j) := \mathbf{1}[x_i = x_j]$ of dim 1, free vars x_i, x_j

Semantics

$\xi_\varphi(G, x_i/v) := j$ th feature of v

$$\xi_\varphi(G, x_i/v, x_j/w) := \begin{cases} 1 & (v, w) \in E_G \\ 0 & \text{otherwise} \end{cases}$$

$$\xi_\varphi(G, x_i/v, x_j/w) := \begin{cases} 1 & v = w \\ 0 & \text{otherwise} \end{cases}$$

GEL: Function Application

Function application: Syntax

Let $\varphi_1(\mathbf{x}_1), \ldots, \varphi_\ell(\mathbf{x}_1)$ be GEL expressions of dim $d_1, \ldots, d_\ell$ and free vars $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$
Let $F : \mathbb{R}^{d_1+\cdots+d_\ell} \to \mathbb{R}^d$ be a function. Then,

$$\varphi(\mathbf{x}) = F(\varphi_1, \ldots, \varphi_\ell)$$

is again a GEL expression of dim d and free vars $\mathbf{x} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_\ell$

GEL: Function Application

Function application: Syntax

Let $\varphi_1(\mathbf{x}_1), \ldots, \varphi_\ell(\mathbf{x}_1)$ be GEL expressions of dim $d_1, \ldots, d_\ell$ and free vars $\mathbf{x}_1, \ldots, \mathbf{x}_\ell$
Let $F : \mathbb{R}^{d_1+\cdots+d_\ell} \to \mathbb{R}^d$ be a function. Then,

$$\varphi(\mathbf{x}) = F(\varphi_1, \ldots, \varphi_\ell)$$

is again a GEL expression of dim d and free vars $\mathbf{x} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_\ell$

Semantics

$$\xi_\varphi(G, \mathbf{x}/\mathbf{v}) := F\Big(\underset{\mathbb{R}^{d_1}}{\underset{\Cap}{\xi_{\varphi_1}(G, \mathbf{x}_1/\mathbf{v}_1)}}, \ldots, \underset{\mathbb{R}^{d_\ell}}{\underset{\Cap}{\xi_{\varphi_\ell}(G, \mathbf{x}_p/\mathbf{v}_p)}}\Big) \in \mathbb{R}^d$$

Linear algebra
Activation functions
Anything you want…

GEL: Aggregation

Aggregation: Syntax

Let $\varphi_1(\mathbf{x}, \mathbf{y})$ and $\varphi_2(\mathbf{x}, \mathbf{y})$ be GEL expressions of dim d_1 and d_2 and free vars $\mathbf{x}, \mathbf{y}$. Let Θ be a function mapping bags of vectors in $\mathbb{R}^{d_1}$ to a vector in $\mathbb{R}^{d}$. Then,

$$\varphi(\mathbf{x}) = \mathsf{agg}_{\mathbf{y}}^{\Theta}[\varphi_1 \mid \varphi_2]$$

is a GEL expression of dim d and free vars $\mathbf{x}$

Semantics

$$\xi_\varphi(G, \mathbf{x}/\mathbf{v}) := \Theta\Big(\{\!\!\{ \underbrace{\xi_{\varphi_1}(G, \mathbf{x}/\mathbf{v}, \mathbf{y}/\mathbf{w})}_{\in\, \mathbb{R}^{d_1}} \mid \qquad\qquad \mathbf{w} \in V_G^{|\mathbf{y}|} \}\!\!\} \Big)$$

GEL: Aggregation

Aggregation: Syntax

Let $\varphi_1(\mathbf{x}, \mathbf{y})$ and $\varphi_2(\mathbf{x}, \mathbf{y})$ be GEL expressions of dim d_1 and d_2 and free vars $\mathbf{x}, \mathbf{y}$. Let Θ be a function mapping bags of vectors in $\mathbb{R}^{d_1}$ to a vector in $\mathbb{R}^{d}$. Then,

$$\varphi(\mathbf{x}) = \mathsf{agg}_{\mathbf{y}}^{\Theta}[\varphi_1 \mid \varphi_2]$$

is a GEL expression of dim d and free vars $\mathbf{x}$

Semantics

$$\xi_\varphi(G, \mathbf{x}/\mathbf{v}) := \Theta\Big(\{\!\!\{ \underset{\substack{\cap \\ \mathbb{R}^{d_1}}}{\xi_{\varphi_1}(G, \mathbf{x}/\mathbf{v}, \mathbf{y}/\mathbf{w})} \mid \underbrace{\xi_{\varphi_2}(G, \mathbf{x}/\mathbf{v}, \mathbf{y}/\mathbf{w}) \neq \mathbf{0}}_{\text{guard}} \; \mathbf{w} \in V_G^{|\mathbf{y}|} \}\!\!\} \Big)$$

GEL: Aggregation example

$$\varphi = \mathsf{agg}^{\mathsf{sum}}_{x,y,z}\left[\mathbf{1}[y = y] \mid E(x,y)\cdot E(y,z)\cdot E(x,z)\cdot \mathbf{1}[x \neq y]\cdot \mathbf{1}[x \neq z]\cdot \mathbf{1}[y \neq z]\right]$$

$\cdot$ = shorthand for product function application

What does this compute?

GEL: Aggregation example

$$\varphi = \mathsf{agg}^{\mathsf{sum}}_{x,y,z}\left[\mathbf{1}[y = y] \mid E(x, y) \cdot E(y, z) \cdot E(x, z) \cdot \mathbf{1}[x \neq y] \cdot \mathbf{1}[x \neq z] \cdot \mathbf{1}[y \neq z]\right]$$

$\cdot$ = shorthand for product function application

What does this compute? Triangle count

GEL: Aggregation example

$$\varphi = \mathsf{agg}^{\mathsf{sum}}_{x,y,z}\left[\mathbf{1}[y = y] \mid E(x,y) \cdot E(y,z) \cdot E(x,z) \cdot \mathbf{1}[x \neq y] \cdot \mathbf{1}[x \neq z] \cdot \mathbf{1}[y \neq z]\right]$$

$\cdot$ = shorthand for product function application

What does this compute? Triangle count

Let us see a more elaborate example

Message Passing Neural Networks

We define $\varphi^{(0)}(x_1) := \mathbf{1}[x_1 = x_1]$
Then for $t > 0$, we get

$$\varphi^{(t)}(x_1) := \mathsf{Upd}^{(t)}\Big(\varphi^{(t-1)}(x_1), \mathsf{agg}^{\Theta^{(t)}}_{x_2}\big[\varphi^{(t-1)}(x_2) \mid E(x_1, x_2)\big]\Big)$$

For readout layer, we get

$$\varphi := \mathsf{agg}^{\Theta}_{x_1}\big[\varphi^{(L)}(x_1) \mid \mathbf{1}[x_1 = x_1]\big]\Big)$$

dummy guard

edge guarded aggregation

This encompasses the GNNs 101

Gilmer, Schoenholz, Riley, Vinyals, Dahl.: *Neural message passing for quantum chemistry.* (2017)

Fragments of GEL

- $\mathrm{GEL}_k(\Omega, \Theta)$: k variable fragment of GEL with functions in Ω and aggregations in Θ

- $\mathrm{GGEL}_2(\Omega, \Theta)$: 2 variable fragment GEL with edge guarded aggregation only

MPNNs without readout phase fit in $\mathrm{GGEL}(\Omega, \Theta)$
MPNNs including readout phase fit in $\mathrm{GEL}_2(\Omega, \Theta)$

