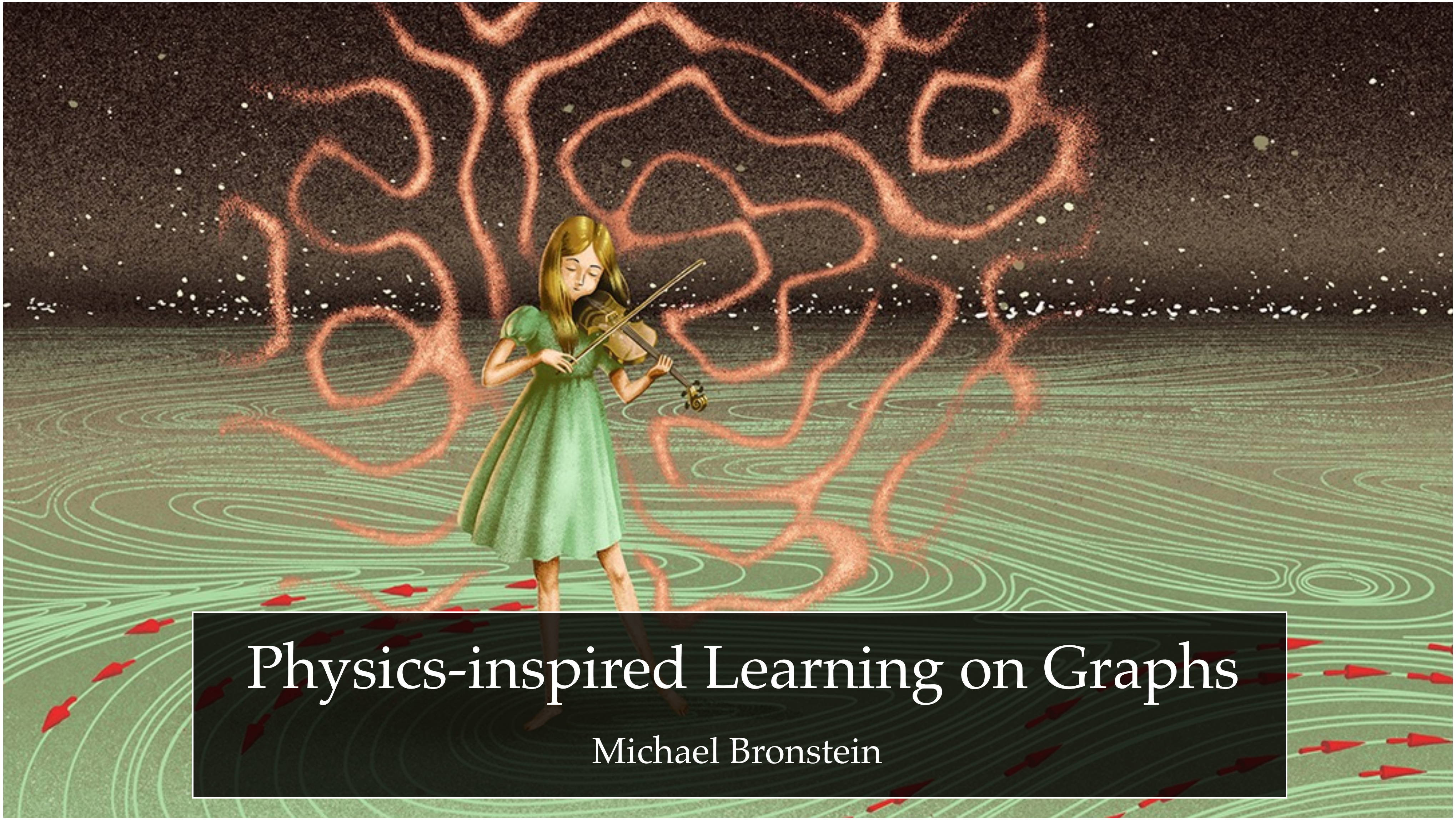


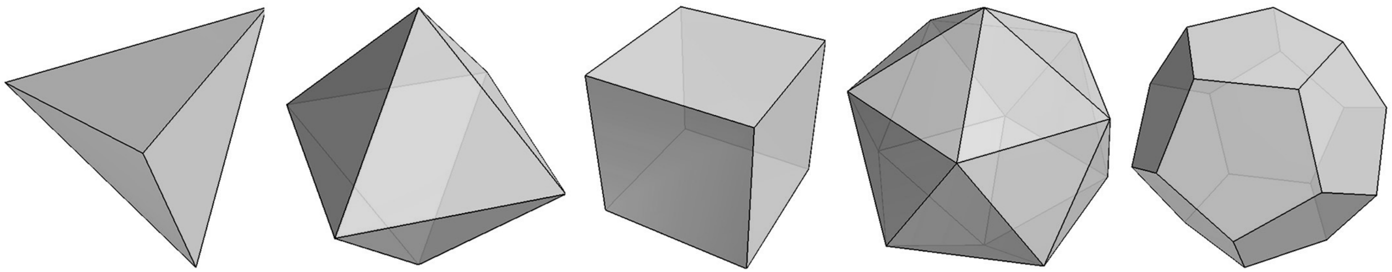
# Physics-inspired Learning on Graphs

Michael Bronstein

"Symmetry, as wide or as narrow as you may define its meaning, is one idea by which man through the ages has tried to comprehend and create order, beauty, and perfection"

**H. Weyl**

Weyl 1952

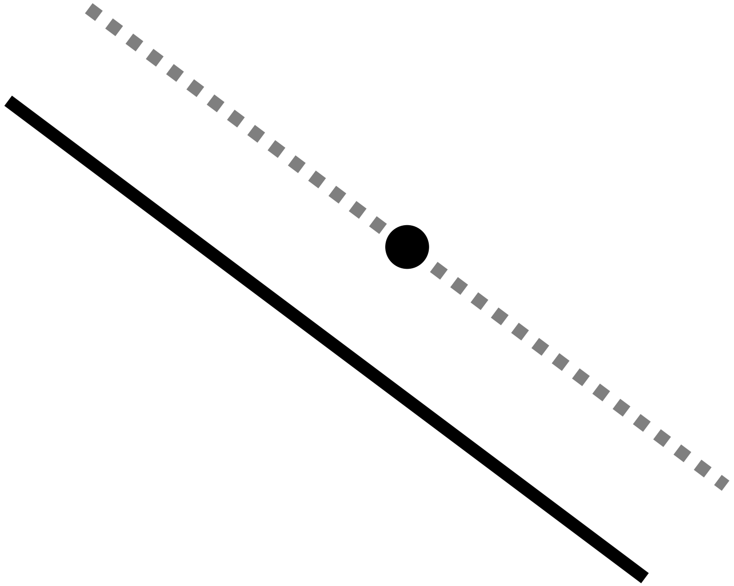


“Platonic solids”

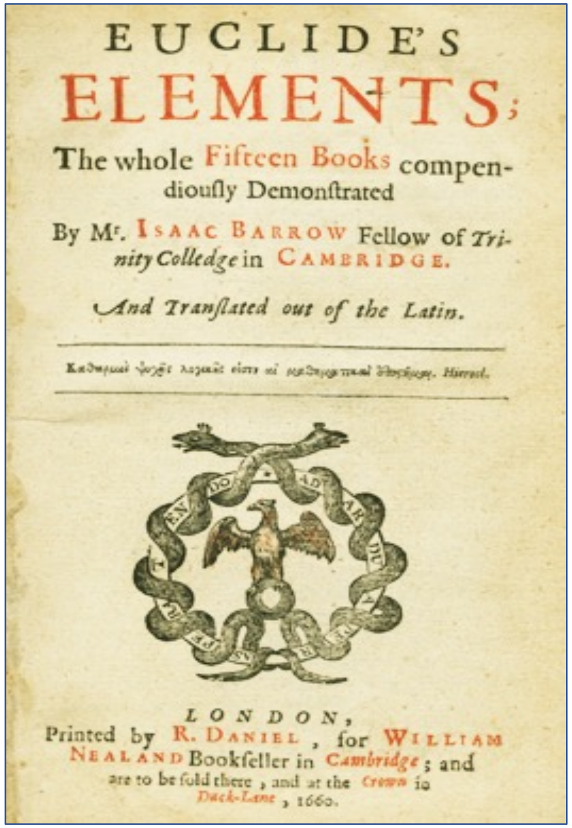
**Plato**

~370 BC

Portrait: Ihor Gorskyi



Fifth Postulate



EUCLIDE'S
ELEMENTS;
The whole Fifteen Books compendiously Demonstrated
By Mr. ISAAC BARROW Fellow of Trinity Colledge in CAMBRIDGE.
And Translated out of the Latin.

LONDON,
Printed by R. DANIEL, for WILLIAM NEALAND Bookseller in Cambridge; and are to be sold there, and at the Crown in Duck-Lane, 1660.

**Euclid**

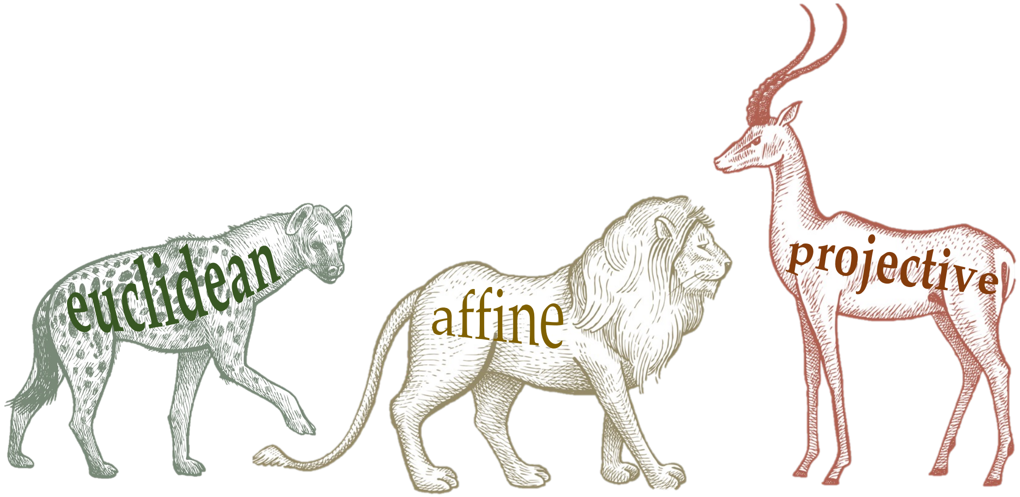
~300 BC

Portrait: Ihor Gorskyi

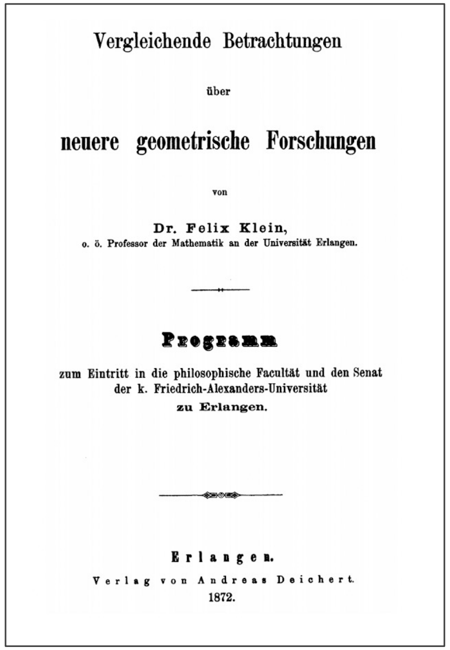
# XIX century


euclidean
affine
projective
ΕΥΚΛΕΙΔΗΣ

# *The Erlangen Programme*




**Geometry = space + transformation group**



Vergleichende Betrachtungen

über

neuere geometrische Forschungen

von

Dr. Felix Klein,

o. ö. Professor der Mathematik an der Universität Erlangen.

Programm

zum Eintritt in die philosophische Facultät und den Senat
der k. Friedrich-Alexanders-Universität
zu Erlangen.

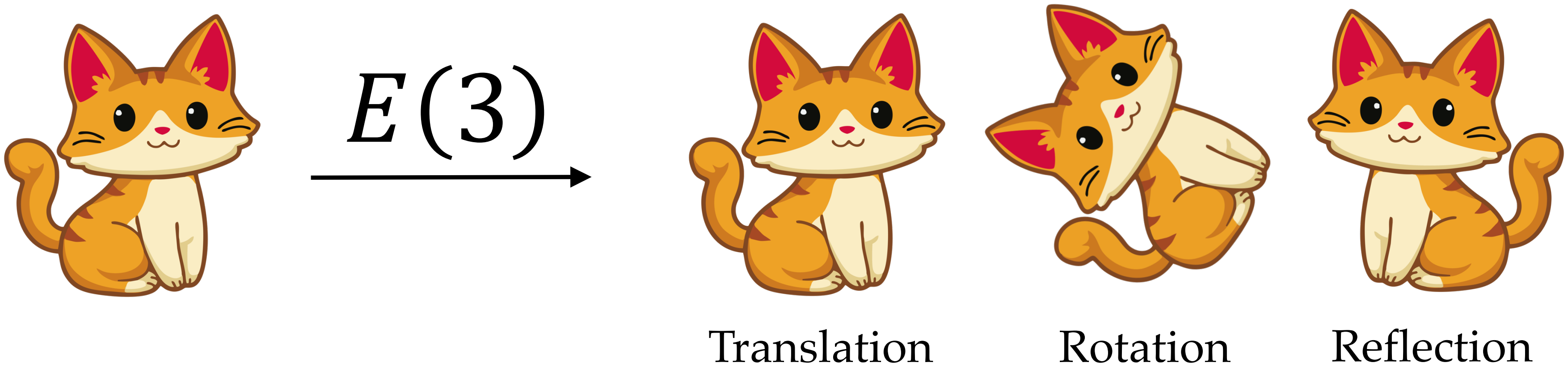
Erlangen.
Verlag von Andreas Deichert.
1872.

**F. Klein**

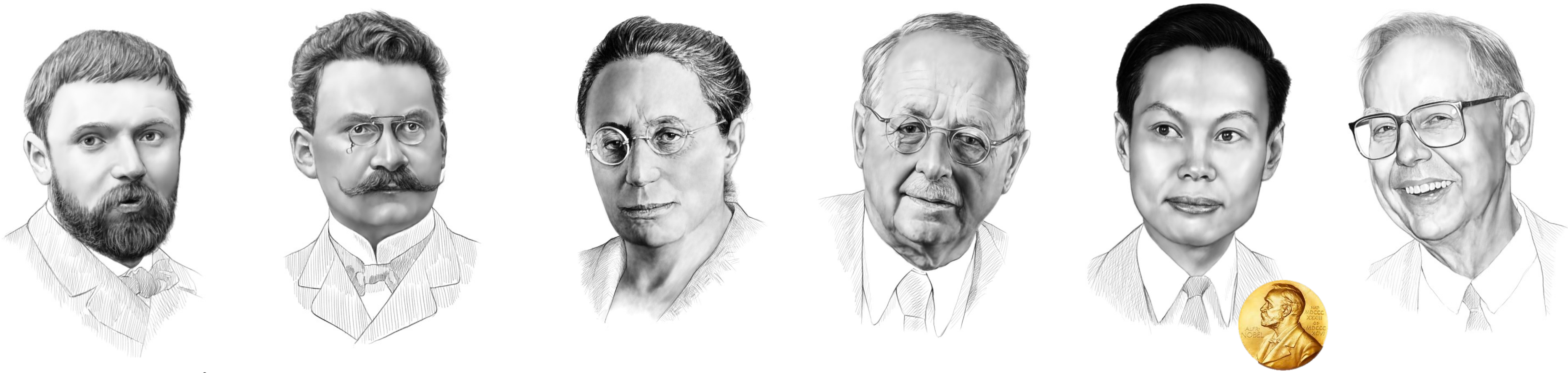
1872

Klein 1872

*Euclidean geometry*



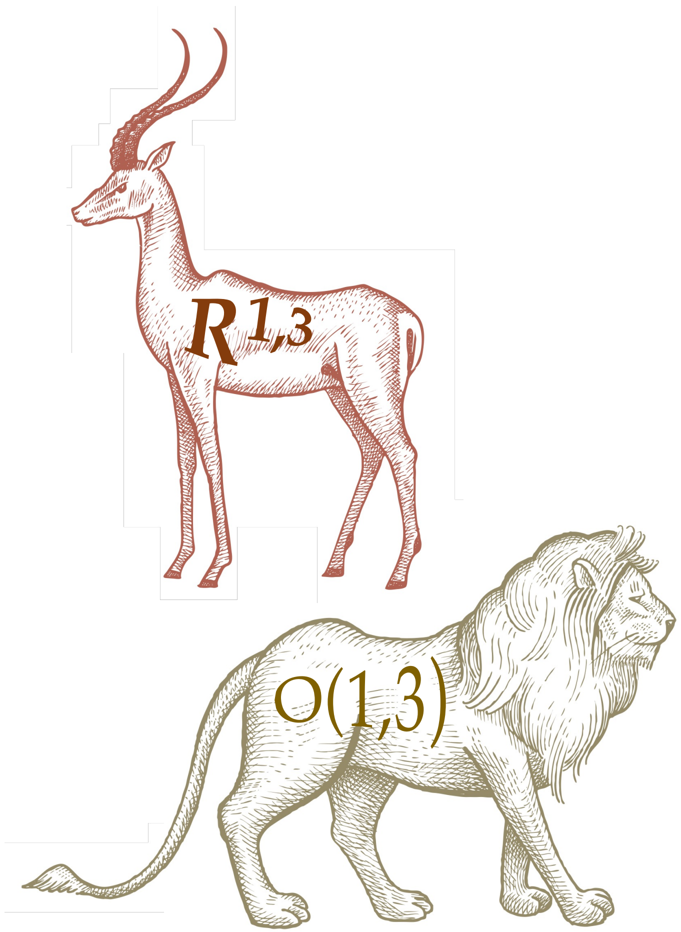

Klein 1872



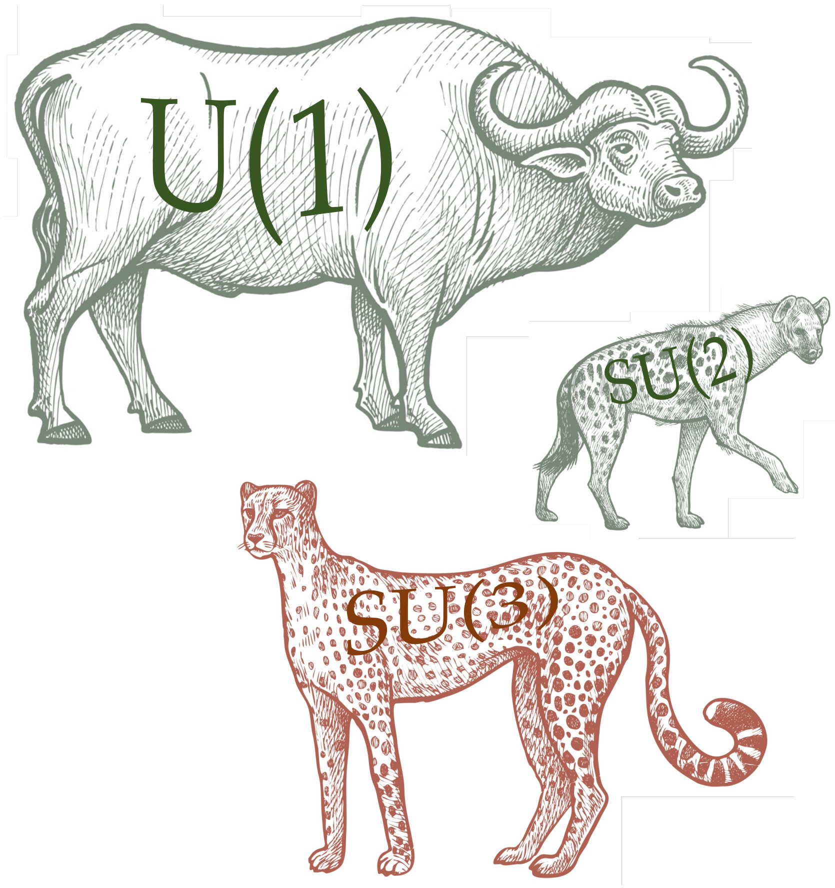
**H. Poincaré** **H. Minkowski** **E. Noether** **H. Weyl** **C. N. Yang** **R. L. Mills**

1904 1907 1918 1929 1954

Poincaré 1904; Noether 1918; Weyl 1929; Yang & Mills 1954; Portraits: Ihor Gorskyi