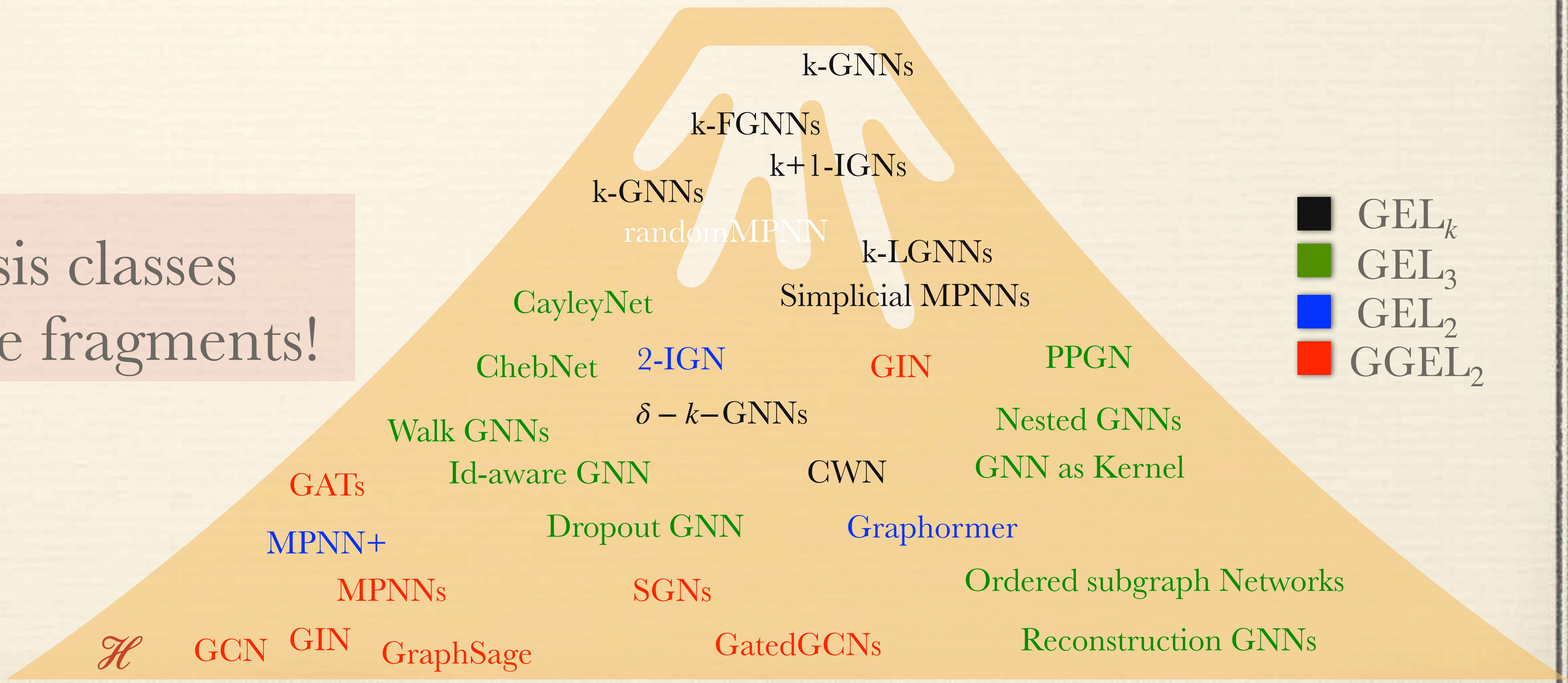
Fragments of GEL

- $\mathrm{GEL}_k(\Omega, \Theta)$: k variable fragment of GEL with functions in Ω and aggregations in Θ
- $\mathrm{GGEL}_2(\Omega, \Theta)$: 2 variable fragment GEL with edge guarded aggregation only

Fragments of GEL

- $\mathrm{GEL}_k(\Omega, \Theta)$: k variable fragment of GEL with functions in Ω and aggregations in Θ
- $\mathrm{GGEL}_2(\Omega, \Theta)$: 2 variable fragment GEL with edge guarded aggregation only

Most hypothesis classes
fit in one of those fragments!



G., Reutter: *Expressiveness and approximation properties of graph neural networks.* (2022)

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query language
2. Analyse expressive power of query language
3. Transfer understanding back to graph learning world

Expressive power?

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

Which language?

1. See graph embedding methods as queries in some query language

Expressive power?

2. Analyse expressive power of query language

3. Transfer understanding back to graph learning world

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query language

GEL_k
GGEL_2

2. Analyse expressive power of query language

Expressive power?

3. Transfer understanding back to graph learning world

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query language

GEL_k
GGEL_2

2. Analyse expressive power of query language

Expressive power?

3. Transfer understanding back to graph learning world

Expressive power

Distinguishing power

- Which inputs can be separated/distinguished by embeddings in $\mathcal{H}$?
- Captured by the following equivalence relation on $\mathcal{G} \times \mathcal{V}^p$:

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

Distinguishing power

- Which inputs can be separated/distinguished by embeddings in $\mathscr{H}$?

- Captured by the following equivalence relation on $\mathscr{G} \times \mathscr{V}^p$:

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

- Strongest power: $\mathscr{H}$ powerful enough to detect non-isomorphic graphs: $\rho(\mathscr{H})$ only contains isomorphic pairs

- Weakest power: $\mathscr{H}$ cannot differentiate any two graphs: $\rho(\mathscr{H})$ contains all pairs of graphs.

Distinguishing power

- Allows for comparing different classes of embeddings methods

$$\rho(\text{methods}_1) \subseteq \rho(\text{methods}_2)$$

methods$_1$ is more powerful than methods$_2$
methods$_2$ is bounded by methods$_1$ in power

$$\rho(\text{methods}_1) = \rho(\text{methods}_2)$$

Both methods are as powerful

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

Expressive power in ML community

- Focus has been on characterising the distinguishing power of classes $\mathscr{H}$ of embedding methods.
- Hopefully, characterisations of $\rho(\mathscr{H})$ shed light on what graph properties a learning method in $\mathscr{H}$ can detect/use.

Expressive power in ML community

- Focus has been on characterising the distinguishing power of classes $\mathscr{H}$ of embedding methods.
- Hopefully, characterisations of $\rho(\mathscr{H})$ shed light on what graph properties a learning method in $\mathscr{H}$ can detect/use.

We will obtain logic-based characterisations

Logic

- First-order logic with k variables and counting quantifiers (C_k).

k=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 each with at least two neighbours labeled "a"

- Guarded fragment GC_2 of C_2

only existential quantification for the form $\exists^{\geq n} y (E(x,y) \wedge \varphi(y))$

Expressive power of GEL

- The following results follow from standard analysis of aggregate query languages: all real number arithmetic can be eliminated.

Theorem (Xu et al. 2019, Morris et al. 2019, G. and Reutter 2022)

$$\rho(\mathrm{GGEL}(\Omega, \Theta)) = \rho(\mathrm{GC}_2)$$

Theorem (G. and Reutter 2022)

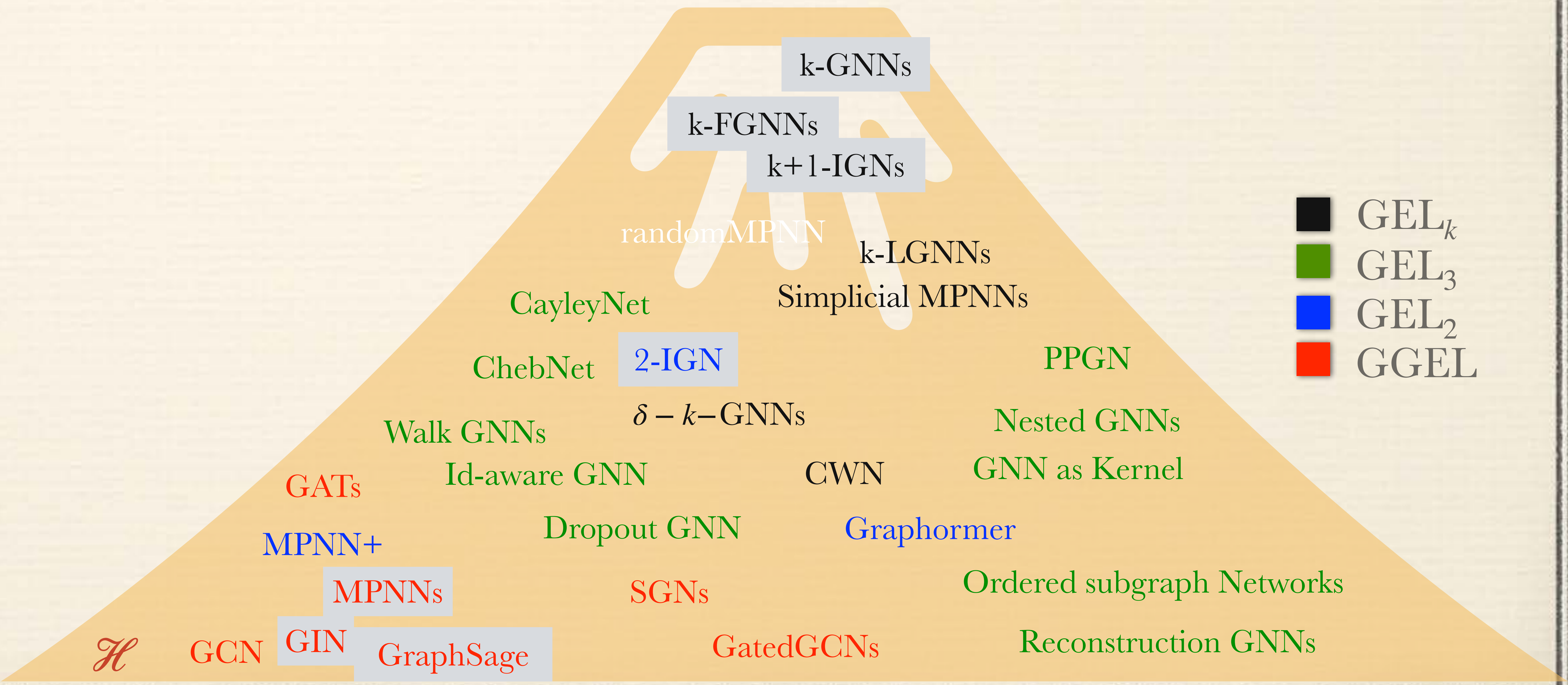
$$\rho(\mathrm{GEL}_k(\Omega, \Theta)) = \rho(\mathsf{C}_k)$$