**External symmetry**



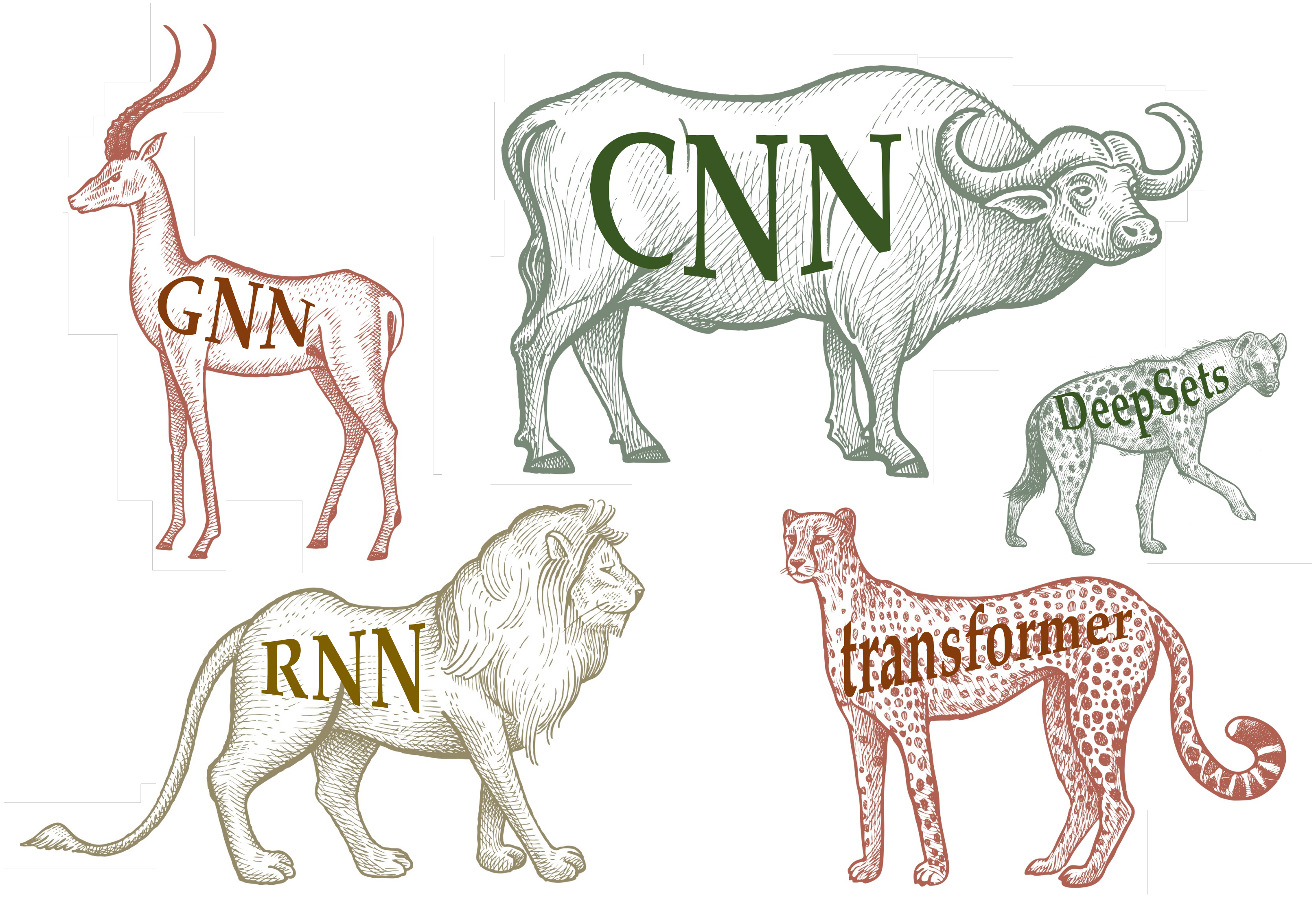

**Internal symmetry**

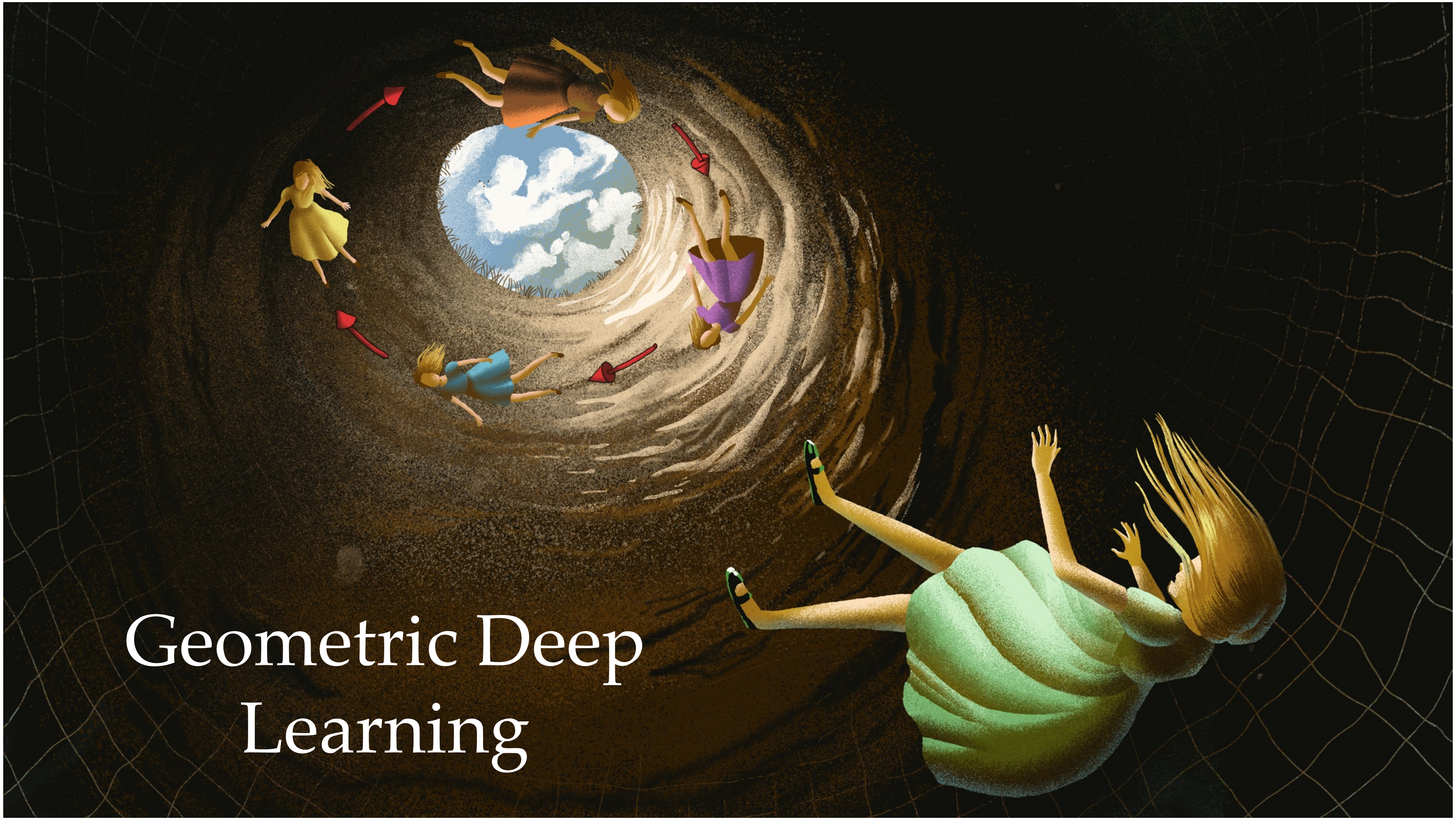
"It is only slightly overstating the case to say that Physics is the study of symmetry"

— *More is different*

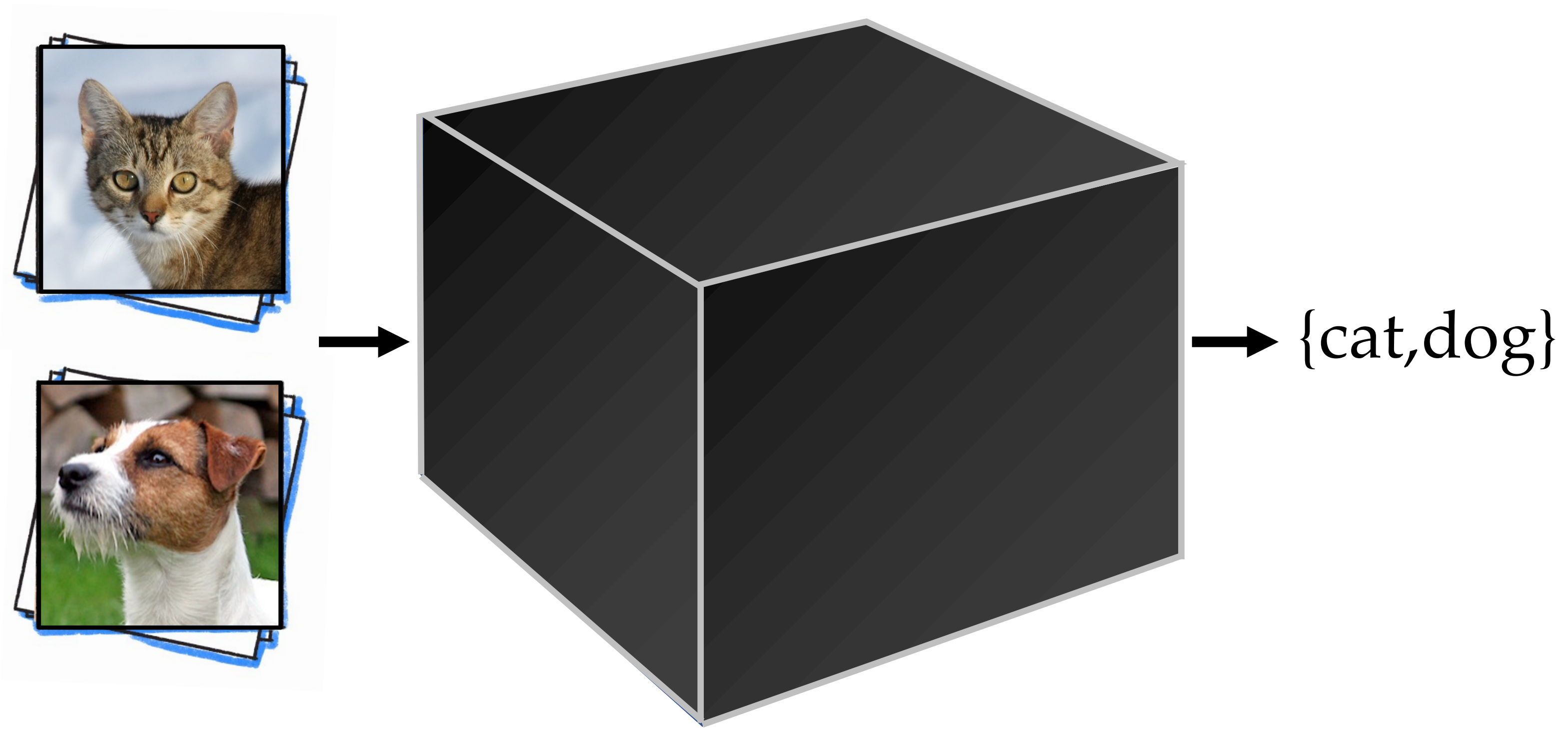
**P. Anderson**

Anderson 1972

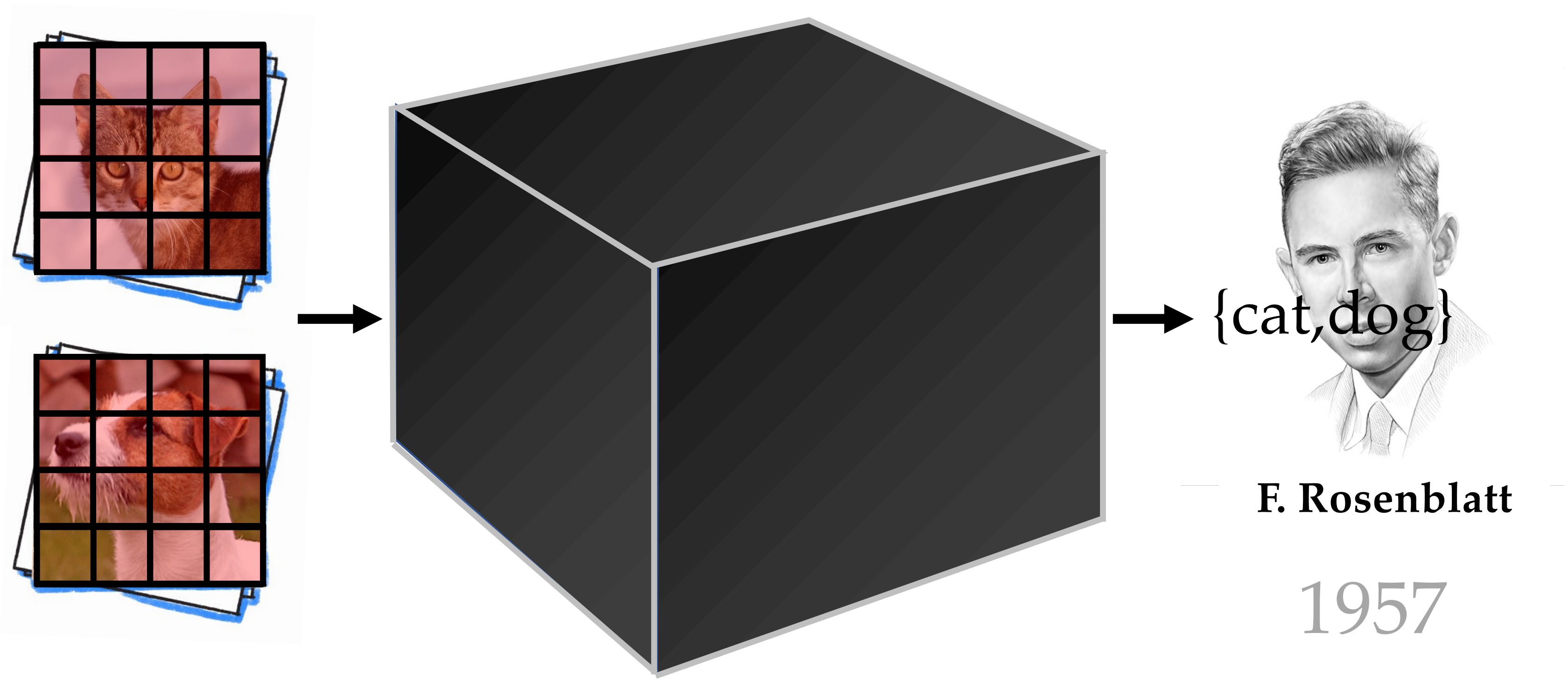

GNN
CNN
DeepSets
RNN
transformer


Geometric Deep
Learning

*Supervised ML = Function Approximation*

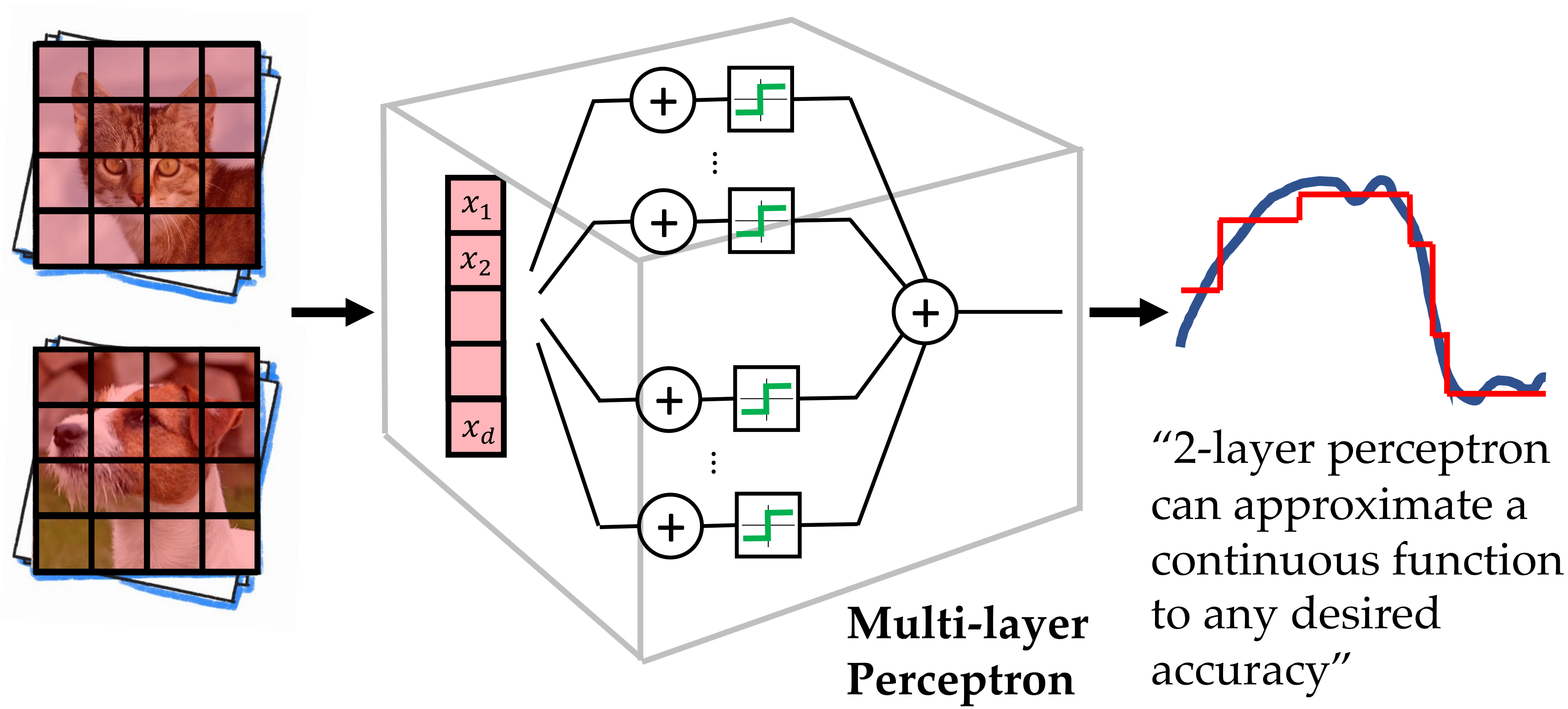


*Supervised ML = Function Approximation*




Rosenblatt 1957; Portrait: Ihor Gorskyi

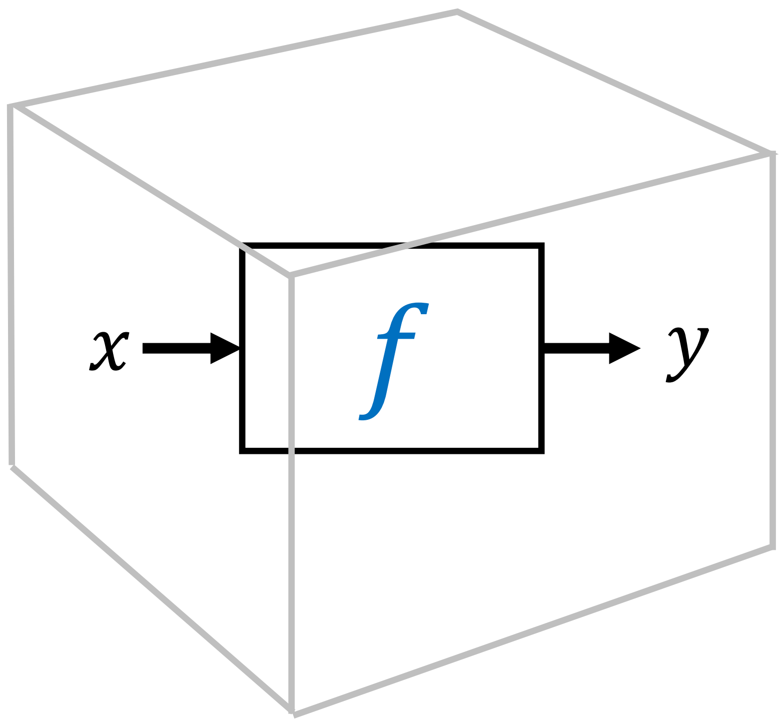
# *Universal Approximation*



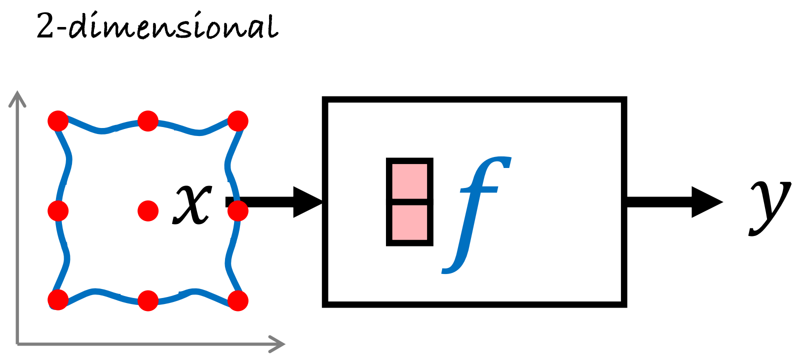

"2-layer perceptron can approximate a continuous function to any desired accuracy"

Universal Approximation: Hilbert's 13th problem 1900; Kolmogorov 1956; Arnold 1957; Cybenko 1989; Hornik 1991; Barron 1993; Leshno et al 1993; Maiorov 1999; Pinkus 1999