- Lower bounds: Ω contains linear combinations, concatenation, product (or activation functions) and Θ contains summation

Xu, Hu, Leskovec, Jegelka: *How powerful are graph neural networks?* (2019)
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019
Hella, Libkin, Nurmonen, Wong: *Logics with Aggregates.* (2001)
Cai, Fürer, Immerman: *An optimal lower bound on the number of variables for graph identification.* (1992)
G., Reutter: *Expressiveness and approximation properties of graph neural networks.* (2022)
M. Grohe: *The logic of graph neural networks.* (2021)

Consequences

- If embedding method M can be cast in $\mathrm{GEL}_k(\Omega, \Theta)$ then $\rho(\mathsf{C}_k) \subseteq \rho(M)$
- If embedding method M can also encode formulas in C_k then $\rho(\mathsf{C}_k) \supseteq \rho(M)$

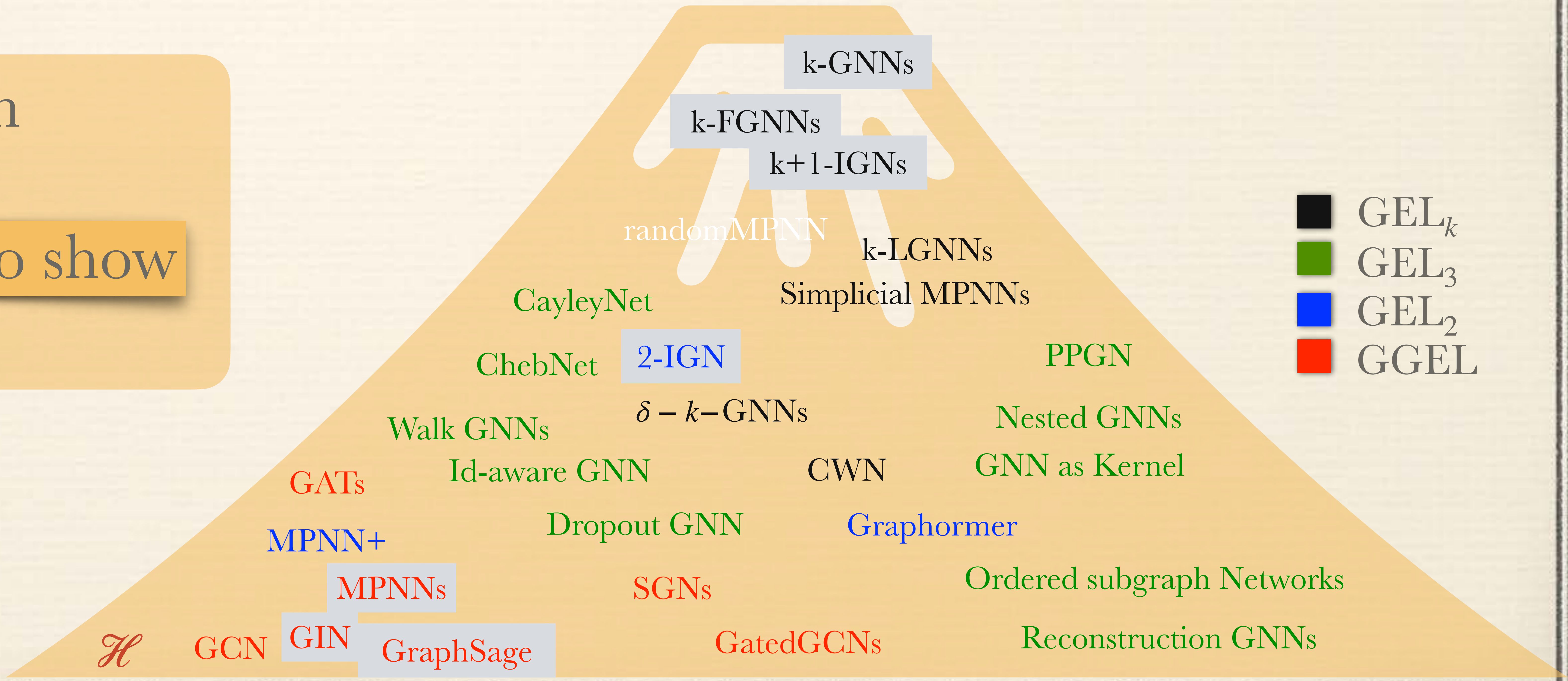


Xu, Hu, Leskovec, Jegelka: *How powerful are graph neural networks?* (2019)
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019
Maron, Ben-Hamu, Serviansky, Lipman: *Provably powerful graph networks.* (2019)
Maron, Fetaya, Segol, Lipman: *Invariant and equivariant graph networks.* (2019)
G. *The expressive power of kth-order invariant graph networks.* (2019)
G.,Reutter: *Expressiveness and approximation properties of graph neural networks.* (2022)

Consequences

- If embedding method M can be cast in $\mathrm{GEL}_k(\Omega,\Theta)$ then $\rho(\mathsf{C}_k) \subseteq \rho(M)$
- If embedding method M can also encode formulas in C_k then $\rho(\mathsf{C}_k) \supseteq \rho(M)$

"Automatic" upper bounds on distinguishing power.
Needs case-by-case analyse to show "hardness"