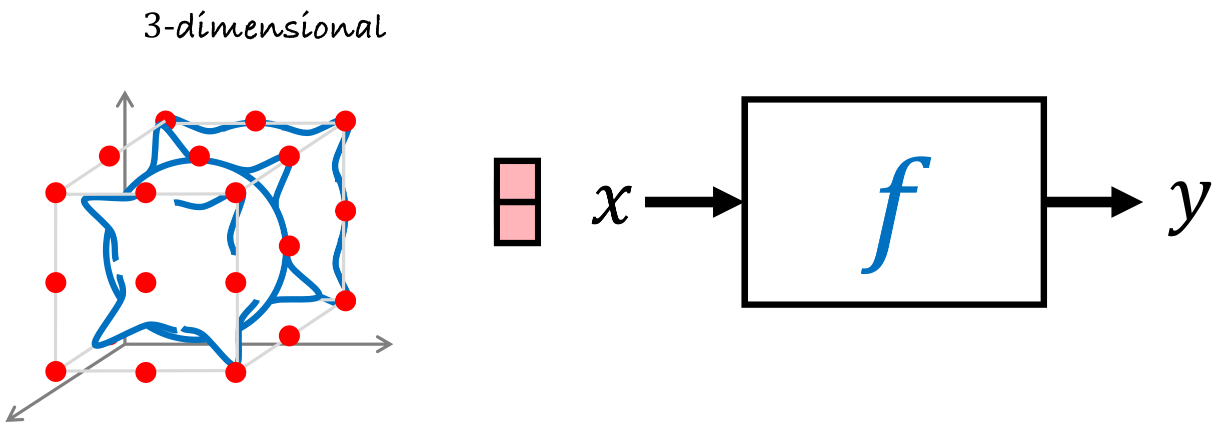
*The Curse of Dimensionality*



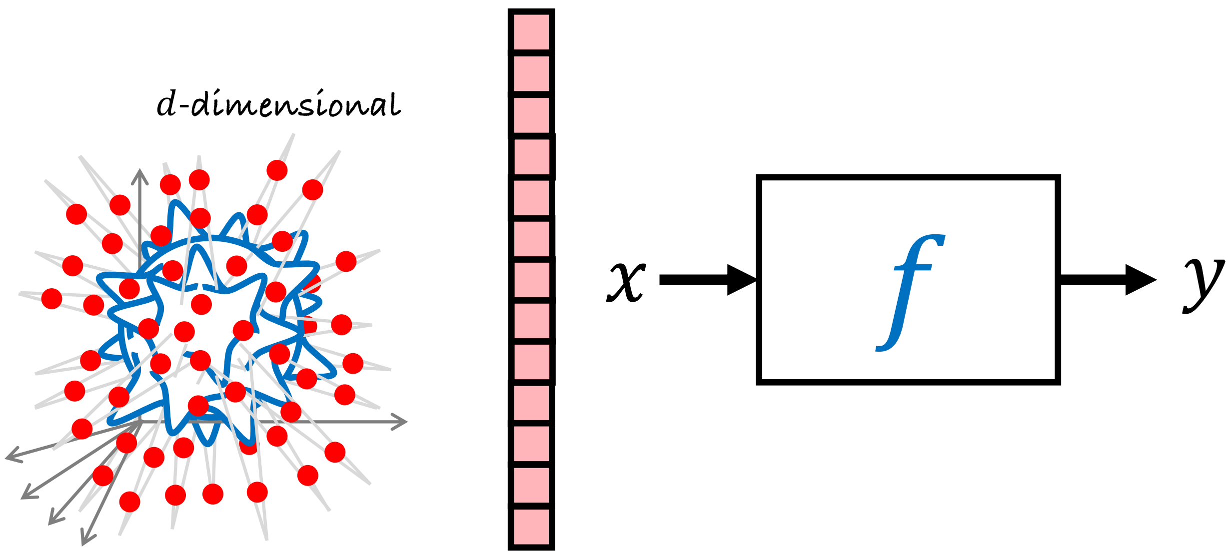
# *The Curse of Dimensionality*



*The Curse of Dimensionality*

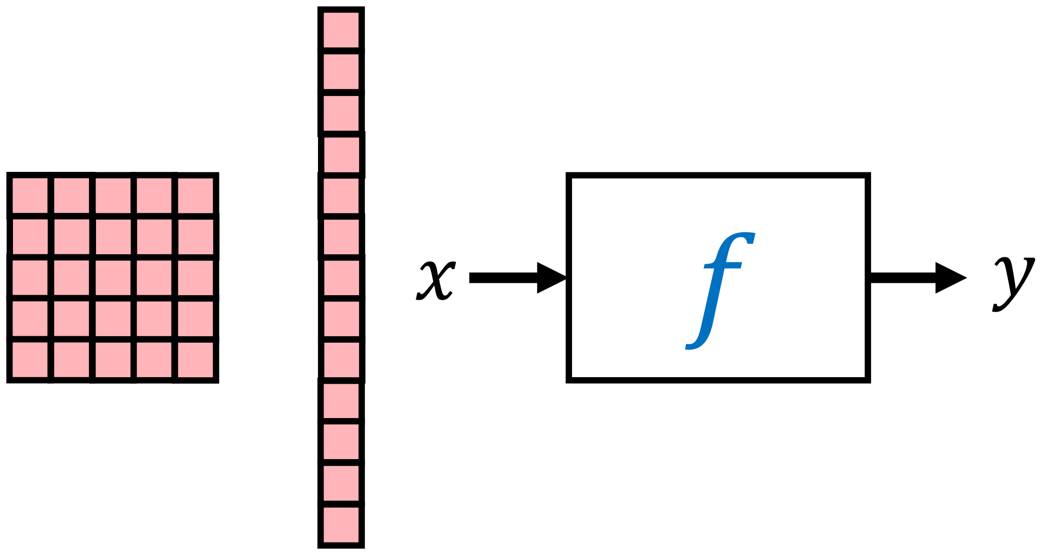


# *The Curse of Dimensionality*

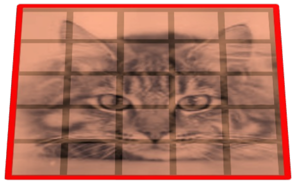


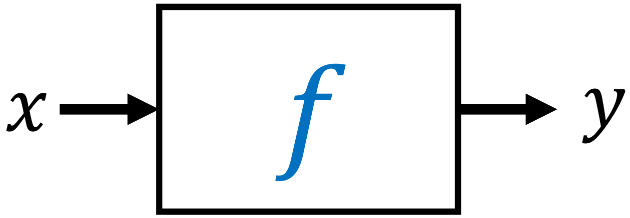

$O(\varepsilon^{-d})$ samples

*Geometric priors*

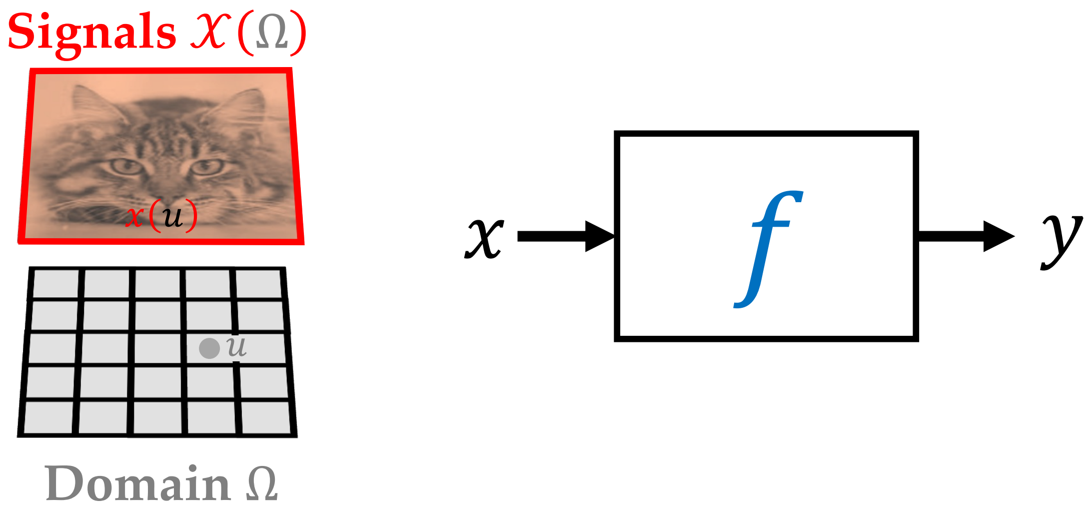


*Geometric priors*

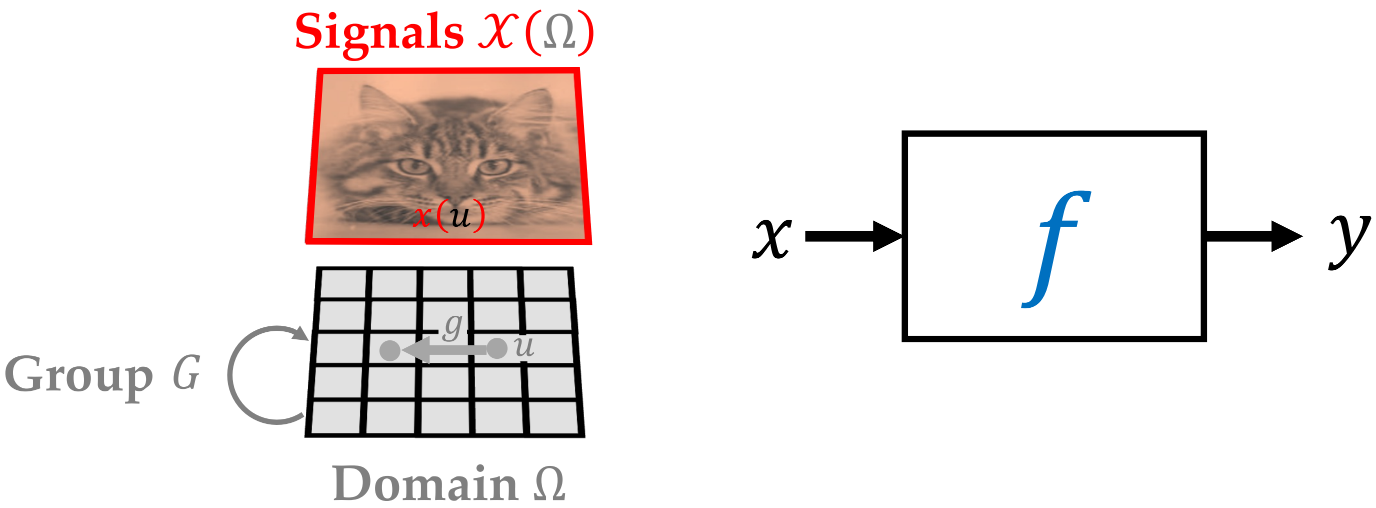




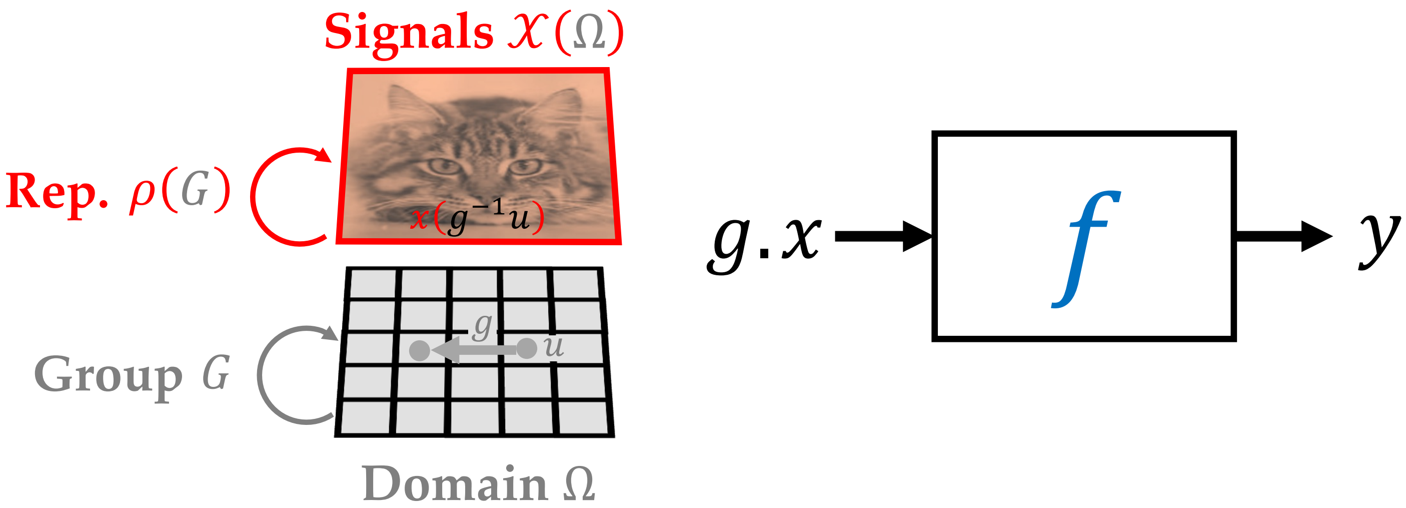
*Geometric priors*



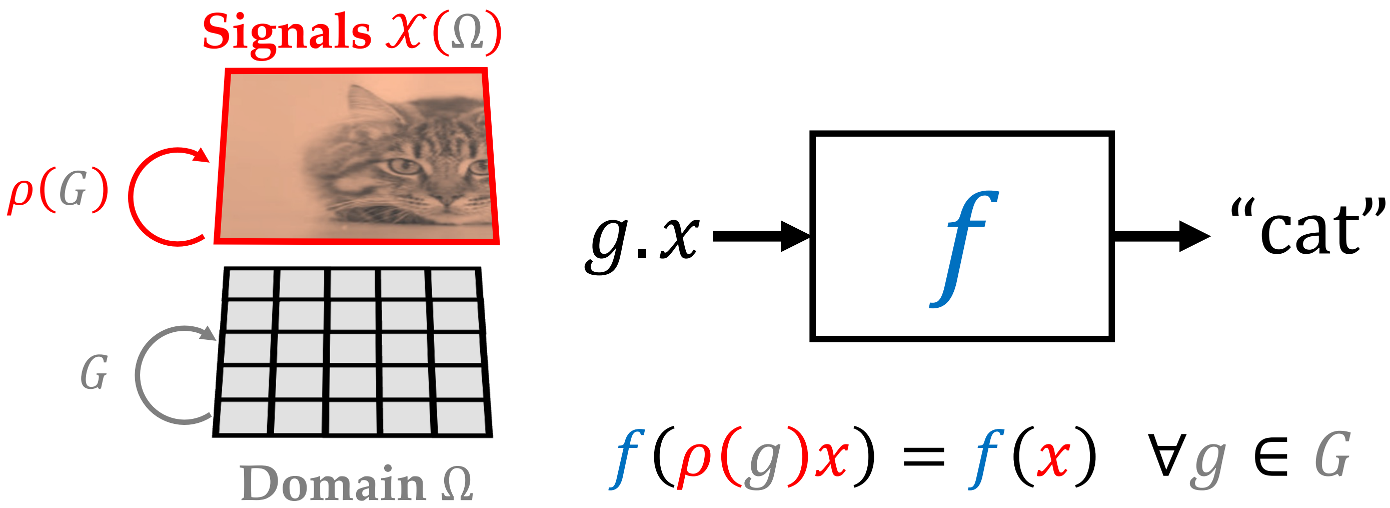
# *Geometric priors*



*Geometric priors*

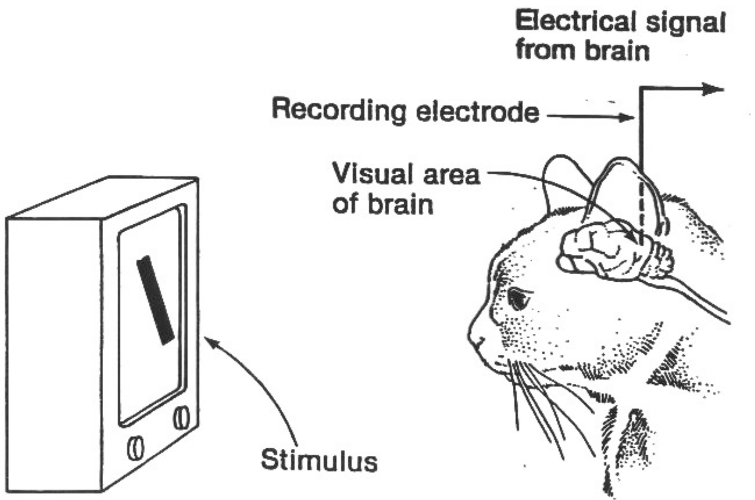


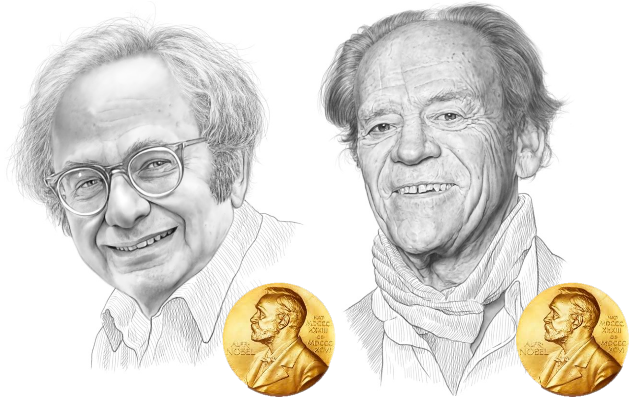
*Geometric priors: Invariance*




$$f(\rho(g)x) = f(x) \quad \forall g \in G$$

# *Early Geometric Architectures*



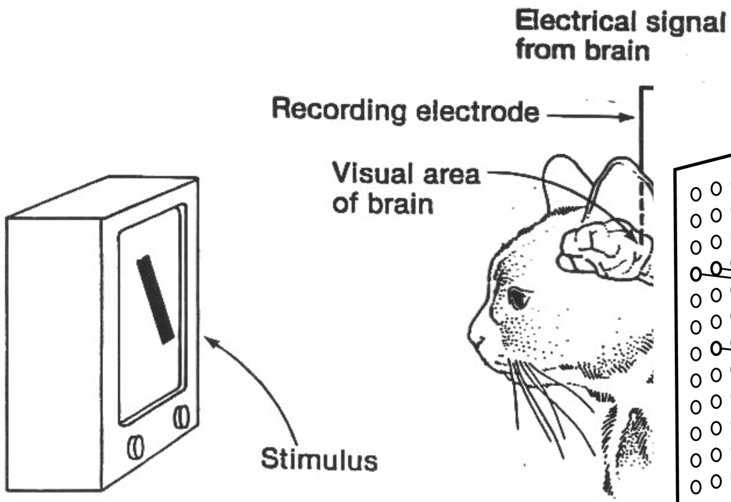



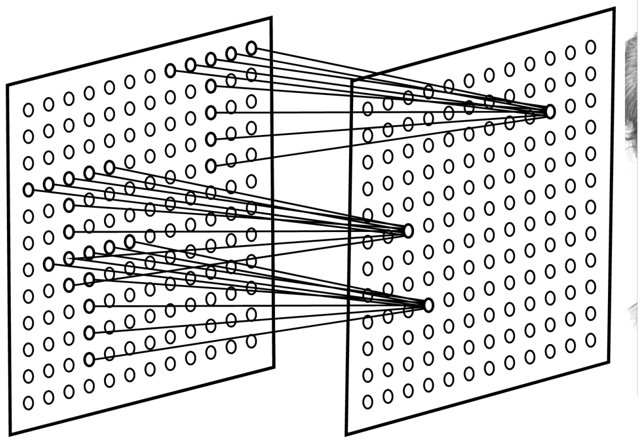
**D. Hubel** **T. Wiesel**

1959

Hubel, Wiesel 1959, 1962; Portraits: Ihor Gorskyi

*Early Geometric Architectures*



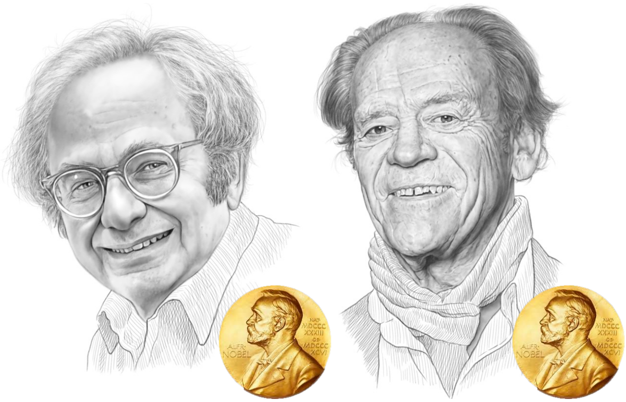



**K. Fukushima**

1959 1980

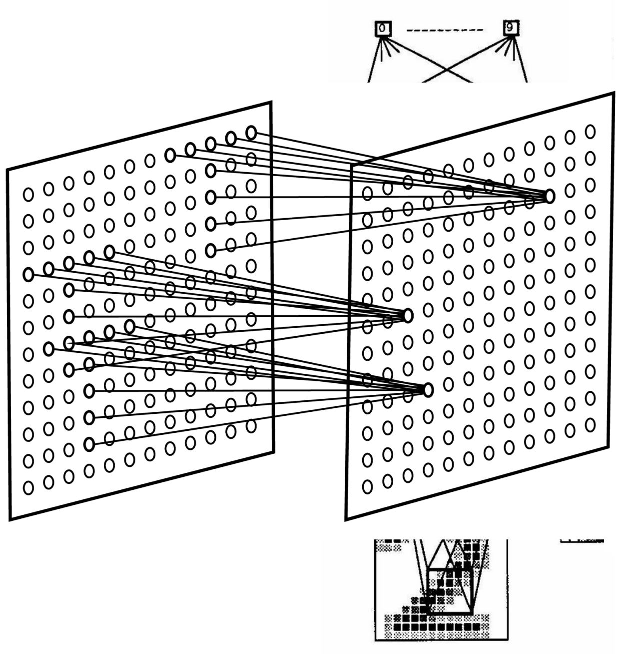
Hubel, Wiesel 1959, 1962; Fukushima 1980; Portraits: Ihor Gorskyi

# *Early Geometric Architectures*



**D. Hubel** **T. Wiesel**

1959

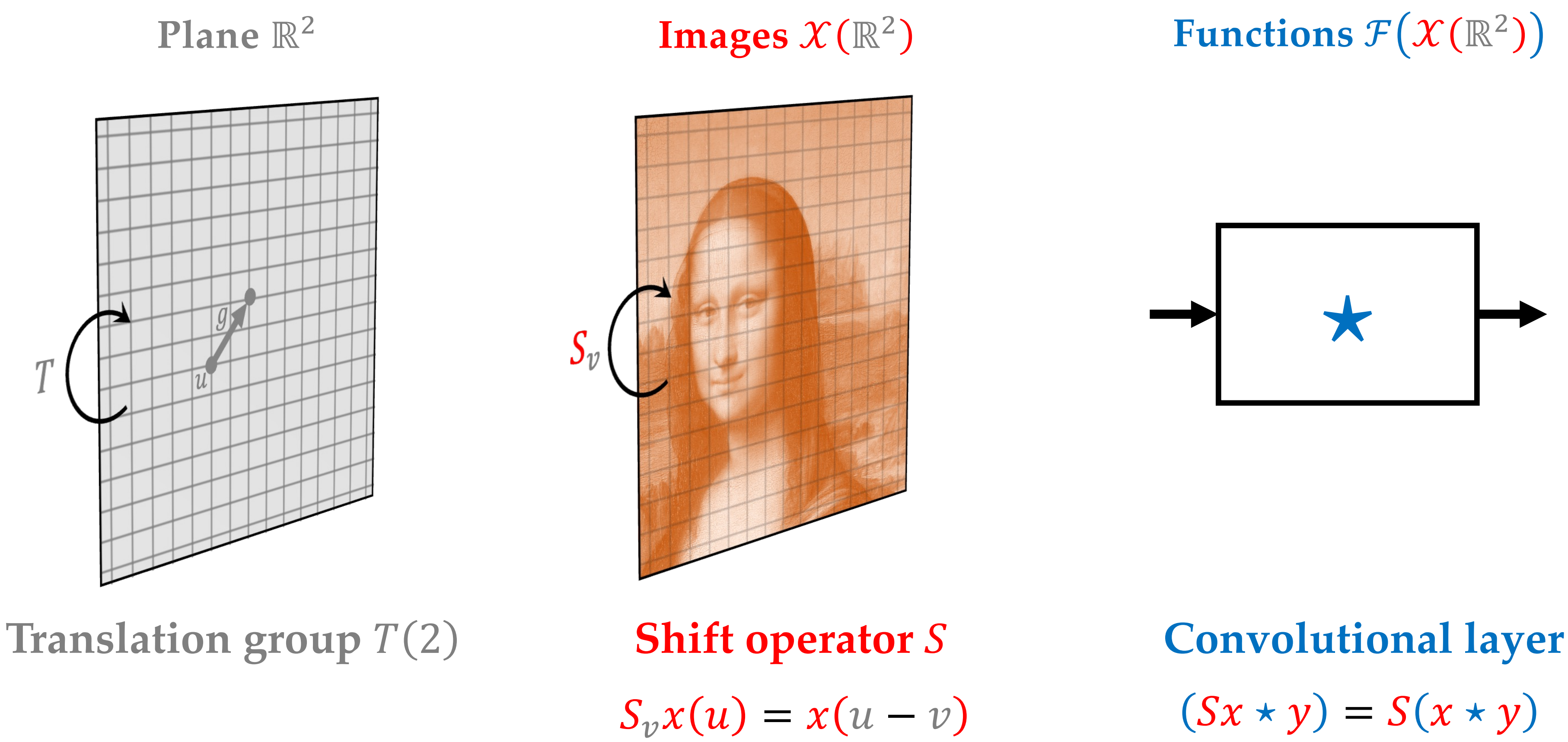


**K. Fukushima**

1980

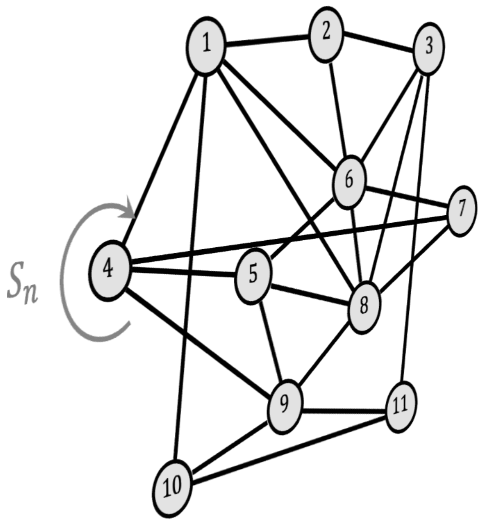
Hubel, Wiesel 1959, 1962; Fukushima 1980; LeCun et al. 1989; Portraits: Ihor Gorskyi

*Convolutional Neural Networks*



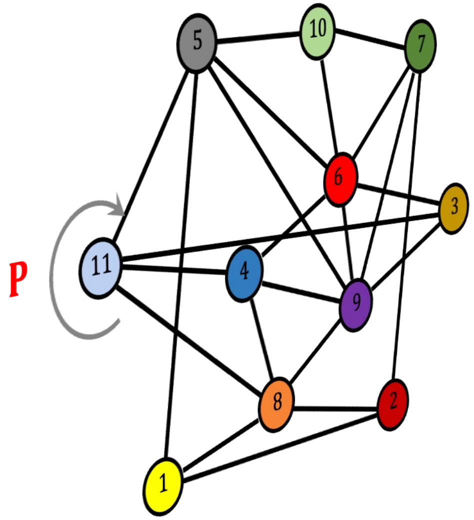
# *Graph Neural Networks*

**Graph** $G = (V, E)$




**Permutation group** $S_n$

**Node features** $\mathcal{X}(G)$




**Permutation matrix P**

$$\mathbf{PX} = \left(x_{\pi^{-1}(i),j}\right)$$

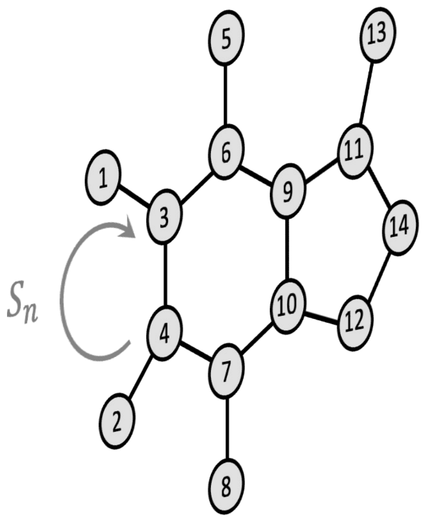
**Functions** $\mathcal{F}(\mathcal{X}(G))$


**Message passing**

$$\mathbf{F}\left(\mathbf{PX}, \mathbf{PAP}^{\top}\right) = \mathbf{PF}(\mathbf{X}, \mathbf{A})$$

# *Geometric ("Equivariant") Graph Neural Networks*

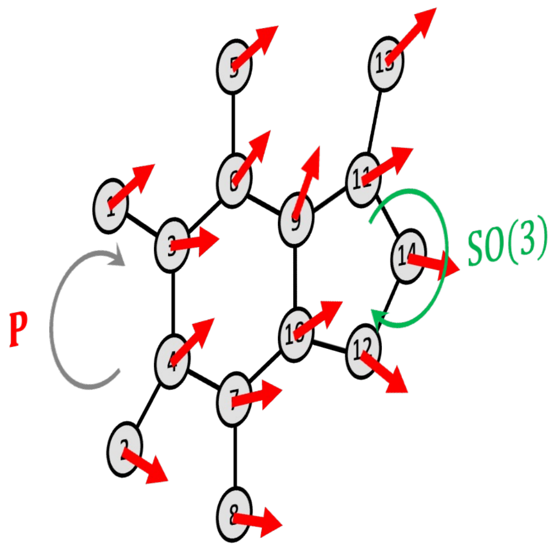
**Geometric Graph $G$**




**Permutation group $S_n$**

*"domain symmetry"*

**Node features $\mathcal{X}(G)$**




**Permutation matrix P**

**Rotation R**

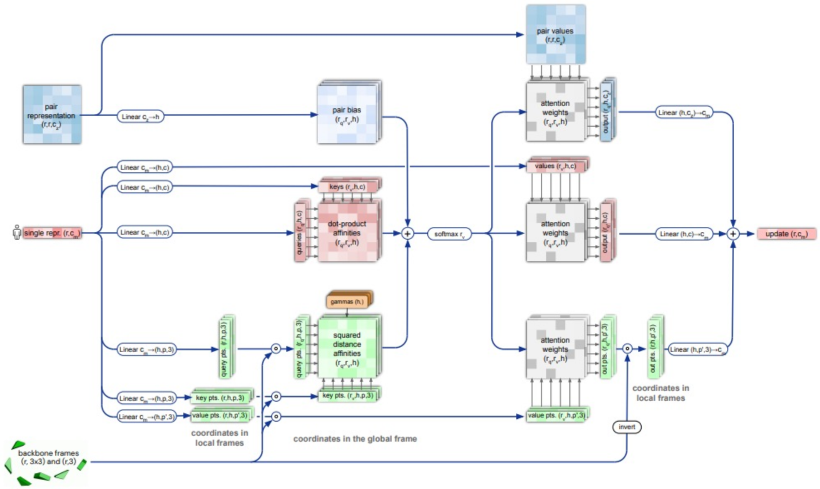
*"data symmetry"*

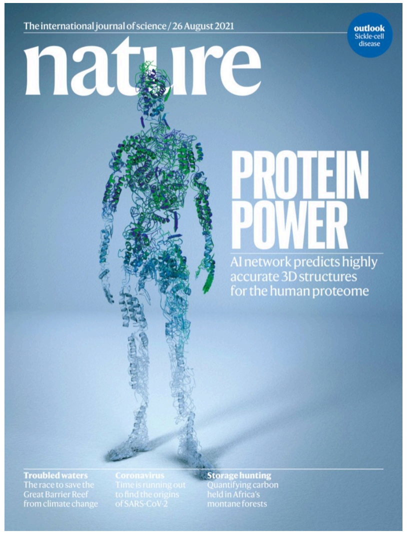
**Functions $\mathcal{F}(\mathcal{X}(G))$**


**Geometric message passing**

$$\mathbf{F}(\mathbf{PXR}, \mathbf{PAP}^\top) = \mathbf{PF}(\mathbf{X}, \mathbf{A})\mathbf{R}$$

# *Revolution in Structural Biology*



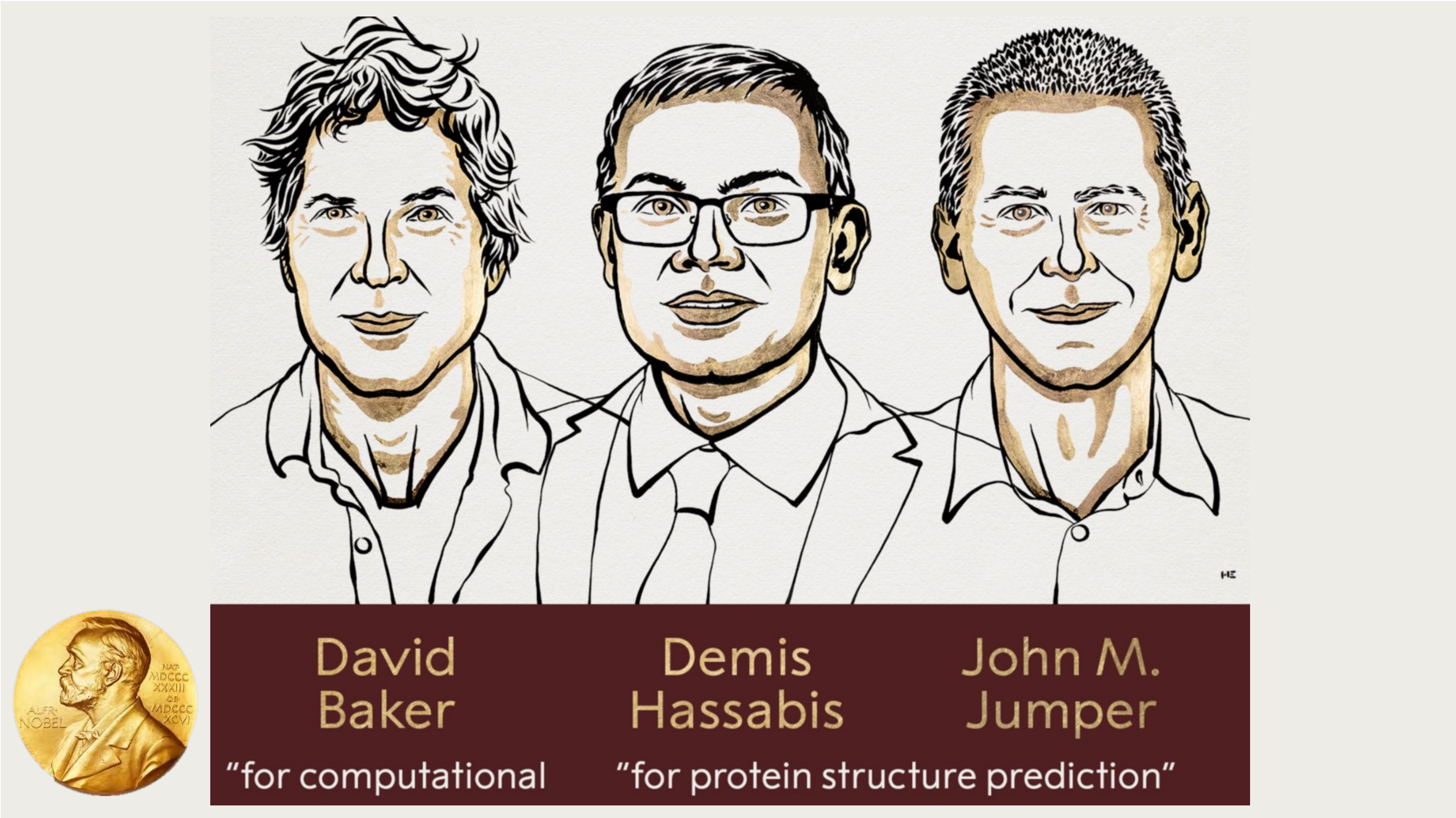



Jumper et al. 2021

**AlphaFold 2**
"Invariant point attention"


Baek et al. 2021

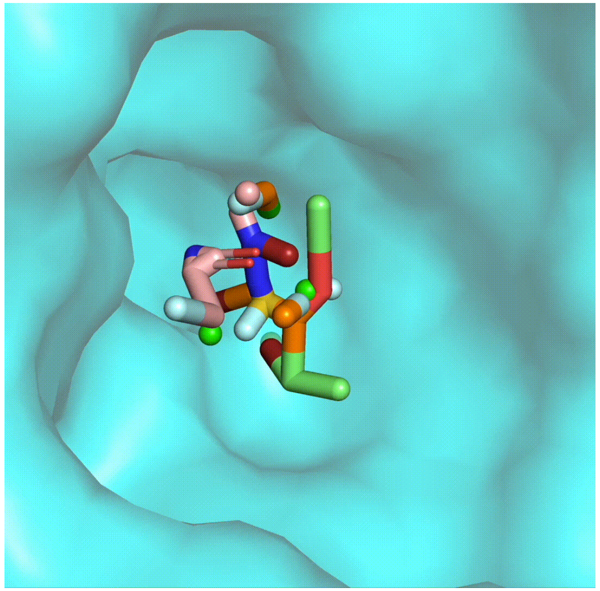
**RosettaFold**
SE(3)-equivariant Transformer


David
Baker
"for computational
Demis
Hassabis
John M.
Jumper
"for protein structure prediction"

*Geometric Generative Models for Chemistry*

"Painting of an astronaut riding a dog on the Moon"

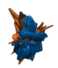
DALL-E 2023 (prompt by B)



"Drug-like molecule binding a protein pocket"

Schneuing et Welling, B, Correia 2022
(Animation: C. Harris)

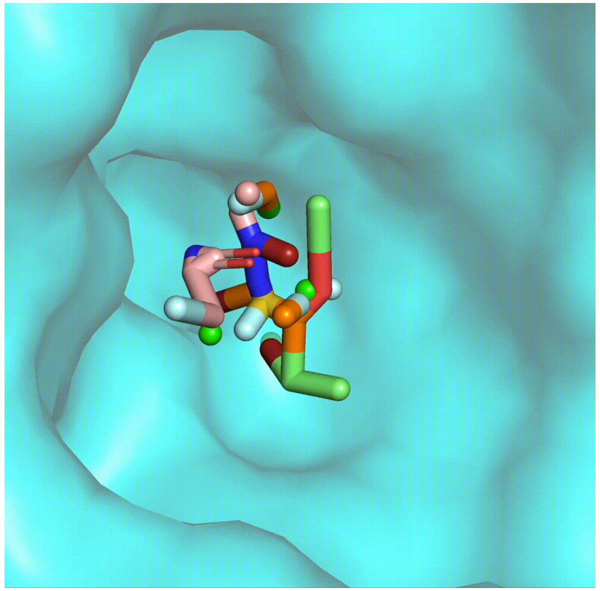
*Geometric Generative Models for Chemistry*



**FoldFlow:** Equivariant flow matching for protein generation

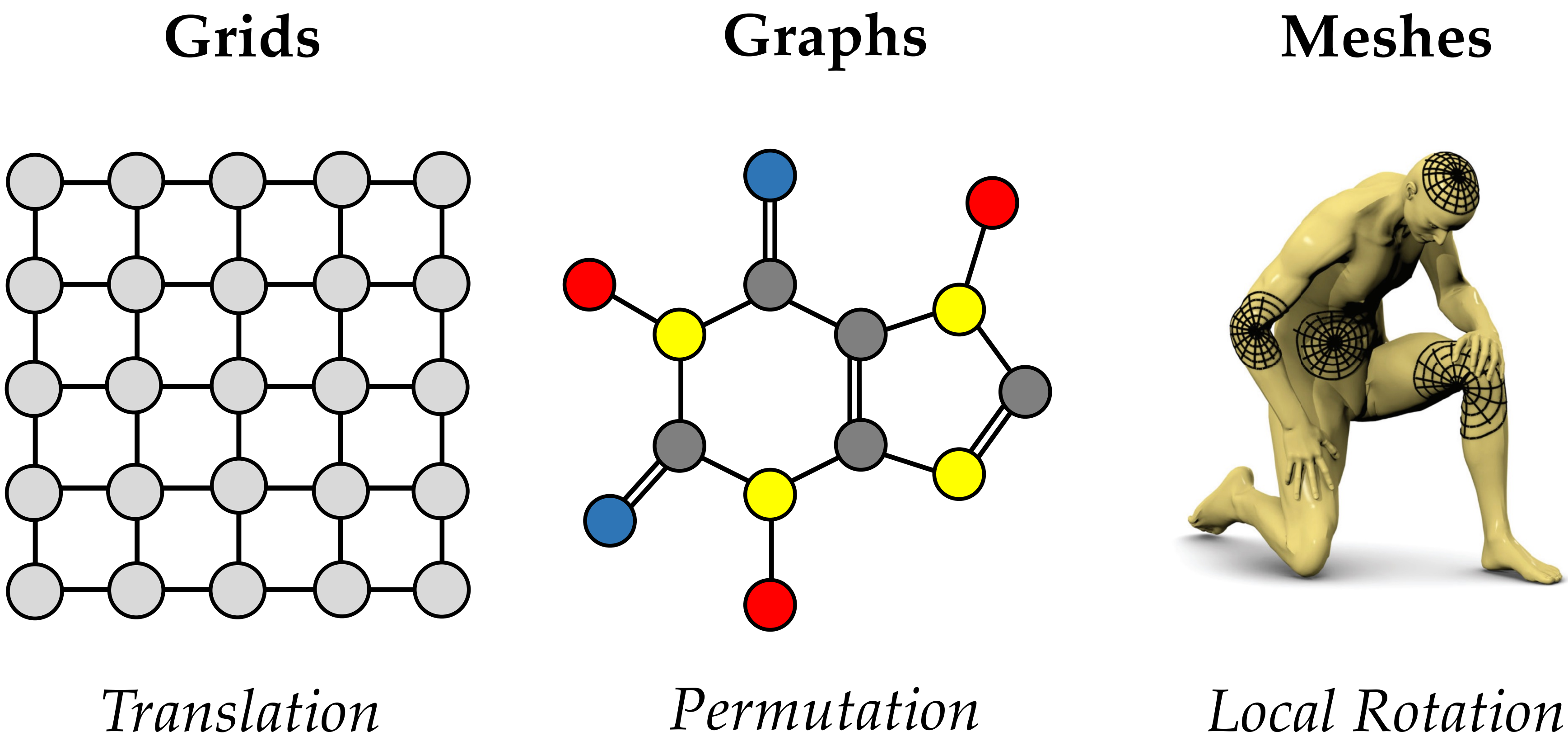
Bose et B, Tong 2024 (FoldFlow)
(Animation: Dreamfold)

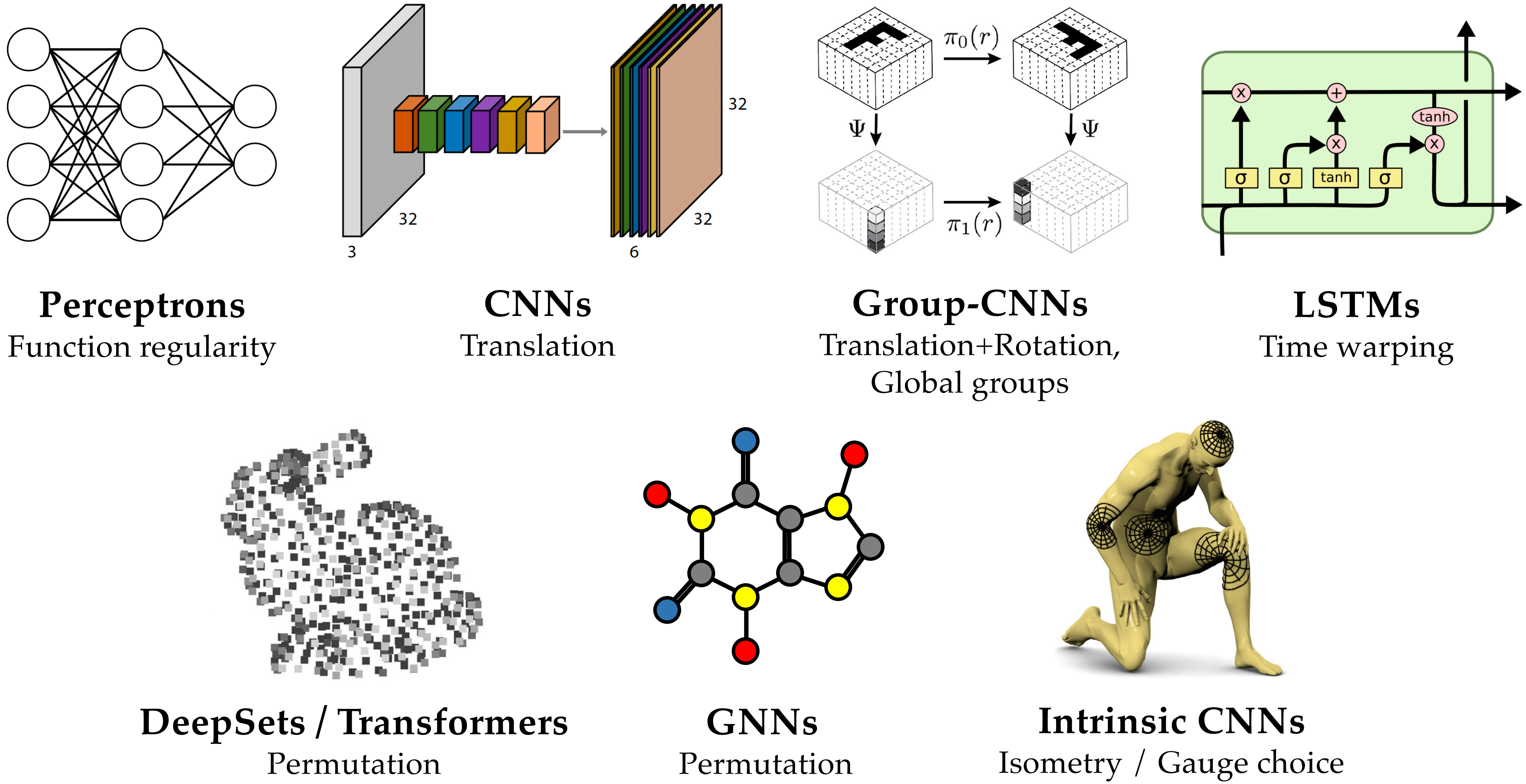
dreamfold



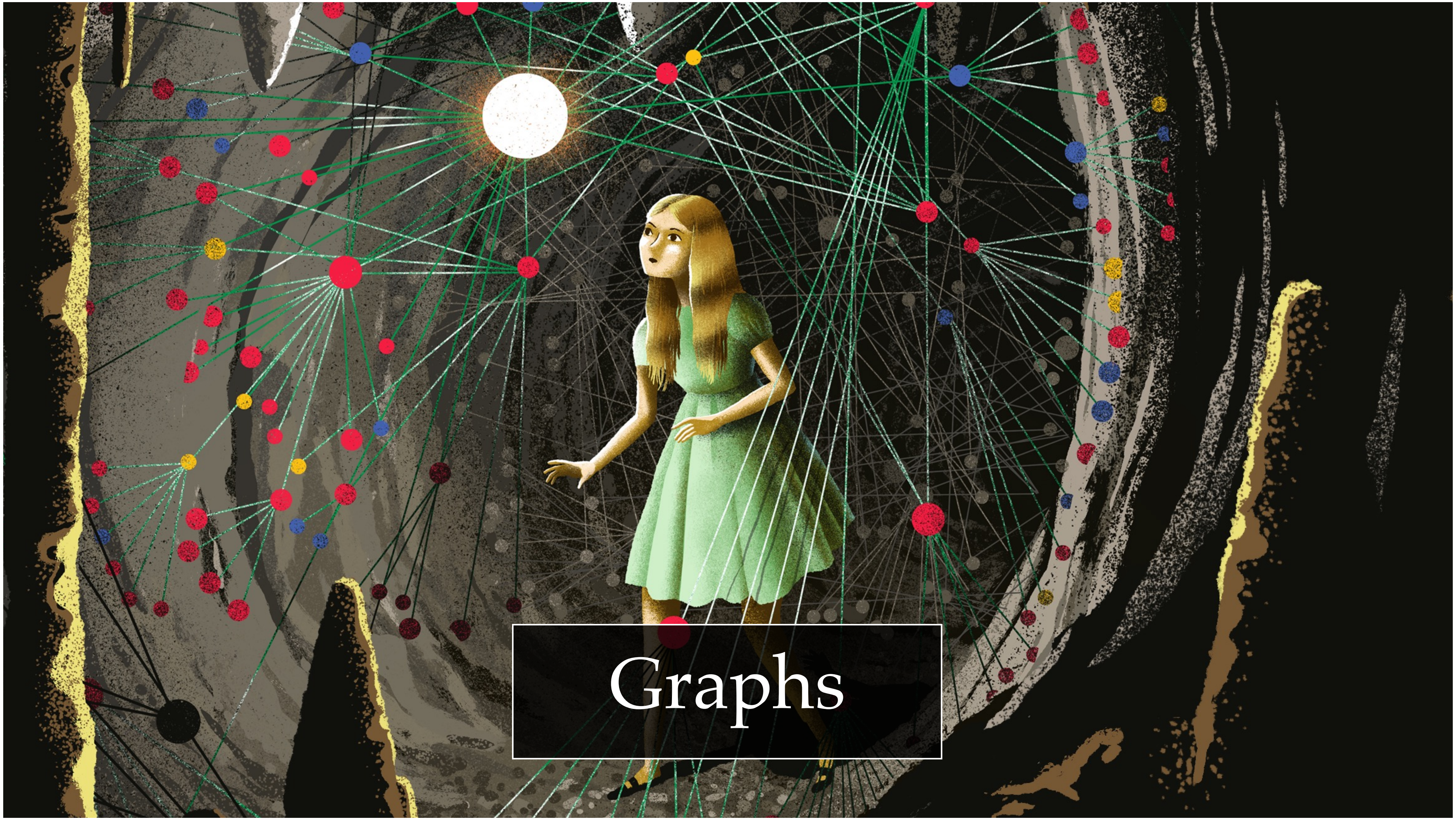
Equivariant diffusion model for constrained molecule generation

Schneuing et Welling, B, Correia 2022
(Animation: C. Harris)