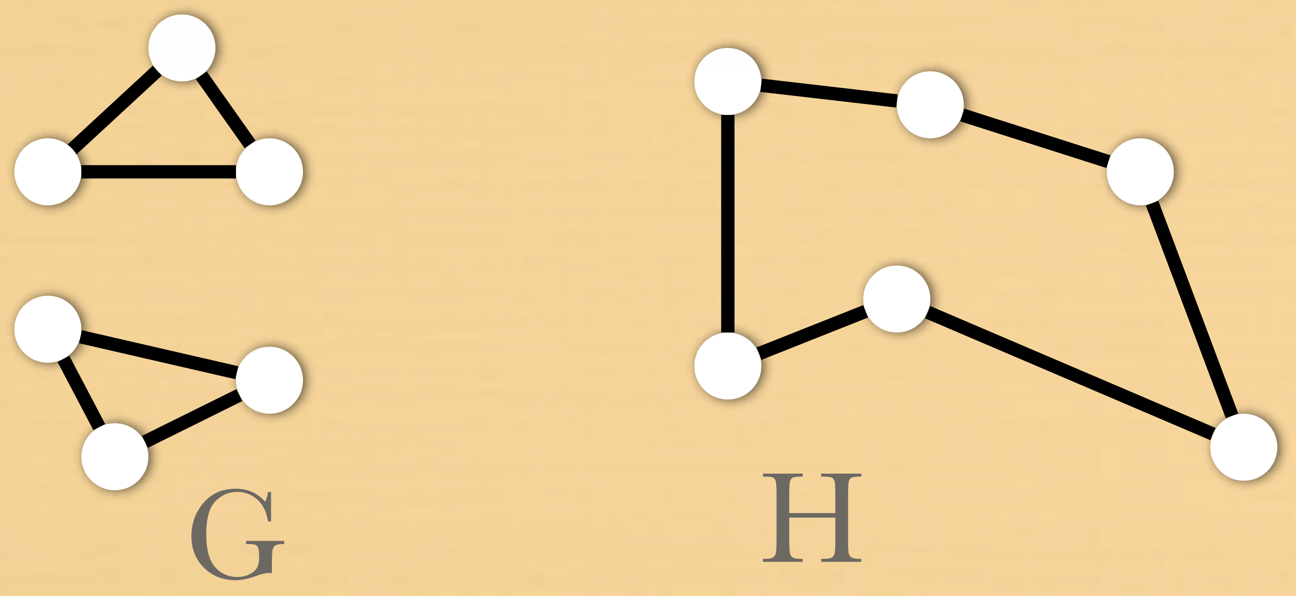
Xu, Hu, Leskovec, Jegelka: *How powerful are graph neural networks?* (2019)
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019
Maron, Ben-Hamu, Serviansky, Lipman: *Provably powerful graph networks.* (2019)
Maron, Fetaya, Segol, Lipman: *Invariant and equivariant graph networks.* (2019)
G. *The expressive power of kth-order invariant graph networks.* (2019)
G.,Reutter: *Expressiveness and approximation properties of graph neural networks.* (2022)

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\text{GNN101}) = \rho(\mathrm{C}_2)$$

GNN 101s, MPNNs are pretty weak



Can we train a GNN 101 which embeds G differently from H?

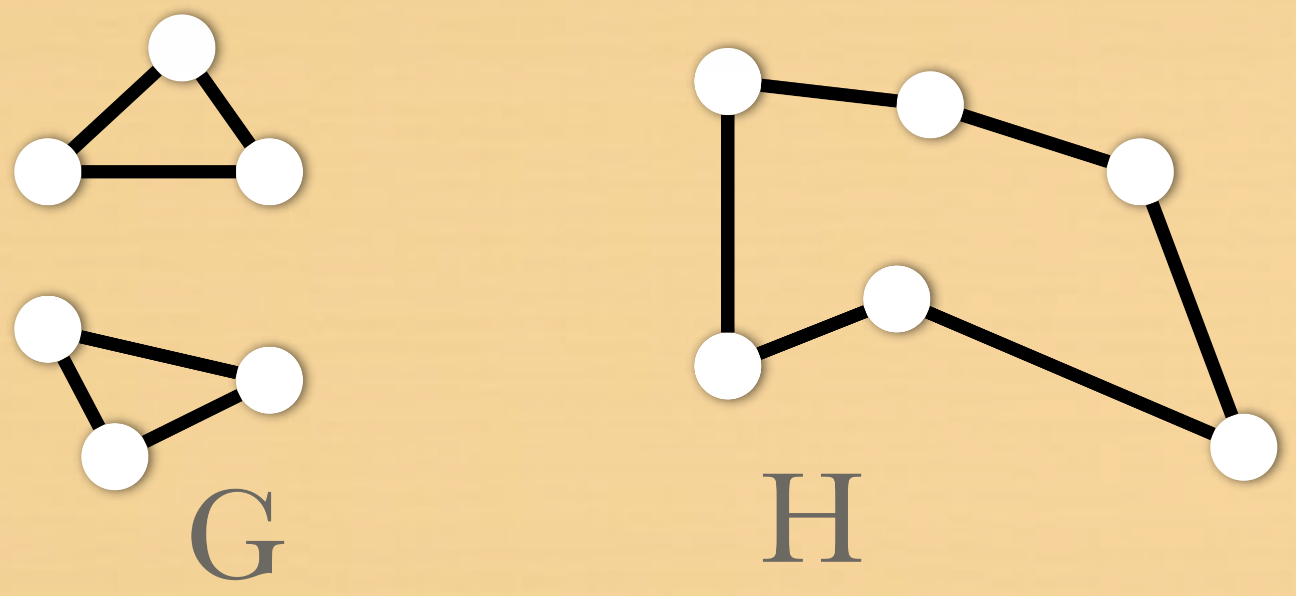
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\text{GNN101}) = \rho(\mathrm{C}_2)$$

GNN 101s, MPNNs are pretty weak



Can we train a GNN 101 which embeds G differently from H?

NO!

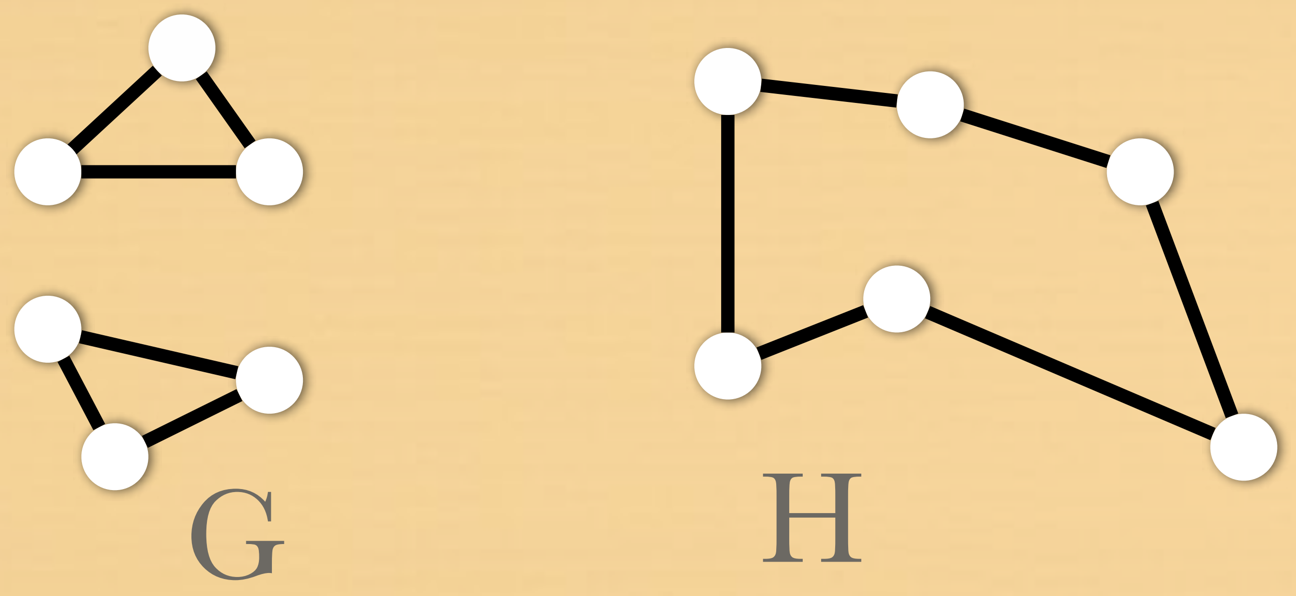
Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\mathrm{GNN101}) = \rho(\mathsf{C}_2)$$

GNN 101s, MPNNs are pretty weak



Can we train a GNN 101 which embeds G differently from H?

NO!