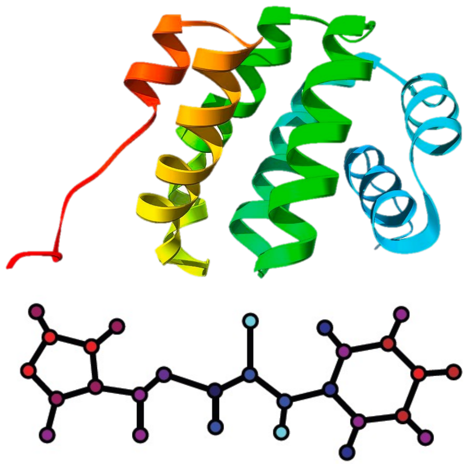

Grids
Graphs
Meshes
Translation
Permutation
Local Rotation


3
32
6
32
32
$\pi_0(r)$
$\Psi$
$\Psi$
$\pi_1(r)$
σ
σ
tanh
σ
tanh
Perceptrons
Function regularity
CNNs
Translation
Group-CNNs
Translation+Rotation,
Global groups
LSTMs
Time warping
DeepSets / Transformers
Permutation
GNNs
Permutation
Intrinsic CNNs
Isometry / Gauge choice

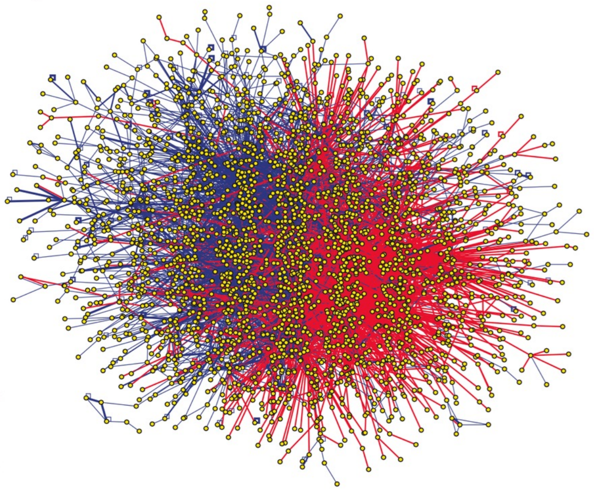

ANY NN
ARCHITECTURE
GEOMETRIC
DEEP LEARNING
TikTok
@tired_actor


Graphs

*Graphs = Systems of Relations and Interactions*



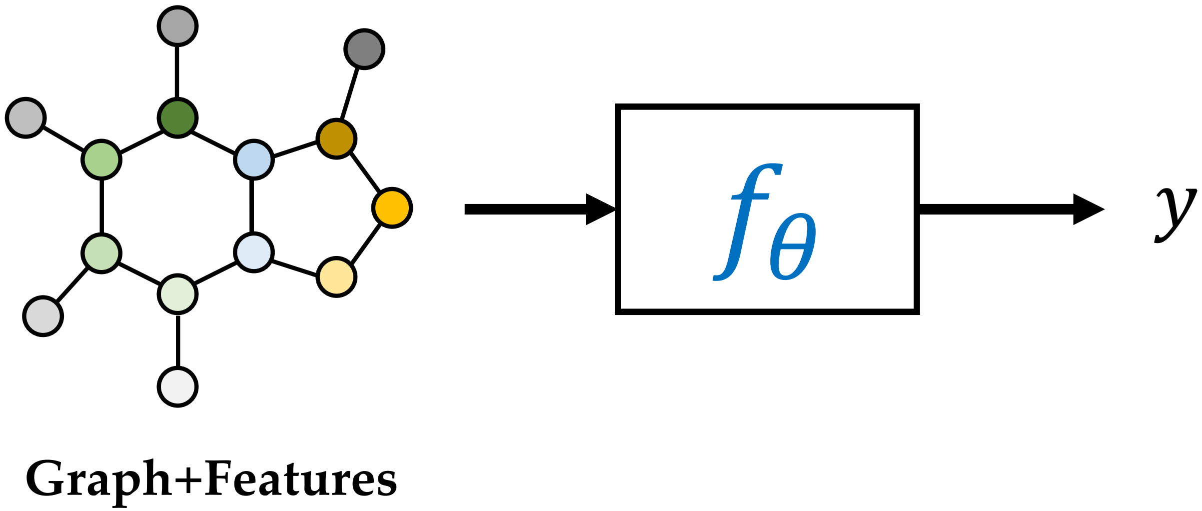
**Molecules**



**Interactomes**

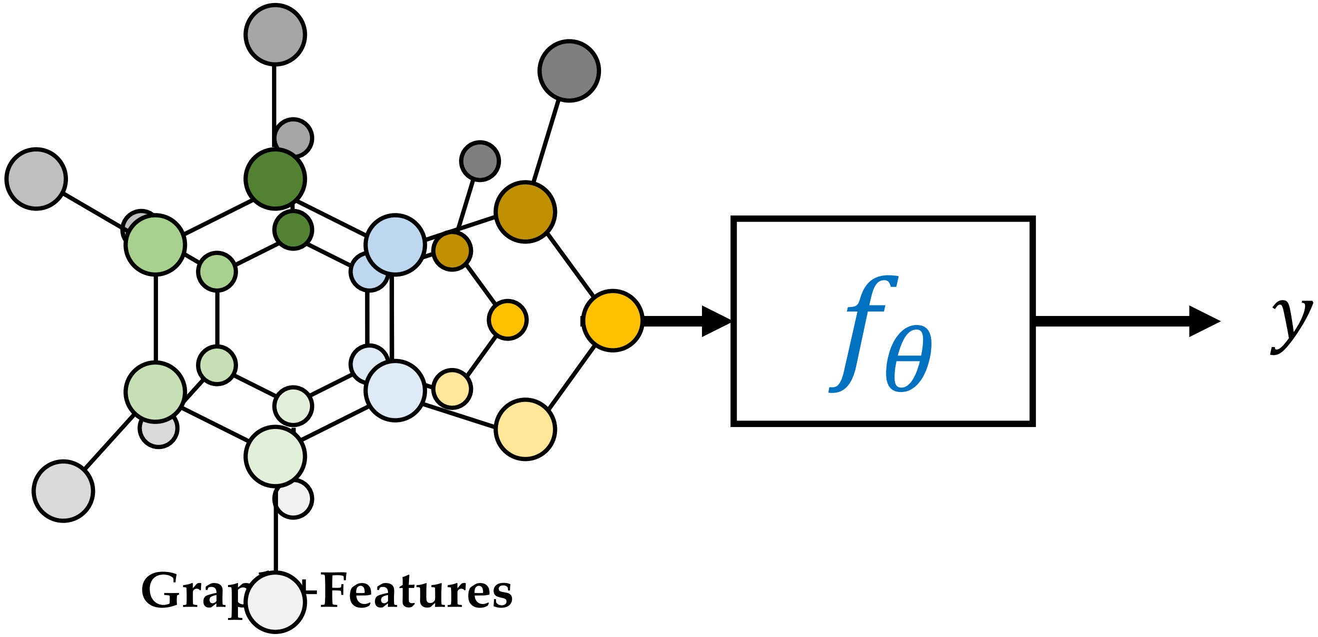
**Social networks**

*GNNs = Parametric graph functions*

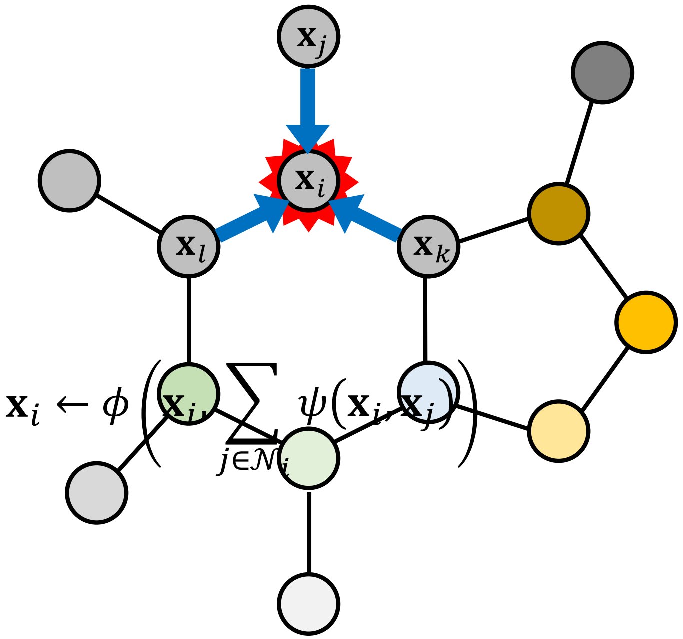



First architecture: Sperduti et al. 1994; Goller, Küchler 1996; Gori et al. 2005; Scarselli et al. 2008 (GNN); Micheli et al. 2009 (NN4G)

*Message Passing Neural Networks*

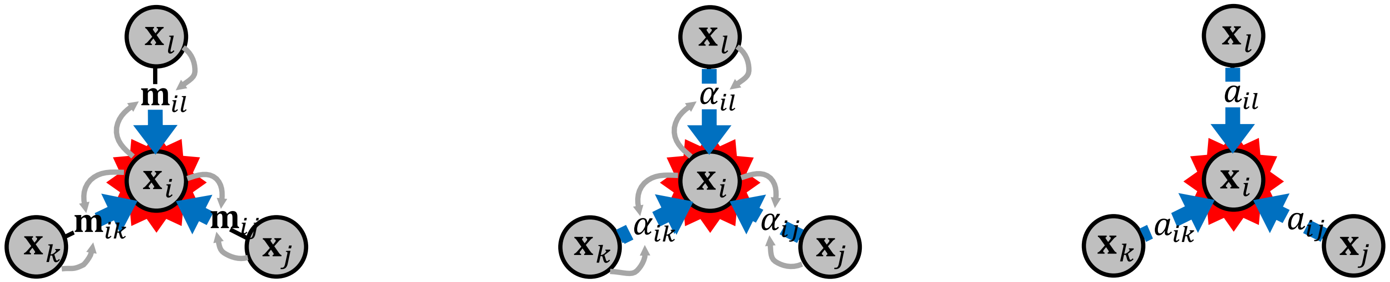


# *Message Passing Neural Networks*




- Every neighbour $j$ sends a **message** $\mathbf{m}_{ij} = \psi(\mathbf{x}_i, \mathbf{x}_j)$ to update $i$
- Messages must be aggregated using a *permutation-invariant* **aggregation operator** (e.g. sum)

# *Message Passing Neural Networks*



$$\phi\left(\mathbf{x}_i, \sum_{j\in\mathcal{N}_i} \psi(\mathbf{x}_i,\mathbf{x}_j)\right) \supset \phi\left(\mathbf{x}_i, \sum_{j\in\mathcal{N}_i} \alpha(\mathbf{x}_i,\mathbf{x}_j)\psi(\mathbf{x}_j)\right) \supset \phi\left(\mathbf{x}_i, \sum_{j\in\mathcal{N}_i} a_{ij}\psi(\mathbf{x}_j)\right)$$

**Generic Message Passing**

Gilmer et al. 2017 (MPNN)
Battaglia et al. 2018 (Graph Networks)
Wang et B, Solomon 2018 (EdgeConv)

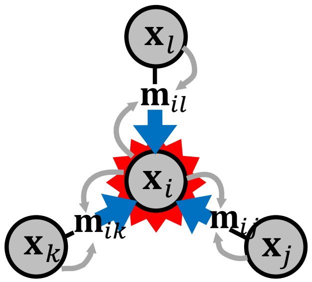
**Attentional**

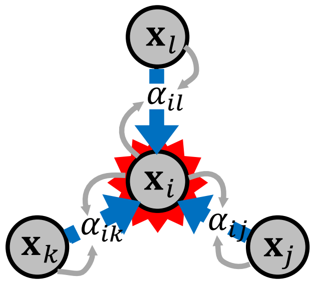
Monti et B 2017 (MoNet)
Veličković et al. 2018 (GAT)

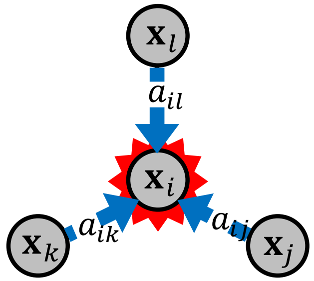
**Convolutional**

Defferard et al. 2016 (ChebNet)
Kipf, Welling 2016 (GCN)
Rossi, Frasca et B 2020 (SIGN)
Ying et al. 2018 (PinSAGE)

# *Message Passing Neural Networks*




$$\boldsymbol{\mathcal{A}}(\mathbf{X}) \quad \supset \quad \mathbf{A}(\mathbf{X})\mathbf{X} \quad \supset \quad \mathbf{AX}$$

**Generic Message Passing**

Gilmer et al. 2017 (MPNN)
Battaglia et al. 2018 (Graph Networks)
Wang et B, Solomon 2018 (EdgeConv)

**Attentional**

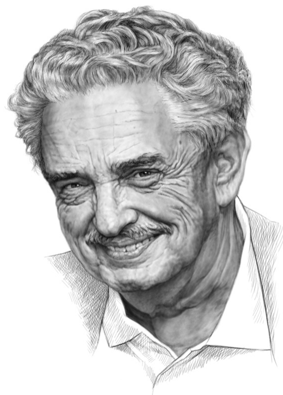
Monti et B 2017 (MoNet)
Veličković et al. 2018 (GAT)

**Convolutional**

Defferard et al. 2016 (ChebNet)
Kipf, Welling 2016 (GCN)
Rossi, Frasca et B 2020 (SIGN)
Ying et al. 2018 (PinSAGE)

*Weisfeiler-Lehman Test*

**Theorem:** (Under some technical conditions) with appropriate choice of aggregation operator and message functions, MPNNs are at most as expressive as the Weisfeiler-Lehman graph isomorphism test.

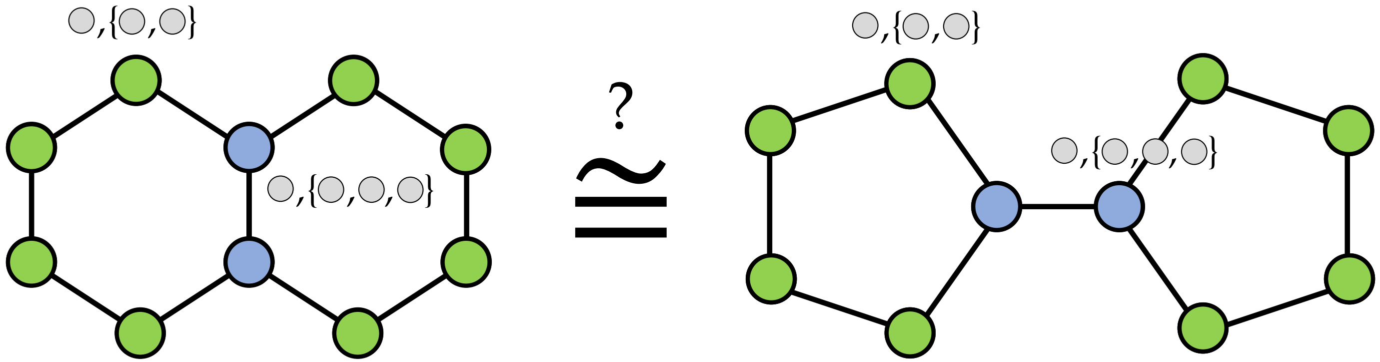


**A. Lehman**

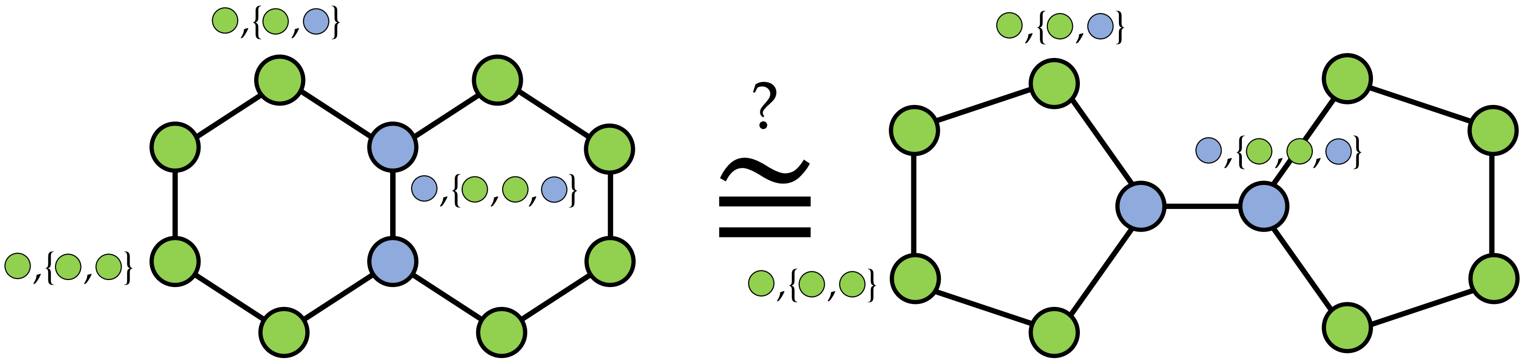
**B. Weisfeiler**

Weisfeiler, Lehman 1968; Xu 2019; Morris 2019

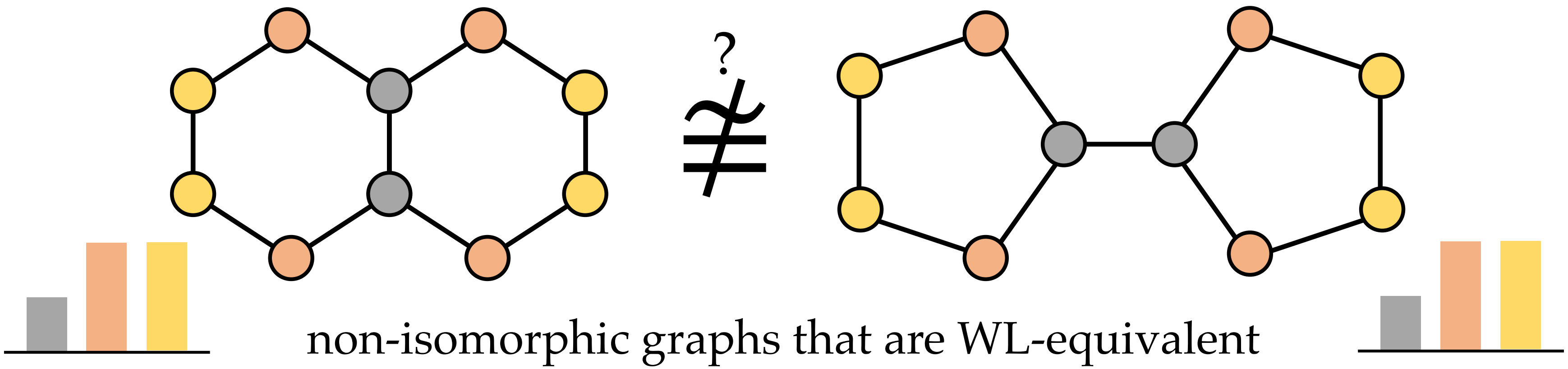
*Weisfeiler-Lehman Test*



*Weisfeiler-Lehman Test*



## *Weisfeiler-Lehman Test*



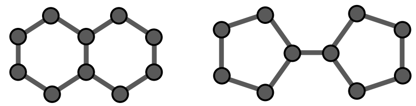

non-isomorphic graphs that are WL-equivalent

**Necessary but insufficient condition for graph isomorphism!**

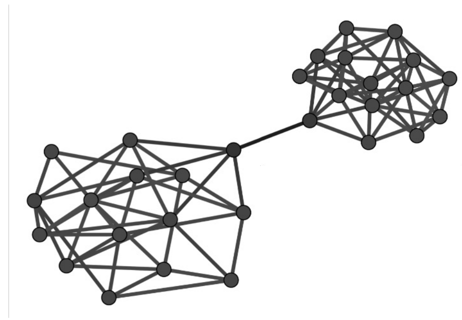
## Theory

WL test = expressive power

## Practice

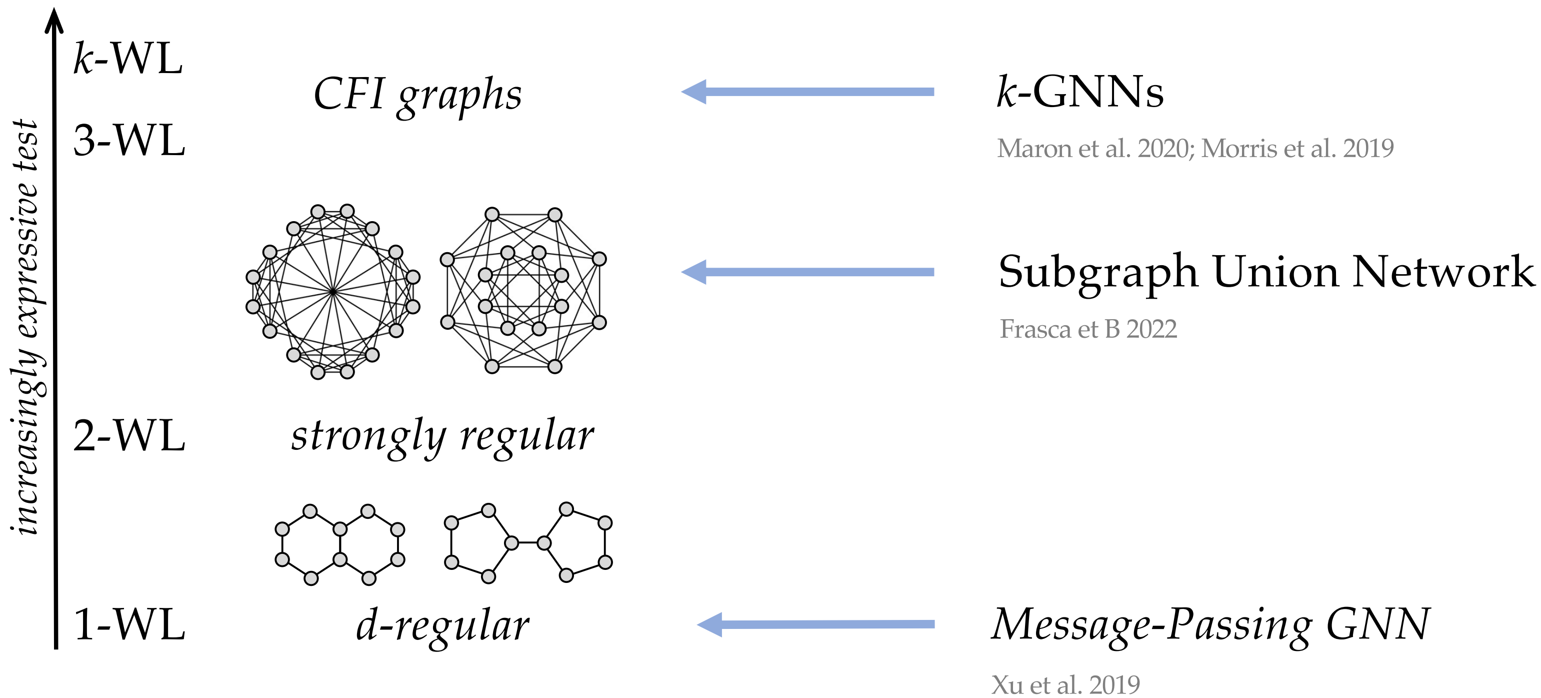


Some non-isomorphic graphs cannot be tested by WL



Some graphs may be unfriendly for message passing

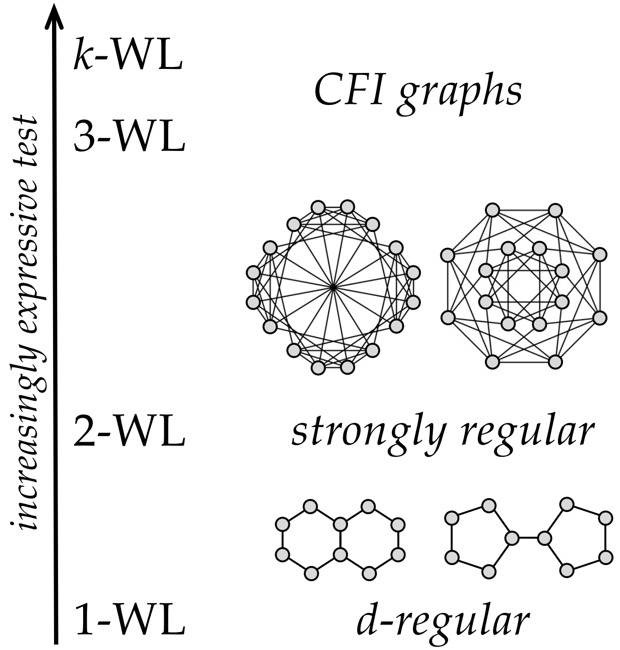
# More expressive isomorphism tests (*k*-WL hierarchy)




## GNN expressive power

Weisfeiler, Lehman 1968 (2-WL); Babai, Mathon 1979 (*k*-WL);
Cai, Furer, Immerman 1992 (CFI graphs)

## More expressive isomorphism tests (*k*-WL hierarchy)

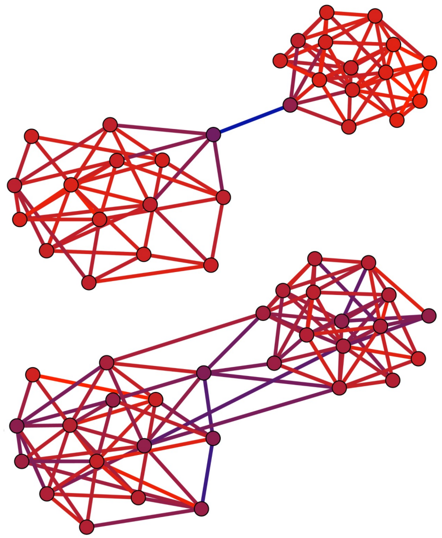



## GNN expressive power

Weisfeiler, Lehman 1968 (2-WL); Babai, Mathon 1979 (*k*-WL); Cai, Fürer, Immerman 1992 (CFI graphs)

# Gap between Theory & Practice

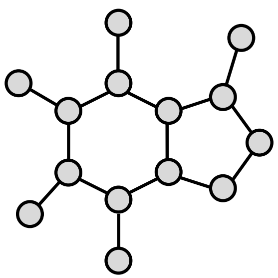
## Decouple input graph from the computational graph



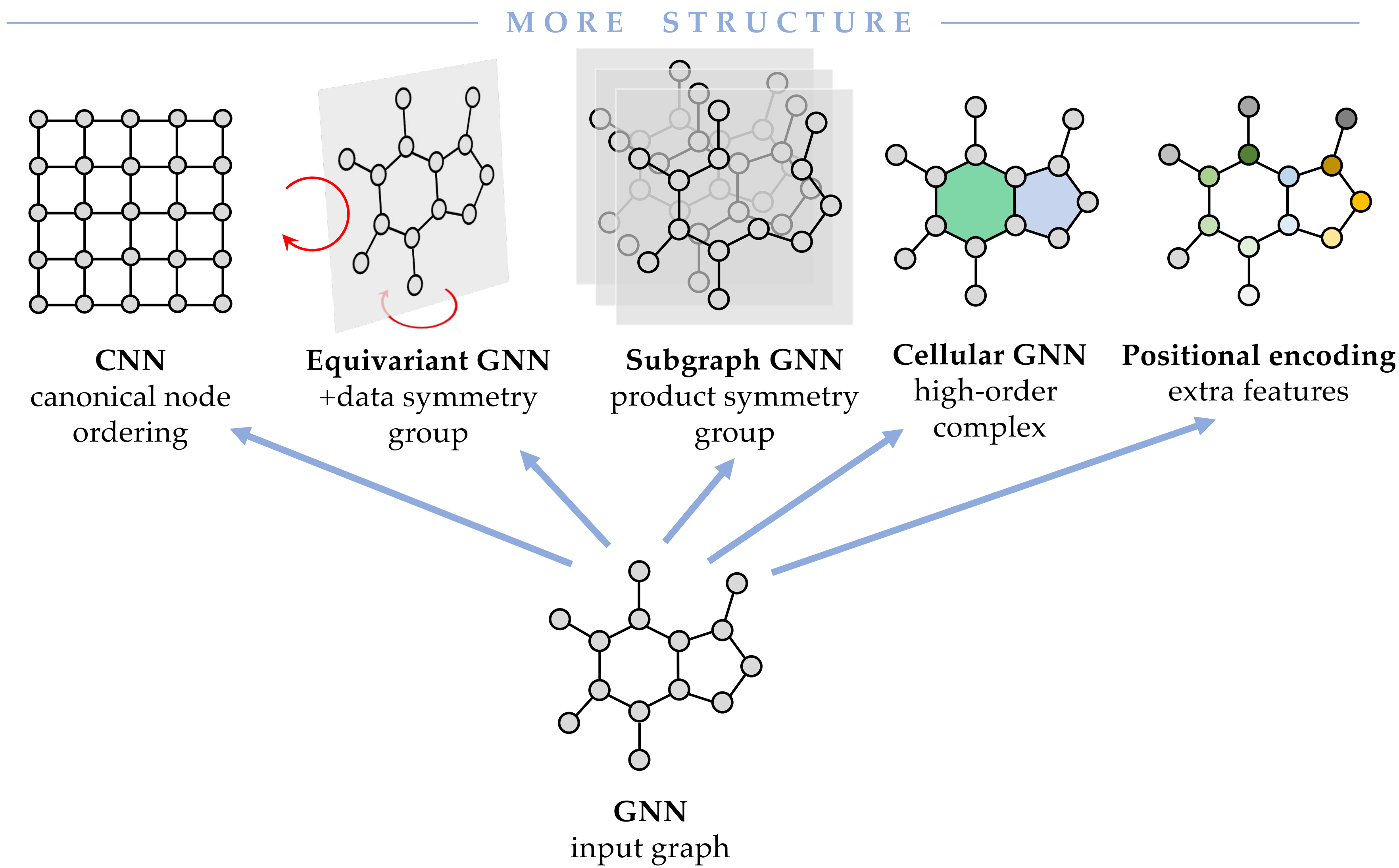
## Graph rewiring

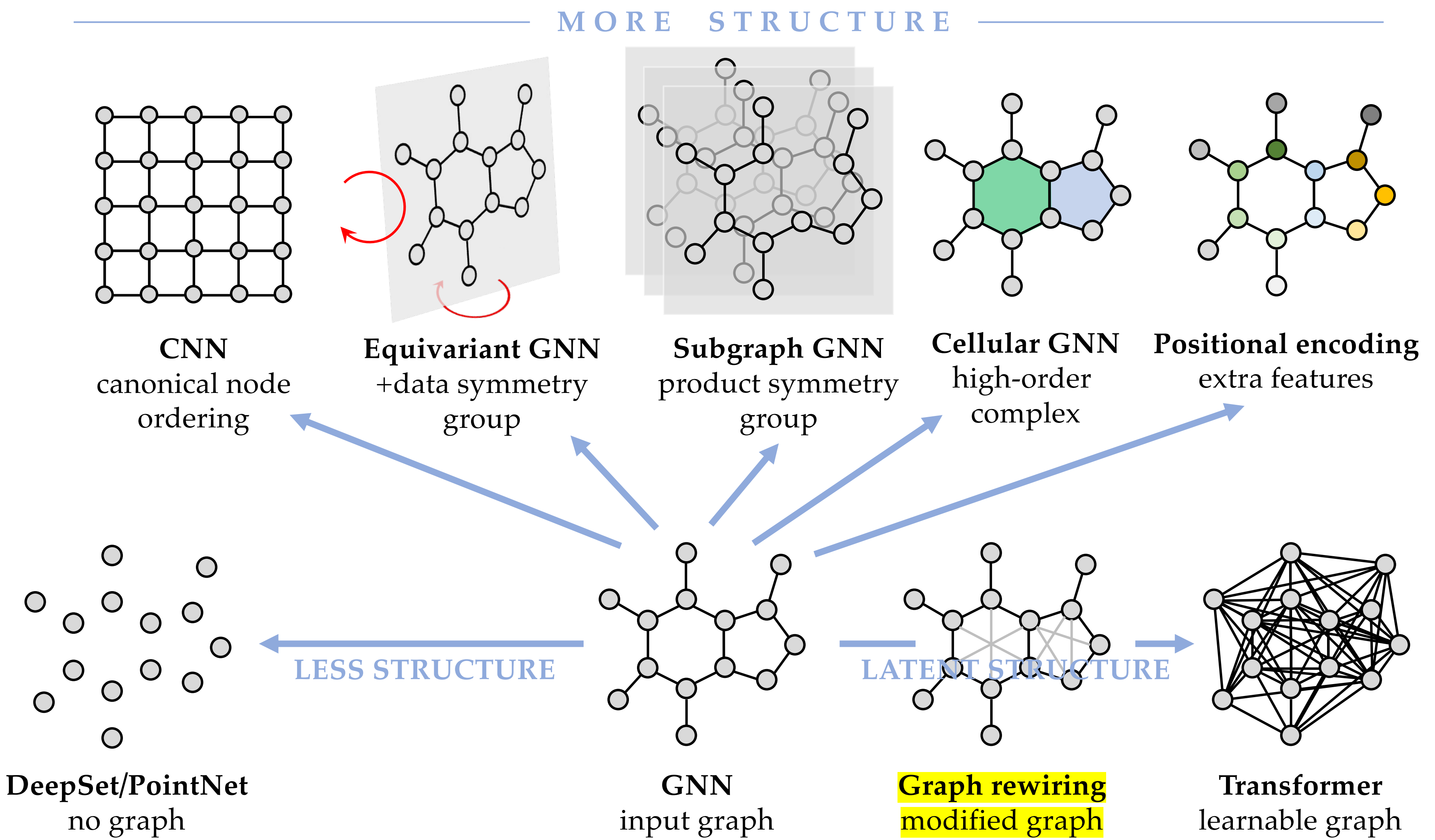
Alon, Yahav 2020 (bottlenecks); Hamilton et al. 2017 (neighbour sampling); Klicpera et al. 2019 (diffusion); Topping, Di Giovanni et B 2022 (Ricci flow); Deac et al. 2022 (expanders); Barbero et B, Di Giovanni 2024 (LASER)

Classical GNNs: propagate information on the input graph

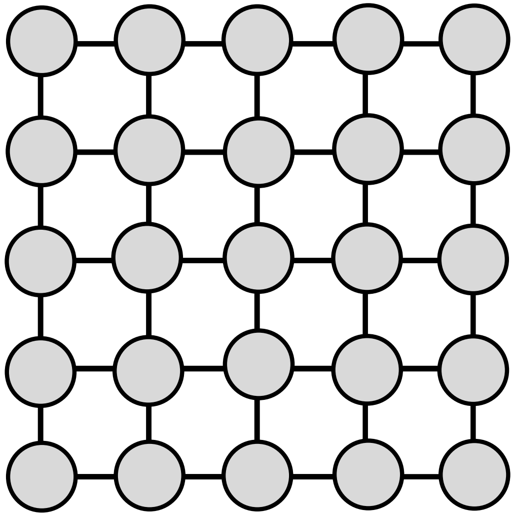


**GNN**
input graph


MORE STRUCTURE
CNN
canonical node ordering
Equivariant GNN
+data symmetry group
Subgraph GNN
product symmetry group
Cellular GNN
high-order complex
Positional encoding
extra features
GNN
input graph

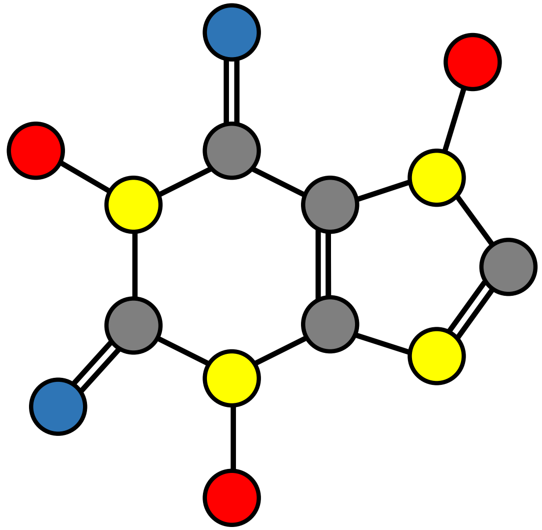

MORE STRUCTURE
CNN
canonical node ordering
Equivariant GNN
+data symmetry group
Subgraph GNN
product symmetry group
Cellular GNN
high-order complex
Positional encoding
extra features
LESS STRUCTURE
LATENT STRUCTURE
DeepSet/PointNet
no graph
GNN
input graph
Graph rewiring
modified graph
Transformer
learnable graph