G and H are known to be indistinguishable by C_2 (pebble game argument)

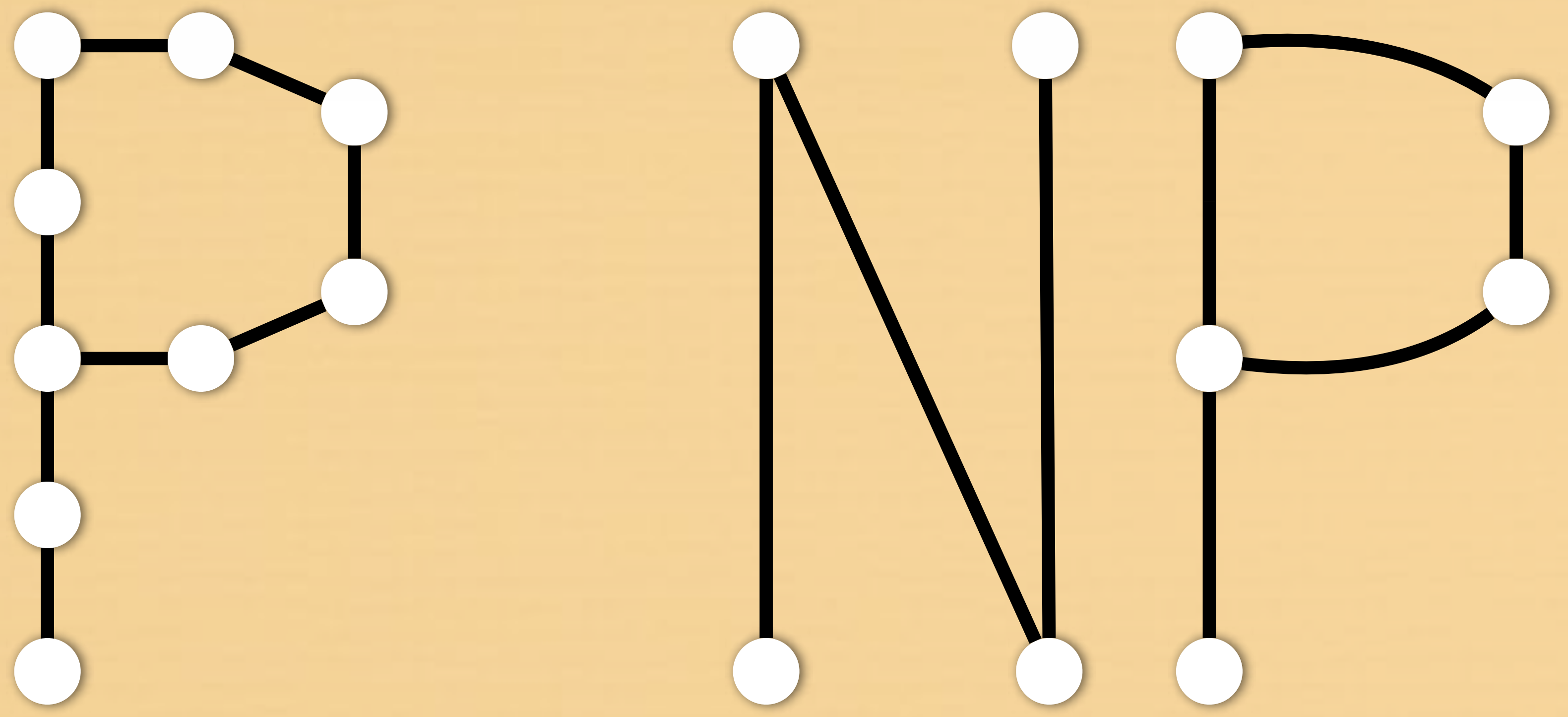
$$\Rightarrow (G, H) \in \rho(\mathsf{C}_2) = \rho(\mathrm{GNN101})$$

Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\mathrm{GNN101}) = \rho(\mathrm{C}_2)$$



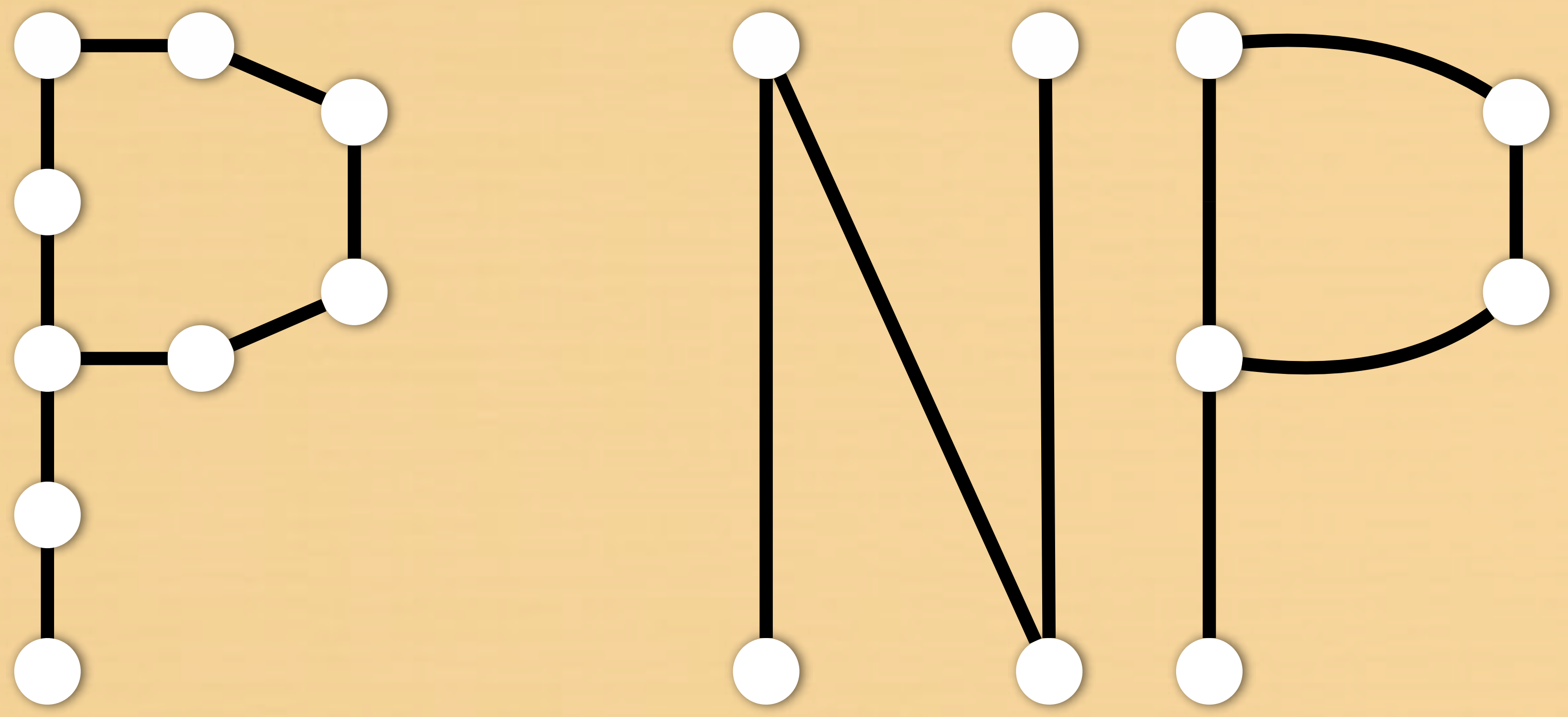
Can we train a GNN101 such that P embeds differently from NP?

Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\mathrm{GNN101}) = \rho(\mathrm{C}_2)$$



Can we train a GNN101 such that P embeds differently from NP?

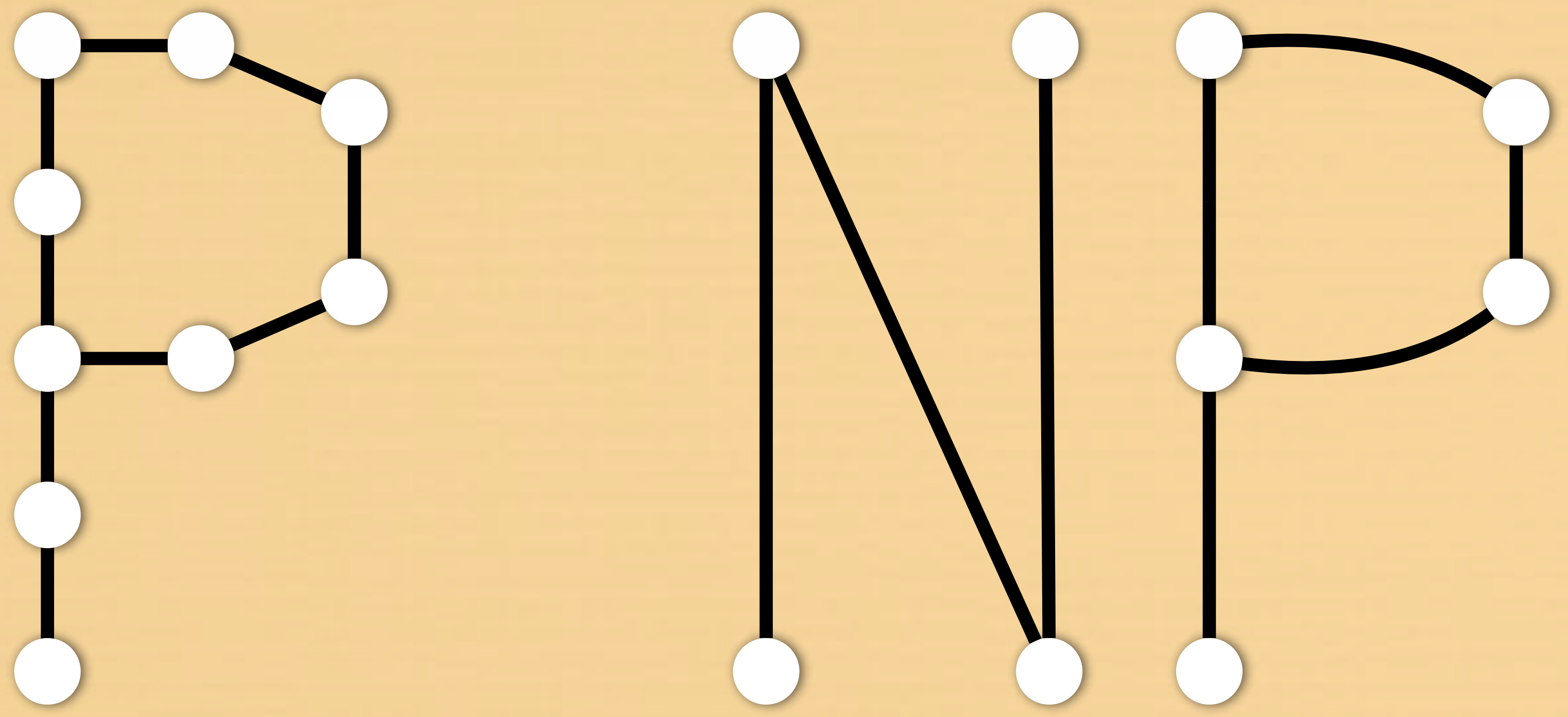
YES!

Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

GNN 101

Theorem (Morris et al. 2019)

$$\rho(\mathrm{GNN101}) = \rho(\mathsf{C}_2)$$



Can we train a GNN101 such that P embeds differently from NP?

YES!

single degree one node

P satisfies $\exists^{=1} x \, \exists^{=1} y \, E(x, y)$ but NP does not

$$\Downarrow$$

$$(P, NP) \notin \rho(\mathsf{C}_2) \Rightarrow (P, NP) \notin \rho(\mathrm{GNN101})$$

Morris, Ritzert, Fey, Hamilton, Lenssen, Rattan, Grohe: *Weisfeiler and Leman go neural: Higher-order graph neural networks.* (2019

Beautiful connections

Theorem (Dell et al. 2019, Dvorák 2010)

$$(G, H) \in \rho(\mathsf{C}_2)$$
if and only if
$$\mathsf{hom}(T, G) = \mathsf{hom}(T, H) \text{ for all } \textit{trees } T$$

Important class of MPNNs can only detect tree-based information

Theorem (Dell et al. 2019, Dvorák 2010)

$$(G, H) \in \rho(\mathsf{C}_{p+1})$$
if and only if
$$\mathsf{hom}(P, G) = \mathsf{hom}(P, H) \text{ for all } \textit{graphs } P \textit{ of treewidth } p$$

- Also connections to the combinatorial graph algorithms color refinement and higher-dimensional Weisfeiler-Leman graph isomorphism tests.

Z. Dvoräk: *On recognizing graphs by numbers of homomorphisms.* (2010)
Dell, Grohe, Rattan: *Lovász meets Weisfeiler and Leman.* (2018)
Cai, Fürer, Immerman: *An optimal lower bound on the number of variables for graph identification.* (1992)
M. Grohe: *The logic of graph neural networks.* (2021)

Expressive power

- Which inputs can be separated/distinguished by embeddings in $\mathscr{H}$.
- Which embeddings can be approximated by embeddings in $\mathscr{H}$?

$\mathscr{H}$ = class of embedding methods

Approximation properties

- Equip the 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.

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

Approximation properties

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

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

Stone-Weierstrass

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

If $\mathcal{H}$ is closed under linear combinations and product, then $\mathcal{H}$ can approximate any continuous function $\Xi : \mathcal{C} \to \mathbb{R}$ satisfying
$$\rho(\mathcal{H}) \subseteq \rho(\{\Xi\}) .$$

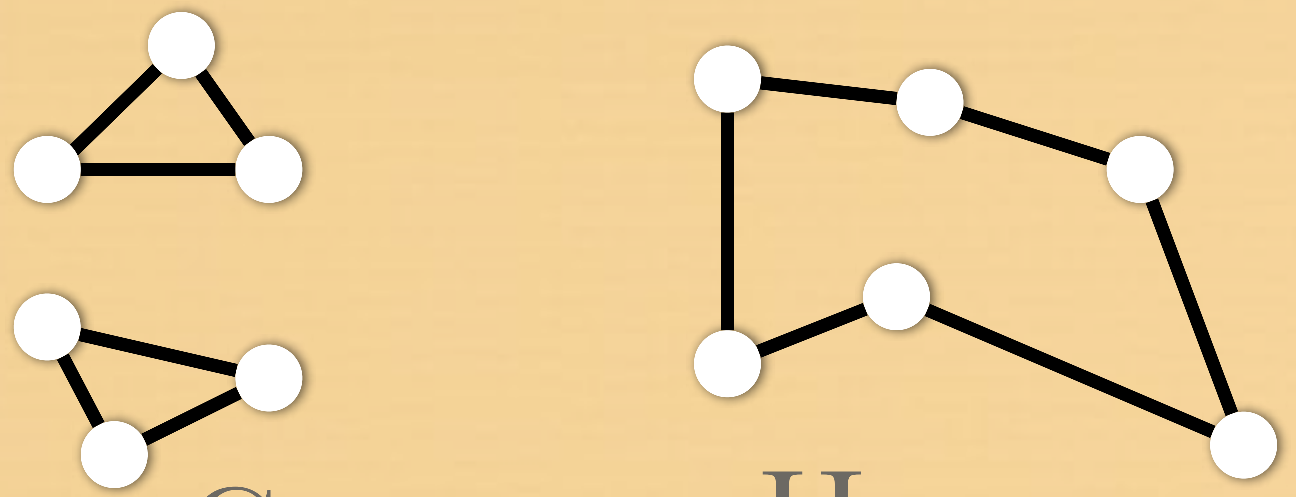
- Can be generalised to embeddings with output space $\mathbb{R}^d$

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

MPNNs: Approximation

Theorem

On compact set of graphs, MPNNs can approximate any continuous graph embedding $\Xi : \mathscr{C} \to \mathbb{R}$ satisfying $\rho(\mathsf{C}_2) \subseteq \rho(\{\Xi\})$



G H

$(G, H) \in \rho(\mathrm{MPNN}) \Rightarrow$ Cannot approximate graph functions based on

- connected components
- 3-cliques
-

- Intricate relation between distinguishing power and approximation properties

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

Expressive power

- Which inputs can be separated/distinguished by embeddings in $\mathscr{H}$.
- Which embeddings can be approximated by embeddings in $\mathscr{H}$?
- What is the VC dimension of $\mathscr{H}$?

$\mathscr{H}$ = class of embedding methods

VC dimension

- A set of graphs $G_1, \ldots, G_s$ can be shattered by $\mathscr{H}$ if for any boolean vector $\tau \in \{0,1\}^s$, there is a $\xi_\tau \in \mathscr{H}$ such that $\xi_\tau(G_i) = \tau_i$ for all $i = 1,\ldots, s$
- We define the VC dimension of $\mathscr{H}$ on $\mathscr{G}' \subseteq \mathscr{G}$ as

$$\mathsf{VC}_{\mathscr{G}'}(\mathscr{H}) := \max\{s \mid \exists G_1, \ldots, G_s \text{ in } \mathscr{G}' \text{ which can be shattered by } \mathscr{H}\}$$

Theorem (Morris et al. 2023)

$$\mathsf{VC}_{\mathscr{G}'}(\mathscr{H}) \leq |\mathscr{G}'/_{\rho(\mathscr{H})}|$$

Equivalence classes induced by $\rho(\mathscr{H})$

Morris, G.,Tönshoff, Grohe; *WL meet VC* (2023).