*Graphs vs Meshes vs Grids*



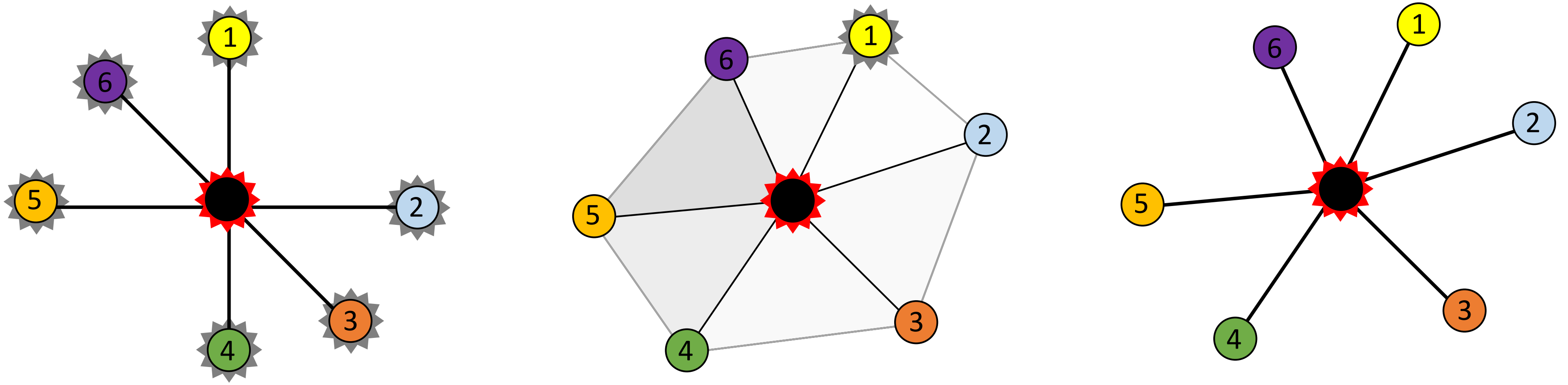
**Grid**

**Mesh**



**Graph**

*Graphs vs Meshes vs Grids*

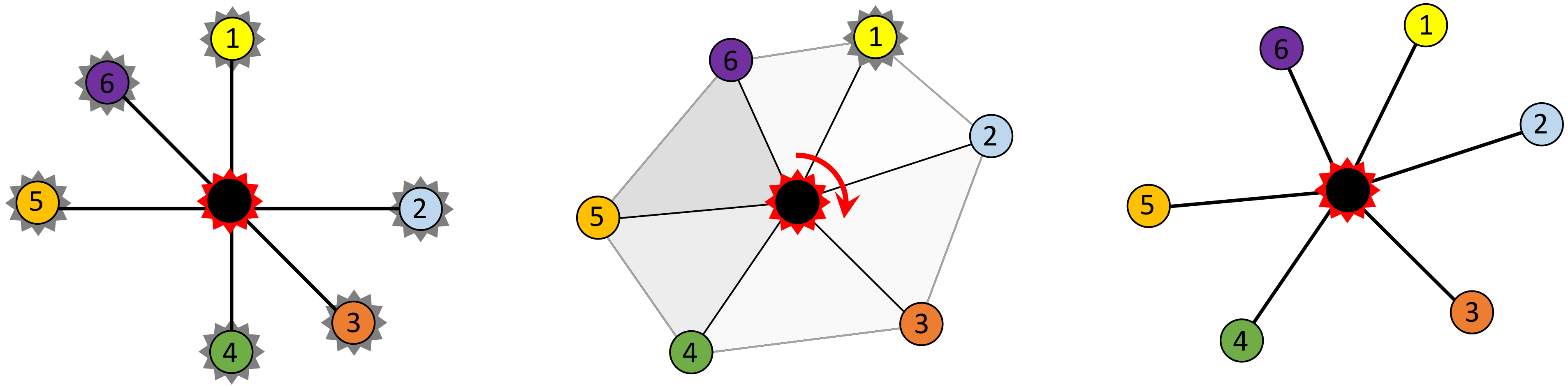



**Grid**
Fixed

**Mesh**

**Graph**

*Graphs vs Meshes vs Grids*

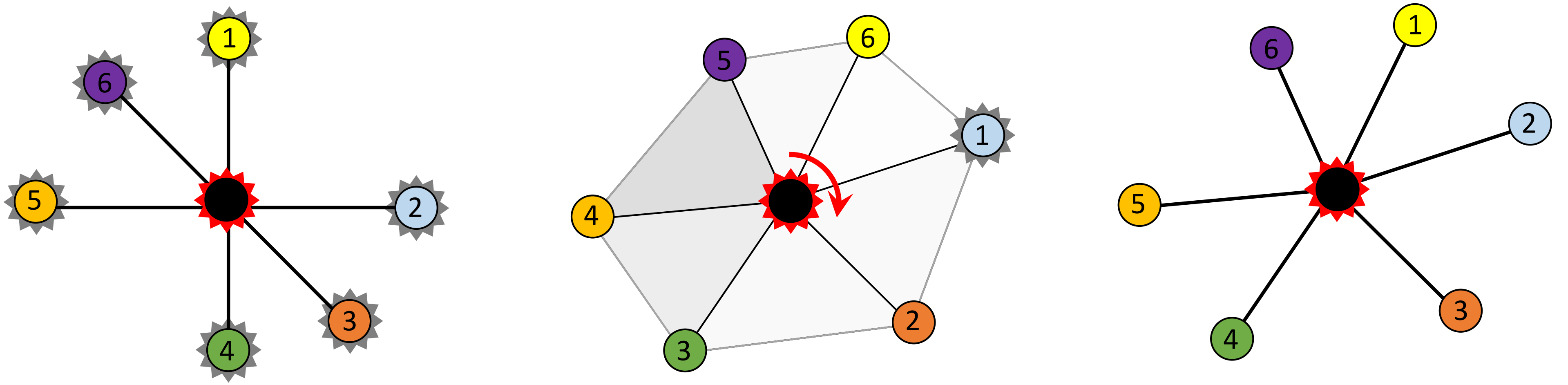



**Grid**
Fixed

**Mesh**
Rotation

**Graph**

*Graphs vs Meshes vs Grids*



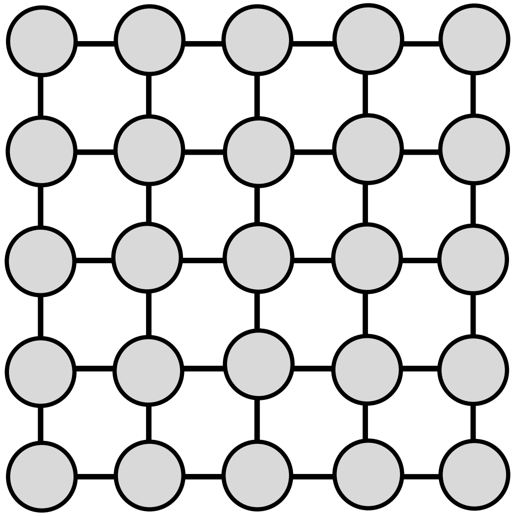

**Grid** Fixed

**Mesh** Rotation

**Graph** Permutation

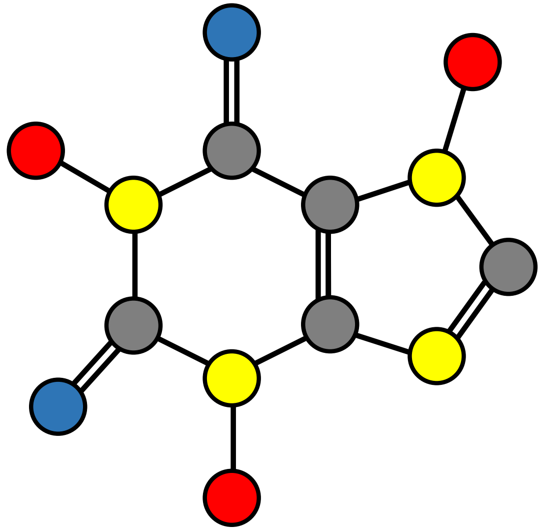
Graphs have the least structure

*Graphs vs Meshes vs Grids*



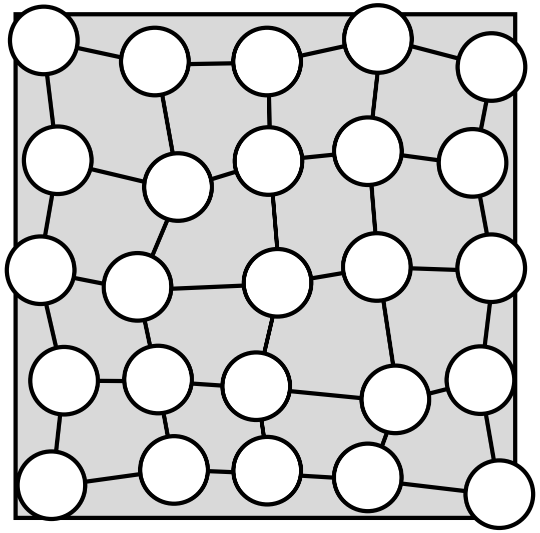
**Grid**

**Mesh**



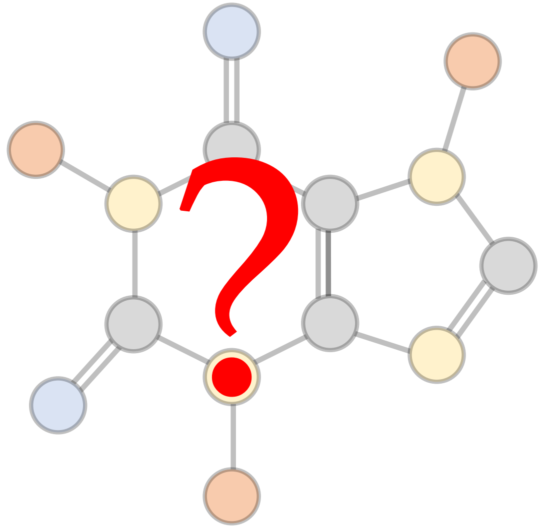
**Graph**

*Graphs vs Meshes vs Grids*



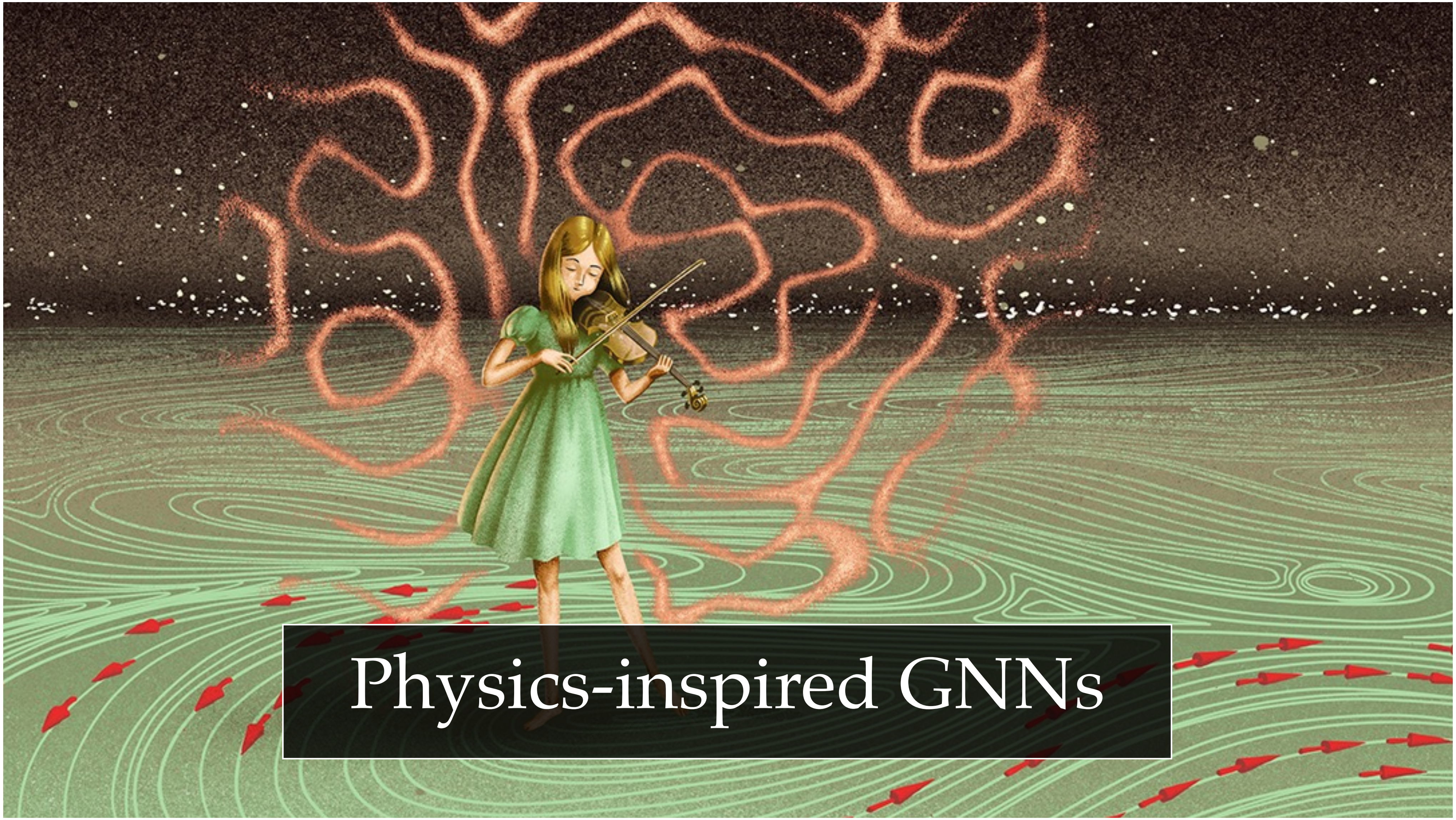
**Grid**

**Mesh**



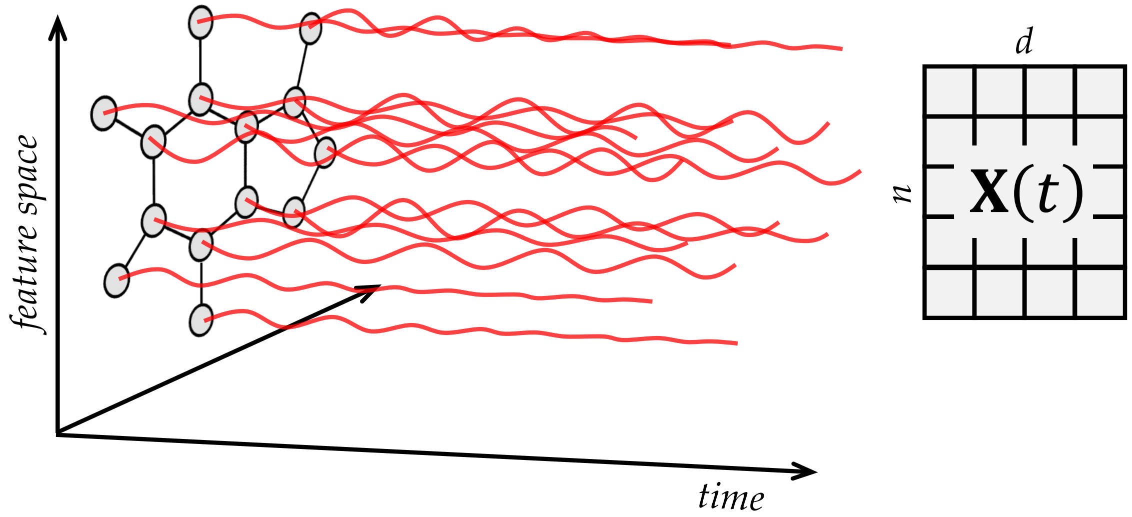
**Graph**

Continuous models for GNNs?


Physics-inspired GNNs

# *Physical metaphor of Graph ML*

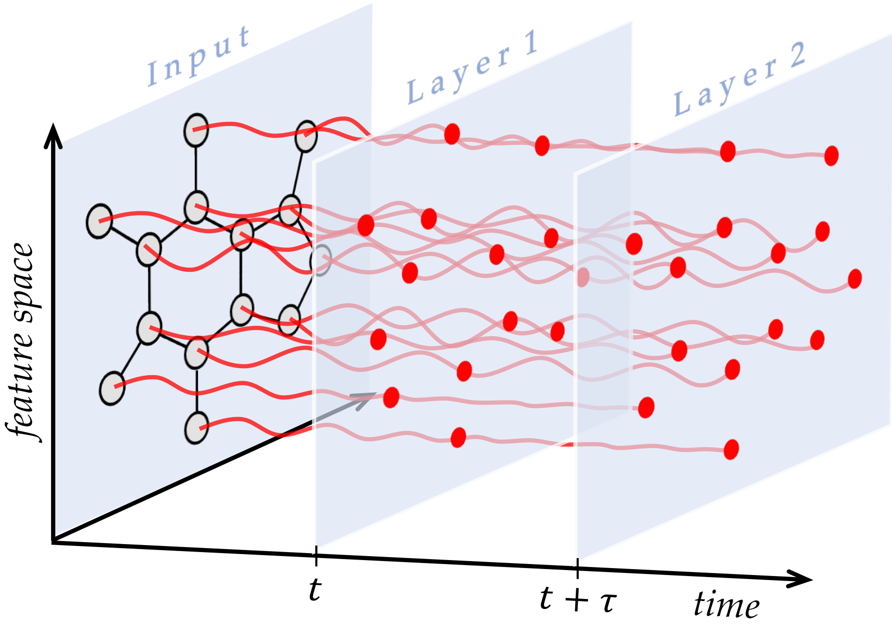
- GNN = dynamic system




$$\dot{\mathbf{X}}(t) = \mathbf{F}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

Haber, Ruthotto 2017; Chen et al. 2019 (Neural ODEs); Xhonneau et al. 2020 (CGNN); Chamberlain, Rowbottom, et B. 2021 (GRAND, BLEND)
Eliasof, Haber 2021 (PDE-GCN); Di Giovanni, Rowbottom et B 2022 (GRAFF), Rusch et B 2022 (GraphCON)

# *Physical metaphor of Graph ML*



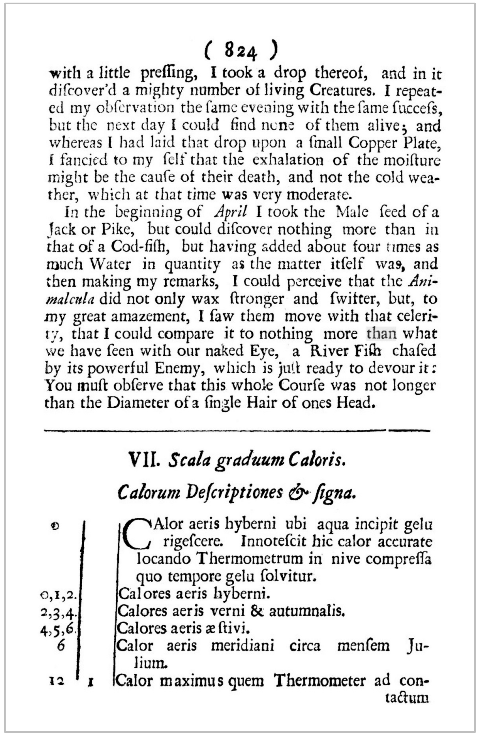

- GNN = dynamic system
- layers = discretisation of time
- graph = coupling function (discretisation of space)

$$\mathbf{X}(t+\tau) = \mathbf{X}(t) + \tau \mathbf{F}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

Haber, Ruthotto 2017; Chen et al. 2019 (Neural ODEs); Xhonneau et al. 2020 (CGNN); Chamberlain, Rowbottom, et B. 2021 (GRAND, BLEND)
Eliasof, Haber 2021 (PDE-GCN); Di Giovanni, Rowbottom et B 2022 (GRAFF), Rusch et B 2022 (GraphCON)

*Heat Diffusion*

**Newton Law of Cooling:**
"the [temperature] a hot body loses in a given time is proportional to the temperature difference between the object and the environment"


( 824 )
with a little preſſing, I took a drop thereof, and in it diſcover'd a mighty number of living Creatures. I repeated my obſervation the ſame evening with the ſame ſucceſs, but the next day I could find none of them alive; and whereas I had laid that drop upon a ſmall Copper Plate, I fancied to my ſelf that the exhalation of the moiſture might be the cauſe of their death, and not the cold weather, which at that time was very moderate.

In the beginning of *April* I took the Male ſeed of a Jack or Pike, but could diſcover nothing more than in that of a Cod-fiſh, but having added about four times as much Water in quantity as the matter itſelf was, and then making my remarks, I could perceive that the *Animalcula* did not only wax ſtronger and ſwifter, but, to my great amazement, I ſaw them move with that celerity, that I could compare it to nothing more than what we have ſeen with our naked Eye, a River Fiſh chaſed by its powerful Enemy, which is juſt ready to devour it: You muſt obſerve that this whole Courſe was not longer than the Diameter of a ſingle Hair of ones Head.

VII. *Scala graduum Caloris.*

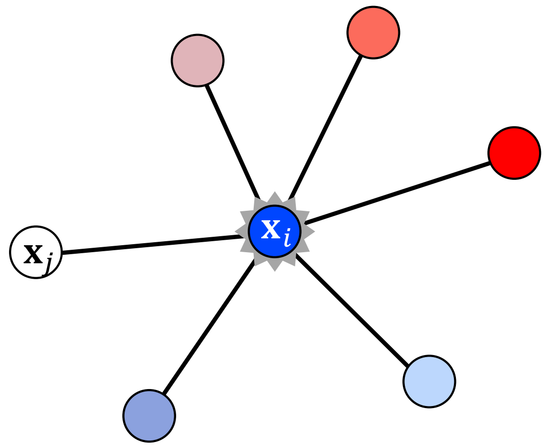
*Calorum Deſcriptiones & ſigna.*

| | | |
|---|---|---|
| 0 | | Calor aeris hyberni ubi aqua incipit gelu rigeſcere. Innoteſcit hic calor accurate locando Thermometrum in nive compreſſa quo tempore gelu ſolvitur. |
| 0,1,2. | | Calores aeris hyberni. |
| 2,3,4. | | Calores aeris verni & autumnalis. |
| 4,5,6. | | Calores aeris æſtivi. |
| 6 | | Calor aeris meridiani circa menſem Julium. |
| 12 | 1 | Calor maximus quem Thermometer ad contactum |


**I. Newton**

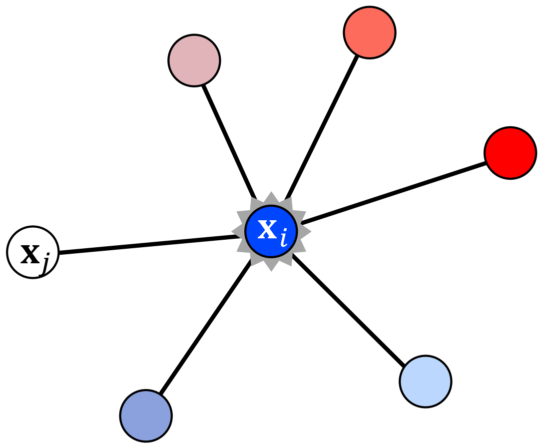
Anonymous 1701

*Heat Diffusion Equation on Graphs*




$$\dot{\mathbf{x}}_i(t) = \mathbf{x}_i(t) - \frac{1}{d_i}\sum_{j\in\mathcal{N}_i} a_{ij}\mathbf{x}_j(t)$$

## *Heat Diffusion Equation on Graphs*



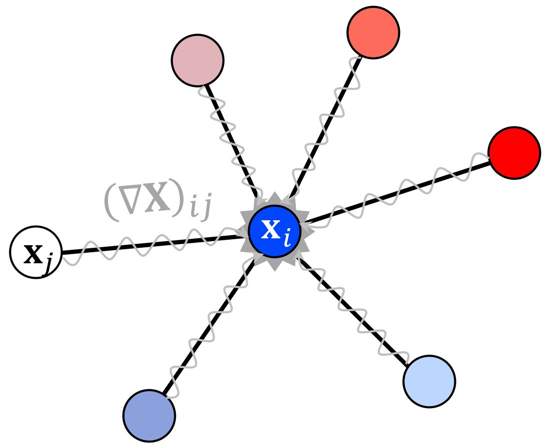

$$\dot{\mathbf{x}}_i(t) = \mathbf{x}_i(t) - \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} a_{ij} \mathbf{x}_j(t)$$

rate of temperature change

self temperature

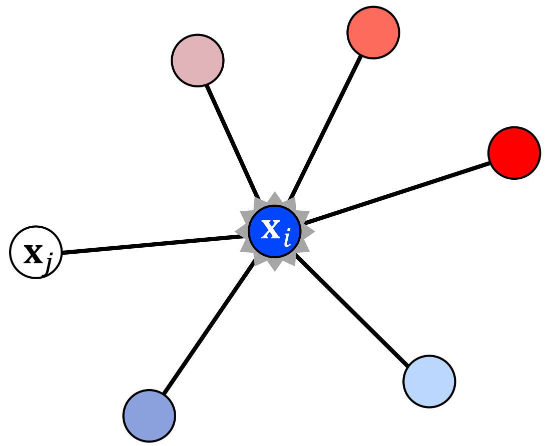
temperature of the environment

*Heat Diffusion Equation on Graphs*



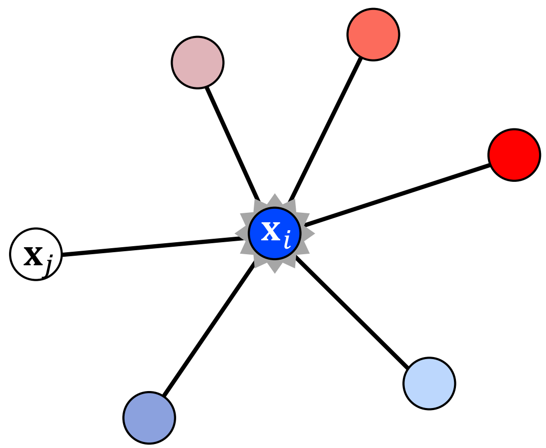

$$\dot{\mathbf{x}}_i(t) = \underbrace{\frac{1}{d_i} \sum_{j \in \mathcal{N}_i}}_{\substack{\text{divergence} \\ \text{div}}} a_{ij} \underbrace{\left(\mathbf{x}_i(t) - \mathbf{x}_j(t)\right)}_{\substack{\text{gradient} \\ -(\nabla \mathbf{X})_{ij}}}$$

*Heat Diffusion Equation on Graphs*




$$\dot{\mathbf{X}}(t) = -\mathrm{div}\big(\nabla \mathbf{X}(t)\big)$$

*Heat Diffusion Equation on Graphs*




$$\dot{\mathbf{X}}(t) = \Delta \mathbf{X}(t)$$

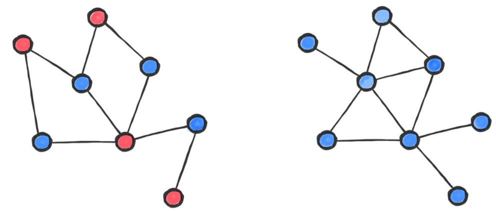
## *Heat Diffusion Equation as a prototypical Gradient Flow*

$$\dot{\mathbf{X}}(t) = -\nabla \mathcal{E}(\mathbf{X}(t))$$

$$\mathcal{E}_{\mathrm{DIR}}(\mathbf{X}) = \frac{1}{2} \sum_{j \in \mathcal{N}_i} \left\| (\nabla \mathbf{X})_{ij} \right\|^2 = \frac{1}{2} \mathrm{trace}\left(\mathbf{X}^{\mathrm{T}} \Delta \mathbf{X}\right)$$

**G. Dirichlet**

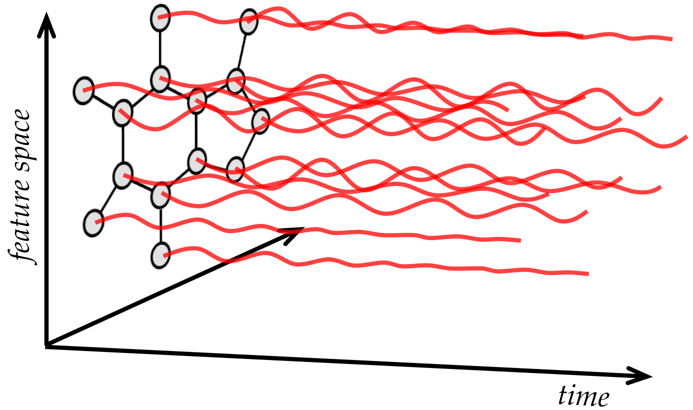
- Heat equation is the gradient flow of the Dirichlet energy
- "Smoothness" of the node features
- Dirichlet energy decreases along the flow
- In the limit $t \to \infty$ results in "oversmoothing"
- Not very expressive: works only in homophilic graphs ("similar neighbours")



*heterophilic* *homophilic*

Zhou, Schölkopf 2005 (label propagation); Rossi et B 2021 (feature propagation)

# *Gradient Flow Framework (GRAFF)*

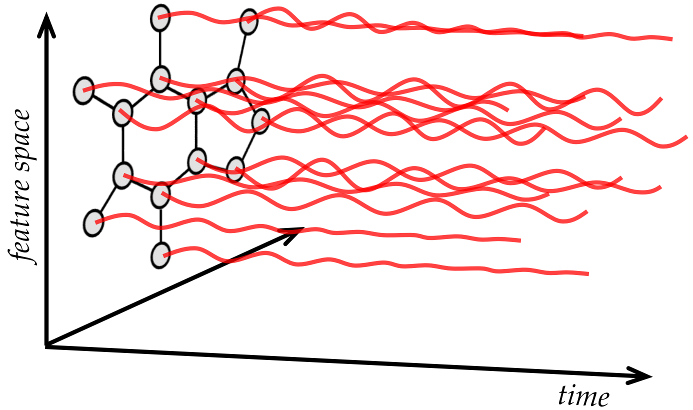



**Traditional GNNs**

$$\dot{\mathbf{X}}(t) = \mathbf{F}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

Di Giovanni, Rowbottom et B 2022

# *Gradient Flow Framework (GRAFF)*




**Traditional GNNs**

$$\mathbf{X}(k+1) = \mathbf{X}(k) + \tau \mathbf{F}_{\boldsymbol{\theta}(k)}(\mathbf{X}(k), \mathcal{G})$$

- Parametrize evolution equations

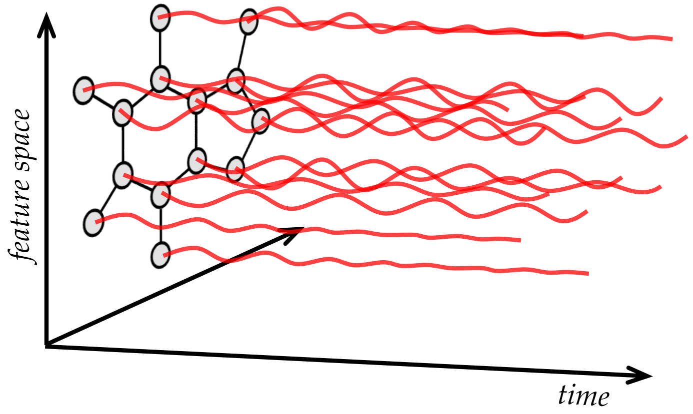
**GRAFF**

$$\mathcal{E}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

- Parametrize energy

Di Giovanni, Rowbottom et B 2022

# *Gradient Flow Framework (GRAFF)*




**Traditional GNNs**

$$\mathbf{X}(k+1) = \mathbf{X}(k) + \tau \mathbf{F}_{\boldsymbol{\theta}(k)}(\mathbf{X}(k), \mathcal{G})$$

- Parametrize evolution equations

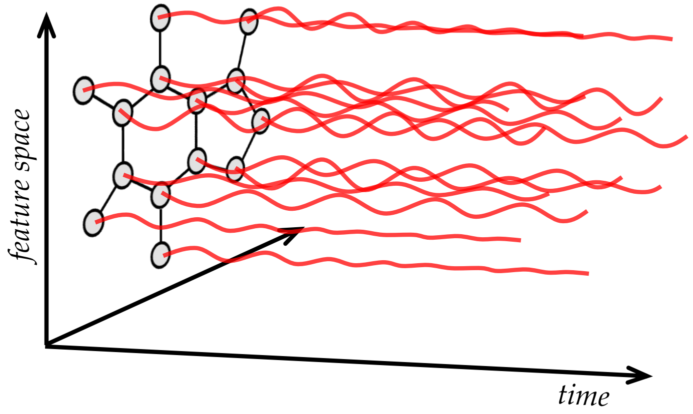
**GRAFF**

$$\dot{\mathbf{X}}(t) = -\nabla \mathcal{E}_{\boldsymbol{\theta}(t)}(\mathbf{X}(t), \mathcal{G})$$