Expressive power

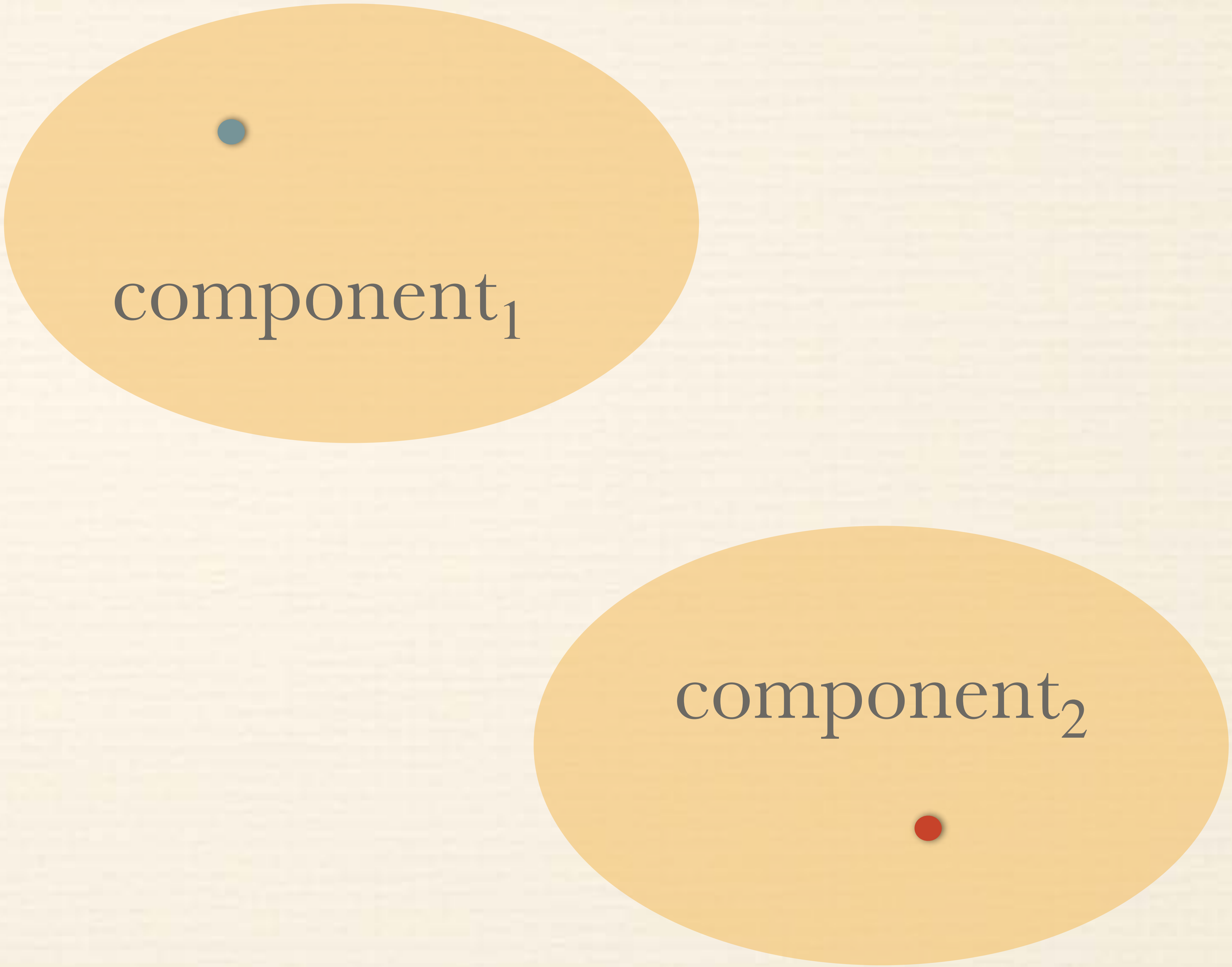
- Which inputs can be separated/distinguished by embeddings in $\mathscr{H}$.
- Which embeddings can be approximated by embeddings in $\mathscr{H}$?
- What is the VC dimension of $\mathscr{H}$?
- Which embeddings can be expressed by embeddings in $\mathscr{H}$?

$\mathscr{H}$ = class of embedding methods

Which unary C_2 formulas can MPNNs express?

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

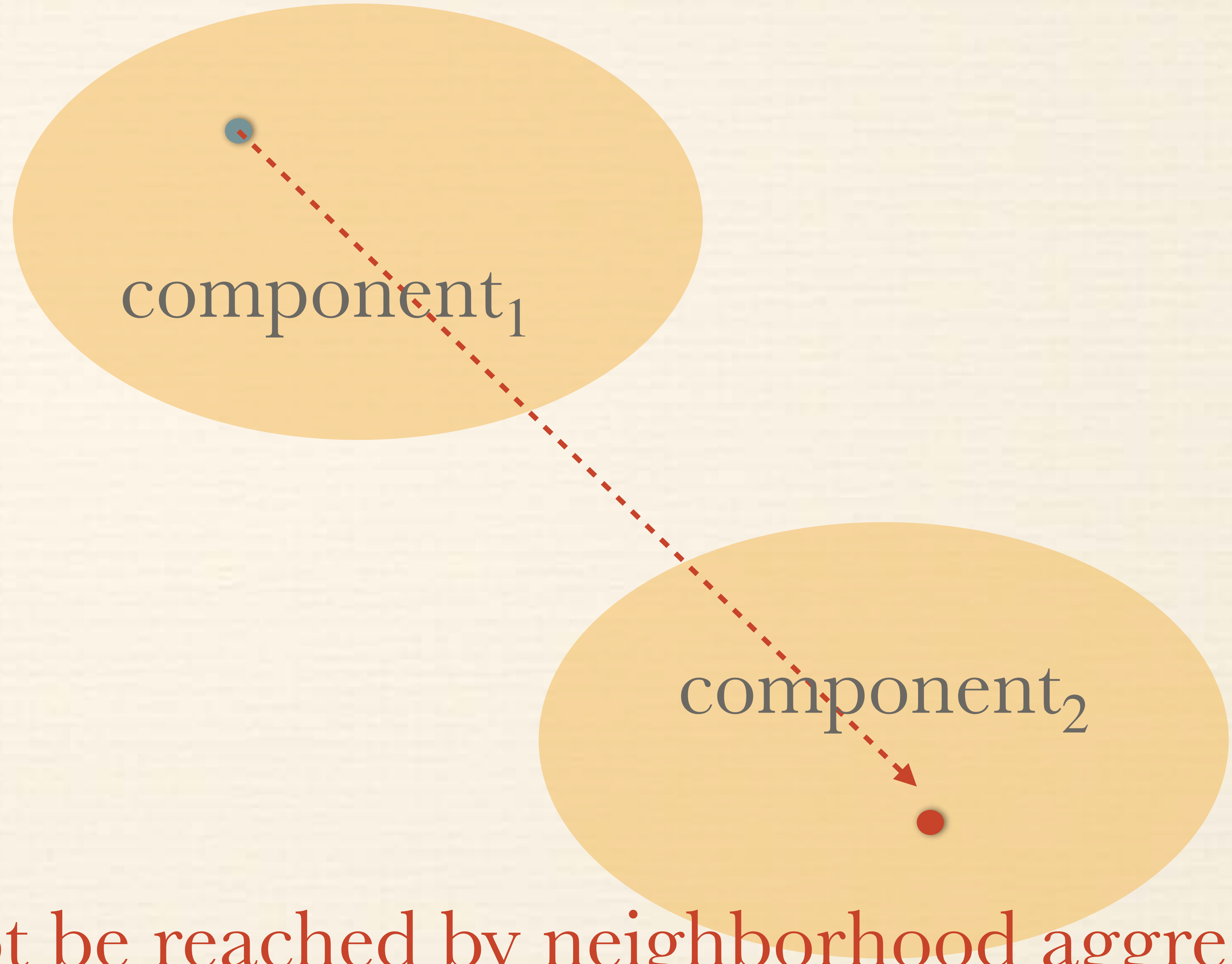
I am blue and there exist
a red vertex somewhere…



Which unary C_2 formulas can MPNNs express?

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

I am blue and there exist
a red vertex somewhere…



Cannot be reached by neighborhood aggregation

Which unary C_2 formulas can MPNNs express?

Theorem (Barceló et al. 2020)

Let $\varphi(x)$ be a unary C_2 formula. Then, $\varphi(x)$ is equivalent to a GC_2 formula *if and only if* $\varphi(x)$ is expressible by the class of MPNNs.

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

Barceló, Kostylev, Monet, Pérez, Reutter, Silva: *The logical expressiveness of graph neural networks* (2020)
Barceló, Kostylev, Monet, Pérez, Reutter, Silva: *The Expressive Power of Graph Neural Networks as a Query Language.* (2020)

MPNNs+

Allow for aggregation over all vertices not only edge-guarded

Theorem (Barceló et al. 2020)

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

Of course, there are queries beyond C_2 which MPNNs can express

Barceló, Kostylev, Monet, Pérez, Reutter, Silva: *The logical expressiveness of graph neural networks* (2020)
Barceló, Kostylev, Monet, Pérez, Reutter, Silva: *The Expressive Power of Graph Neural Networks as a Query Language.* (2020)

Descriptive complexity of GNNs

Theorem (Grohe 2023)

If a unary query Q is computable by a GNN with rational weights and piecewise linear activation functions, then Q is definable in the guarded fragment of $\mathsf{FO}_2 + \mathsf{C}$

Different from C_2
Two sorted logic, numerical predicates etc.

- Extends to general GNNs with real weights and more complex activation functions ⇒ approximate with GNNs as in theorem

M. Grohe. *The Descriptive Complexity of Graph Neural Networks* (2023)

Descriptive complexity of GNNs

Theorem (Grohe 2023)

If a unary query Q is computable by a GNN with rational weights and piecewise linear activation functions, then Q is definable in the guarded fragment of $\mathrm{FO}_2 + \mathsf{C}$

Different from C_2
Two sorted logic, numerical predicates etc.

- Extends to general GNNs with real weights and more complex activation functions $\Rightarrow$ approximate with GNNs as in theorem
- Situates queries expressible by GNNs in (non-uniform) TC^0

Boolean functions computable by non-uniform polynomial-size bounded-depth family of circuits with threshold gates

M. Grohe. *The Descriptive Complexity of Graph Neural Networks* (2023)

Descriptive complexity of GNNs

Theorem (Grohe 2023)

If a unary query Q is computable by a GNN with rational weights and piecewise linear activation functions, then Q is definable in the guarded fragment of $\mathsf{FO}_2 + \mathsf{C}$

Different from C_2
Two sorted logic, numerical predicates etc.

- Extends to general GNNs with real weights and more complex activation functions ⇒ approximate with GNNs as in theorem
- Situates queries expressible by GNNs in (non-uniform) TC^0

Boolean functions computable by non-uniform polynomial-size bounded-depth family of circuits with threshold gates

- Converse holds, with random vertex features.

M. Grohe. *The Descriptive Complexity of Graph Neural Networks* (2023)

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query lang

Distinguishability, approximation, generalisation, uniform and non-uniform expressiveness

2. Analyse expressive power of query language

3. Transfer understanding back to graph learning world

How to compare different classes?

- How to compare such embedding classes theoretically?
- How to bring order to the chaos?

1. See graph embedding methods as queries in some query lang
2. Analyse expressive power of query language
3. Transfer understanding back to graph learning world

GEL

Distinguishability, approximation, generalisation, uniform and non-uniform expressiveness

Conclusion

Todos

- MPNNs: Efficient, most widely used but not expressive (C_2)
- Methods matching C_k for $k > 2$ require tensors making them inefficient
- Ongoing efforts to boost power but preserve efficiency

Todos

- MPNNs: Efficient, most widely used but not expressive (C_2)
- Methods matching C_k for $k > 2$ require tensors making them inefficient
- Ongoing efforts to boost power but preserve efficiency

Feature augmentation

Precompute hom/iso counts

Bouritsas et al.: *Improving graph neural network expressivity via subgraph isomorphism counting* (2020)
Barceló et al.: *Graph neural networks with local graph parameters.* (2021)

Random features

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

Spectral/Global properties

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]

Todos

- MPNNs: Efficient, most widely used but not expressive (C_2)
- Methods matching C_k for $k > 2$ require tensors making them inefficient
- Ongoing efforts to boost power but preserve efficiency

Feature augmentation

Precompute hom/iso counts

Bouritsas et al.: *Improving graph neural network expressivity vi[...]raph isomorphism counting* (2020)
Barceló et al.: *Graph neural networks with local graph [...]*

Random features

Dasoulas et al.: *Coloring gra[...]*
Sato et al.: *Random featur[...]*
Abboud et al. :*The surpr[...]*

Spectral/Glo[...]

Kreuzer et al.: *Rethinking gra[...]*
Ying et al.: *Do transformers really p[...]*
Lim et al.: *Sign and Basis Invariant Netwo[...]*
Zhang et al.: *Rethinking the expressive power of gnns [...]*

Running graph learning method on a derived view. Analysis of expressive power (logic, hom count,…)

Todos

- MPNNs: Efficient, most widely used but not expressive (C_2)
- Methods matching C_k for $k > 2$ require tensors making them inefficient
- Ongoing efforts to boost power but preserve efficiency

Feature augmentation

Precompute hom/iso counts

Bouritsas et al.: *Improving graph neural network expressivity vi[...]raph isomorphism counting* (2020)
Barceló et al.: *Graph neural networks with local graph [...]*

Random features

Dasoulas et al.: *Coloring gra[...]*
Sato et al.: *Random featur[...]*
Abboud et al. :*The surpr[...]*

Spectral/Glo[...]

Kreuzer et al.: *Rethinking gra[...]*
Ying et al.: *Do transformers really p[...]*
Lim et al.: *Sign and Basis Invariant Netwo[...]*
Zhang et al.: *Rethinking the expressive power of gnns [...]*

Running graph learning method on a derived view. Analysis of expressive power (logic, hom count,…)

Subgraph GNNs

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)

Todos

- MPNNs: Efficient, most widely used but not expressive (C_2)
- Methods matching C_k for $k > 2$ require tensors making them inefficient
- Ongoing efforts to boost power but preserve efficiency

Feature augmentation

Precompute hom/iso counts

Bouritsas et al.: *Improving graph neural network expressivity vi… raph isomorphism counting* (2020)
Barceló et al.: *Graph neural networks with local graph …*

Random features

Dasoulas et al.: *Coloring gra…*
Sato et al.: *Random featur…*
Abboud et al. :*The surpr…*

Spectral/Gl…

Kreuzer et al.: *Rethinking gra…*
Ying et al.: *Do transformers really p…*
Lim et al.: *Sign and Basis Invariant Netwo…*
Zhang et al.: *Rethinking the expressive power of gnns via g…*

Running graph learning method on a derived view. Analysis of expressive power (logic, hom count,…)

Subgraph GNNs

Bevilacqua et al: *Equivariant subgraph aggregation network* …22)
Cotta et al.: *Reconstruction for powerful graph representations* …21)
Bevilacqua et al.: *Understanding and extending subgraph G… rethinking their symmetries* (2022)
Huang et al.: *Boosting the cycle counting power of …* (2022)
Papp et al.: *DropGNN: Random drop…*
Qian et al.: *Ordered subgraph …*
You et al.: *Identity-aware …*
Zhang and P. Li. *Nest…*
Zhao et al.: *From sta…*

Running graph learning method on many views, then aggregate. Analysis of expressive power

Todos

- Analysis does not always explain experiments. Is a more fine grained analysis possible, perhaps taking learning process into account?
- Didn't mention graphons (limits of graphs): Expressivity?
- Does connection with GEL (aggregate query language) allow for more transferal of knowledge from database theory/practice to ML?
- If the underlying graph is the result of a query, can one develop a factored graph learning approach?
- Recurrent GNNs are closely related to fixpoint computations. Relationship to query language with recursion?
- Relational embedding methods?

Conclusion

- The query language/logic point of view provides a good abstraction of graph learning methods.
- Leads to interesting insights in capabilities of graph learning methods.
- Great opportunity for database theory and theoretical computer science community contribute …