- Parametrize energy
- Derive evolution equation as GF

Di Giovanni, Rowbottom et B 2022

# *Gradient Flow Framework (GRAFF)*




**Traditional GNNs**

$$\mathbf{X}(k+1) = \mathbf{X}(k) + \tau \mathbf{F}_{\boldsymbol{\theta}(k)}(\mathbf{X}(k), \mathcal{G})$$

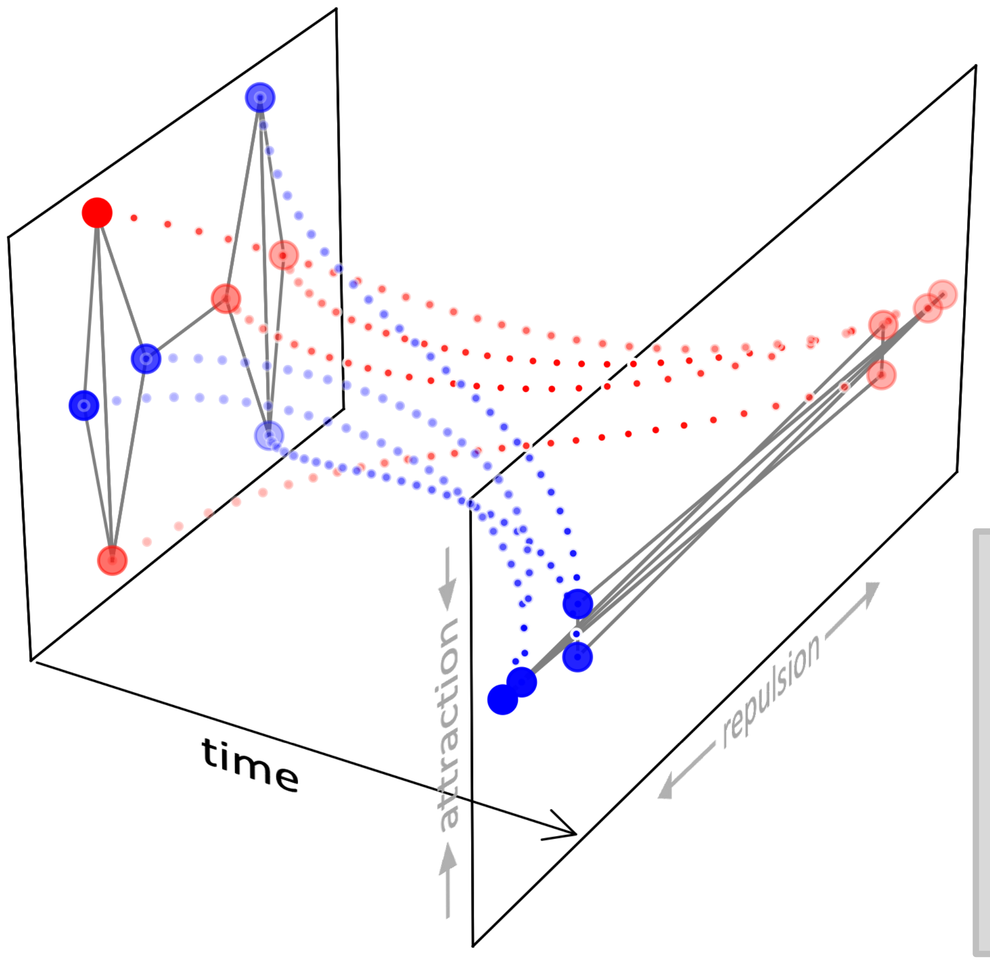
- Parametrize evolution equations

**GRAFF**

$$\mathbf{X}(k+1) = \mathbf{X}(k) - \tau \nabla \mathcal{E}_{\boldsymbol{\theta}(k)}(\mathbf{X}(k), \mathcal{G})$$

- Parametrize energy
- Derive evolution equation as GF
- Better "interpretability"

Di Giovanni, Rowbottom et B 2022



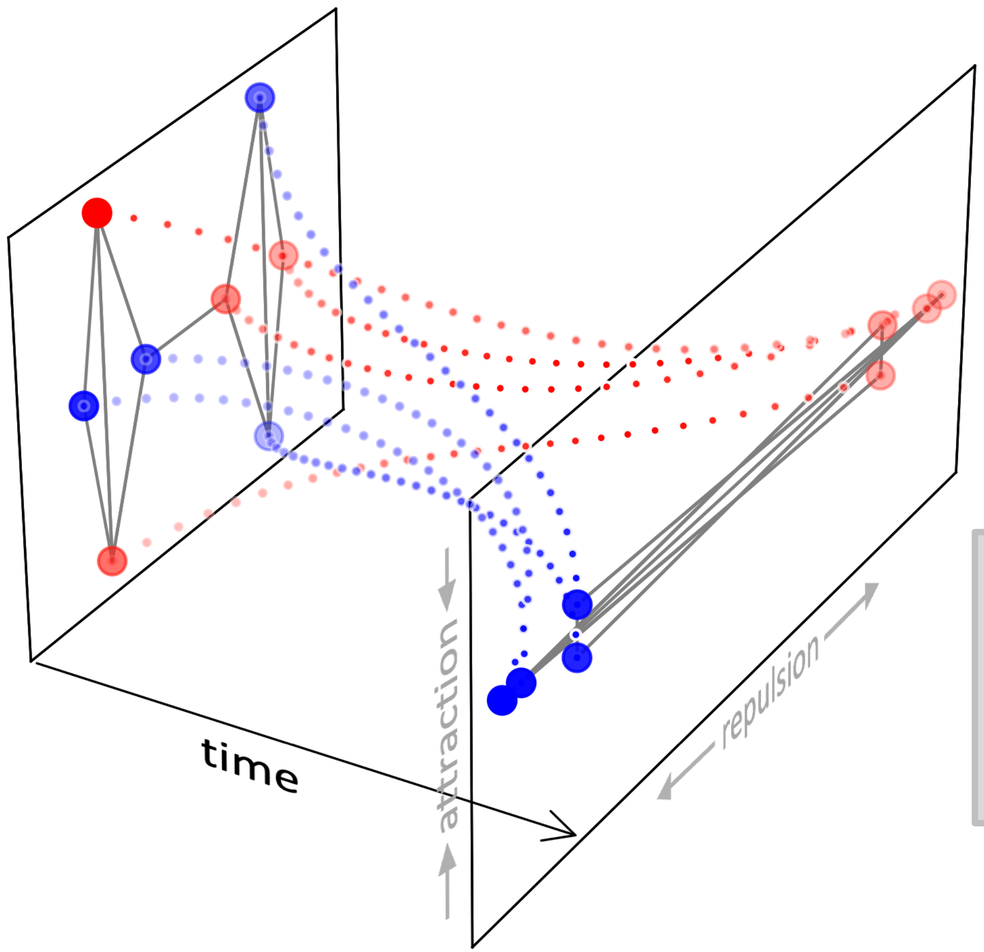

Di Giovanni, Rowbottom et B 2022

$$\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{X}) = -\frac{1}{2}\sum_{j\in\mathcal{N}_i} \bar{a}_{ij}\langle \mathbf{x}_i, \mathbf{W}\mathbf{x}_j\rangle$$

- **Attraction** along positive eigenvectors of **W**
- **Repulsion** along negative eigenvectors of **W**

$$\dot{\mathbf{X}}(t) = \bar{\mathbf{A}}\mathbf{X}(t)\mathbf{W}$$

**Theorem:** Linear graph diffusion ("convolutional GNN") with appropriately designed channel mixing matrix **W** (symmetric & with sufficiently large negative eigenvalues) can provable avoid oversmoothing.




Di Giovanni, Rowbottom et B 2022

$$\mathcal{E}_{\boldsymbol{\theta}}(\mathbf{X}) = -\frac{1}{2}\sum_{j\in\mathcal{N}_i} \bar{a}_{ij}\langle \mathbf{x}_i, \mathbf{W}\mathbf{x}_j\rangle$$

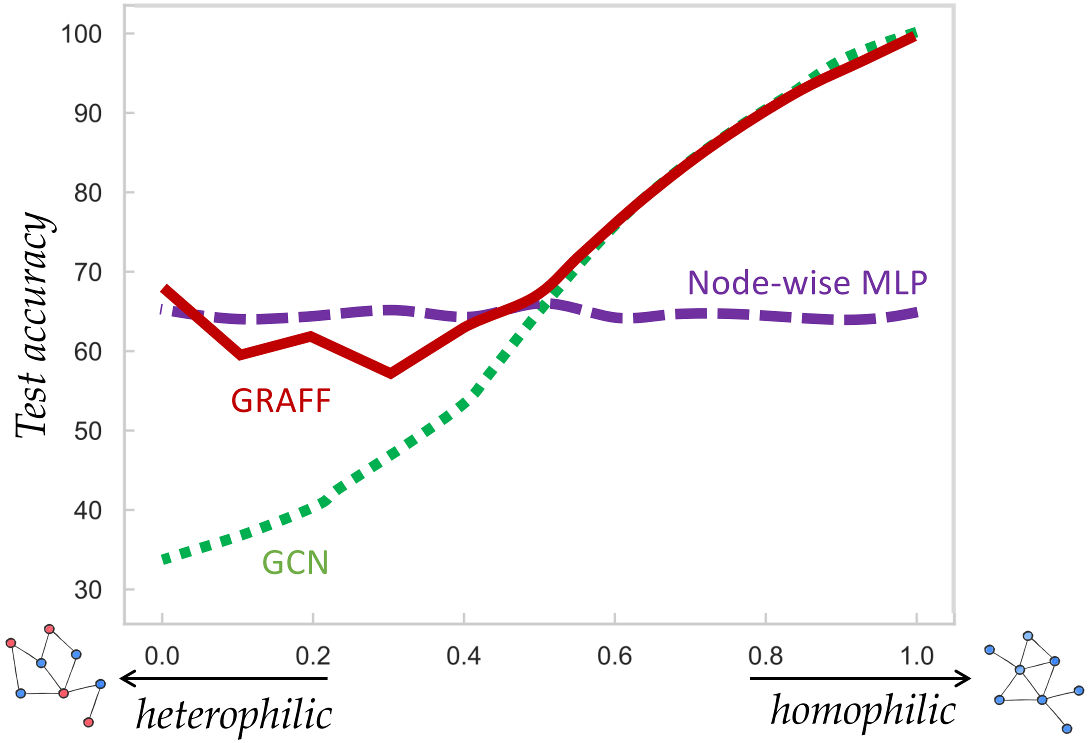
- **Attraction** along positive eigenvectors of **W**
- **Repulsion** along negative eigenvectors of **W**

$$\dot{\mathbf{X}}(t) = \bar{\mathbf{A}}\mathbf{X}(t)\mathbf{W}$$

**Theorem:** Linear graph diffusion ("convolutional GNN") with appropriately designed channel mixing matrix **W** can avoid oversmoothing.

**Contradicts GNN "folklore"!**

# *Homophily vs Heterophily*

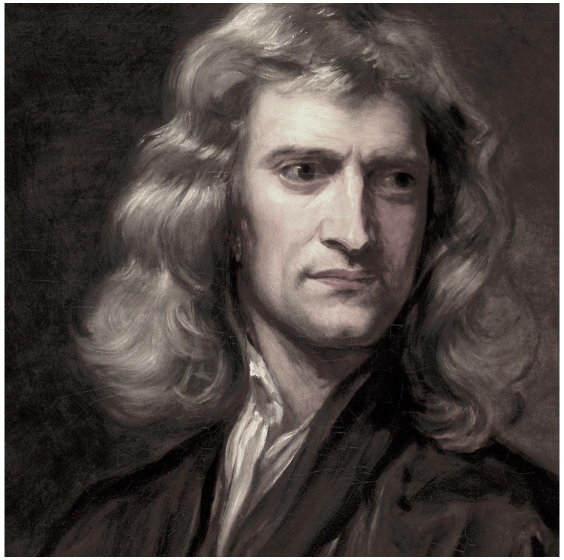



Synthetic Cora node classification task

Di Giovanni, Rowbottom et B 2022

*Homogeneous Diffusion in Image Processing*

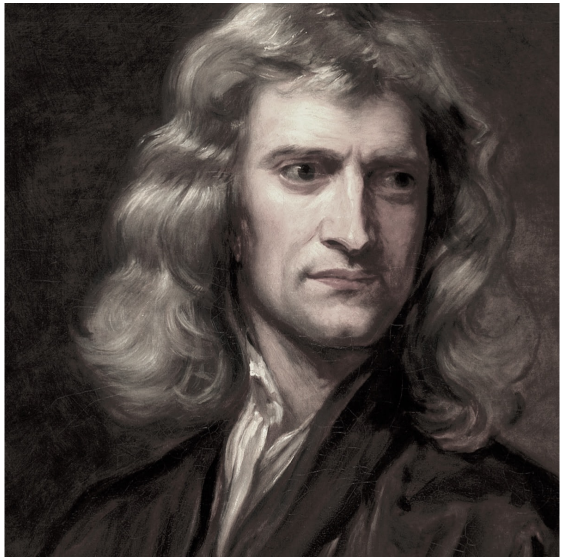
$$\dot{\mathbf{X}}(t) = -\operatorname{div}\big(c\nabla\mathbf{X}(t)\big)$$



$\mathbf{X}(0)$

*Homogeneous Diffusion in Image Processing*

$$\dot{\mathbf{X}}(t) = -\mathrm{div}\big(c\nabla\mathbf{X}(t)\big)$$



$\mathbf{X}(0)$

$\mathbf{X}(t) = \mathbf{X}(0) \star \mathbf{G}_{\sigma\propto t}$

*Non-homogeneous Diffusion in Image Processing*

$$\dot{\mathbf{X}}(t) = -\mathrm{div}\left(\frac{\nabla \mathbf{X}(t)}{1 + c\|\nabla \mathbf{X}(t)\|^2}\right)$$

edge indicator

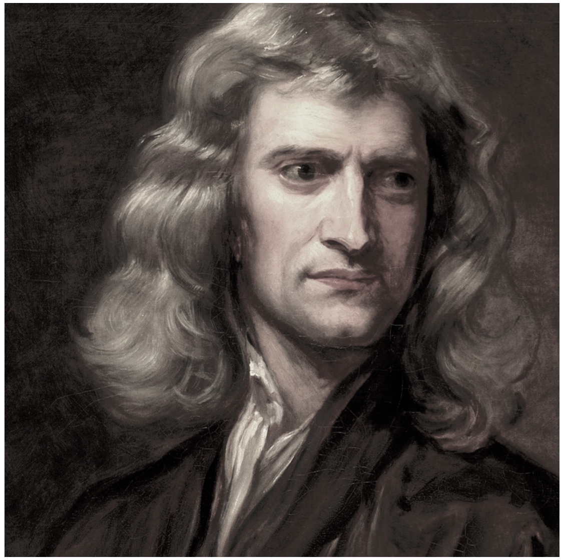
$\mathbf{X}(0)$ “Do not diffuse across edges” $\mathbf{X}(t) = \mathbf{X}(0) \star \mathbf{G}_{\sigma \propto t}$

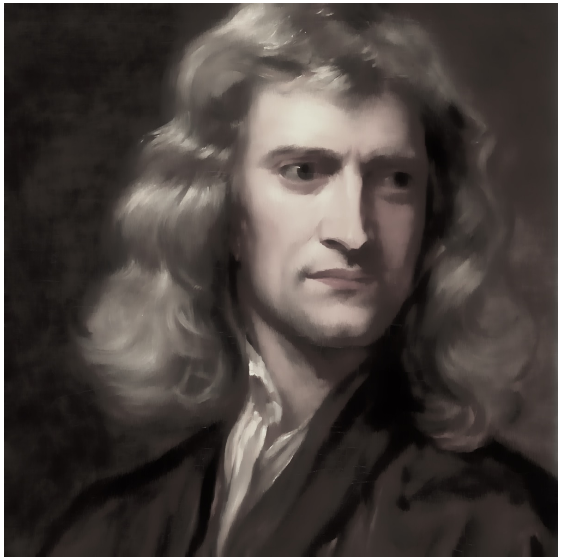
Perona, Malik 1990

*Non-homogeneous Diffusion in Image Processing*

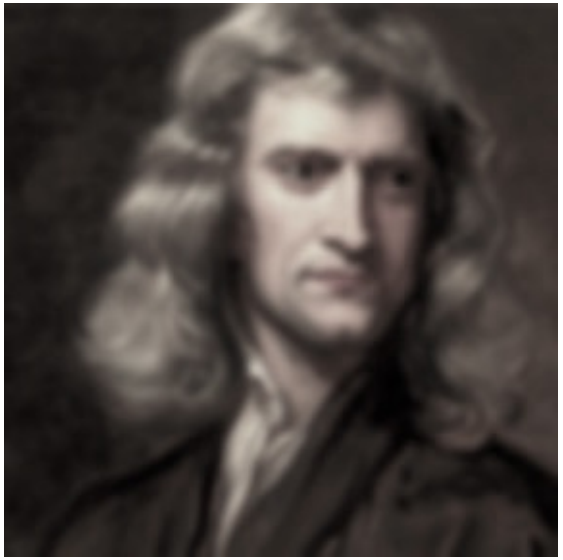
$$\dot{\mathbf{X}}(t) = -\mathrm{div}\left(\frac{\nabla \mathbf{X}(t)}{1 + c\|\nabla \mathbf{X}(t)\|^2}\right)$$

edge indicator





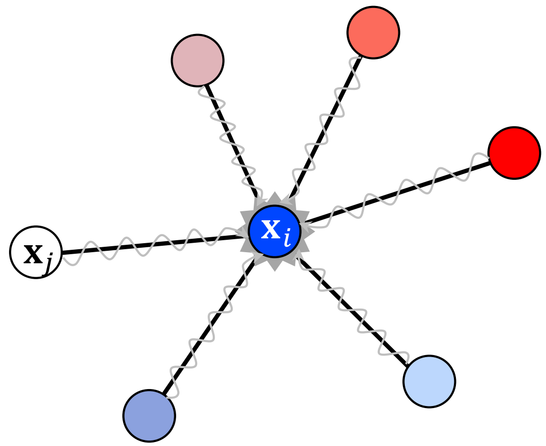
Non-homogeneous



Homogeneous

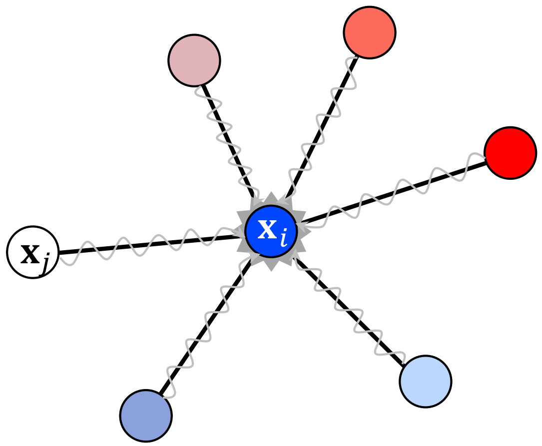
Perona, Malik 1990

*Non-homogeneous Diffusion on Graphs*




$$\dot{\mathbf{x}}_i(t) = \sum_{j \in \mathcal{N}_i} a\left(\mathbf{x}_i(t), \mathbf{x}_j(t)\right)\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)$$

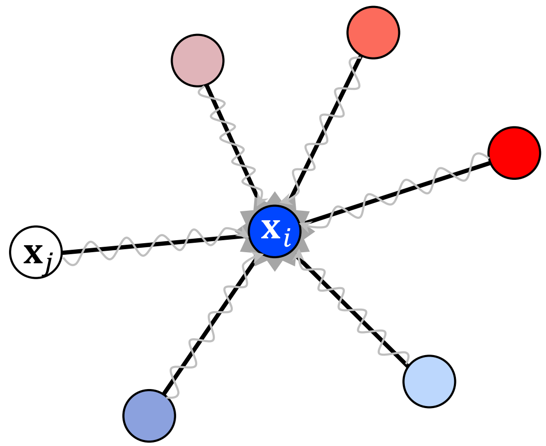
*Non-homogeneous Diffusion on Graphs*




$$\dot{\mathbf{x}}_i(t) = \sum_{j \in \mathcal{N}_i} a\left(\mathbf{x}_i(t), \mathbf{x}_j(t)\right)\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)$$

learnable diffusivity

*Non-homogeneous Diffusion on Graphs*



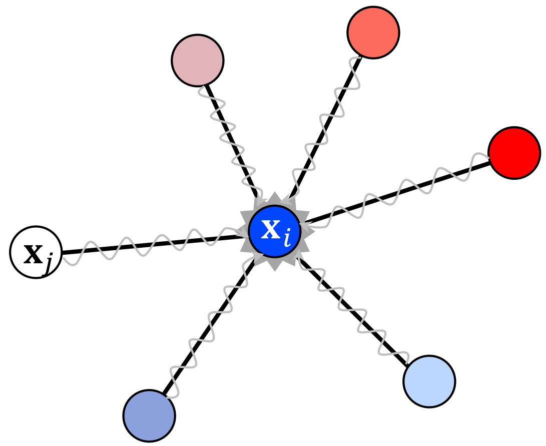

$$\mathbf{x}_i(t+\tau) = \mathbf{x}_i(t) + \tau \sum_{j \in \mathcal{N}_i} a\Big(\mathbf{x}_i(t), \mathbf{x}_j(t)\Big)\Big(\mathbf{x}_j(t) - \mathbf{x}_i(t)\Big)$$

Explicit (Forward Euler) discretization

time step

learnable diffusivity

*Non-homogeneous Diffusion on Graphs*



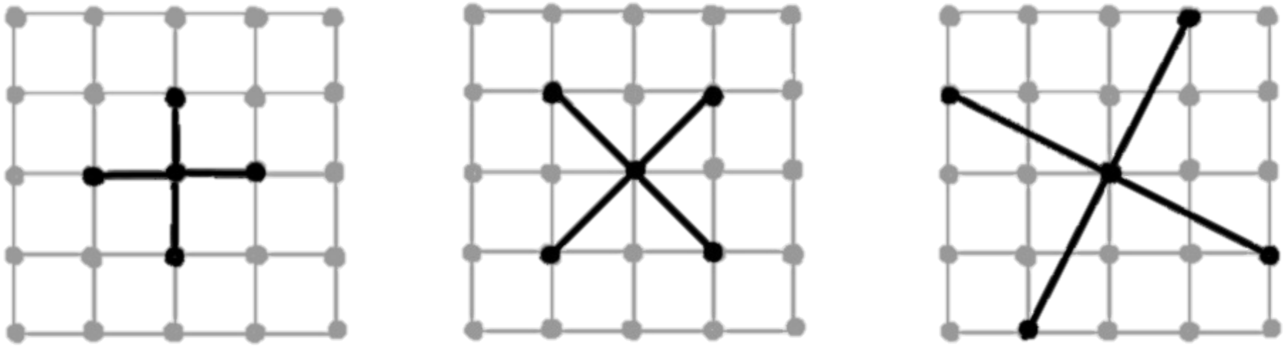

$$\mathbf{x}_i(t+\tau) = \sum_{j \in \mathcal{N}_i} a\Big(\mathbf{x}_i(t), \mathbf{x}_j(t)\Big)\, \mathbf{x}_j(t)$$

GAT!

normalised $\sum_j a_{ij} = 1$

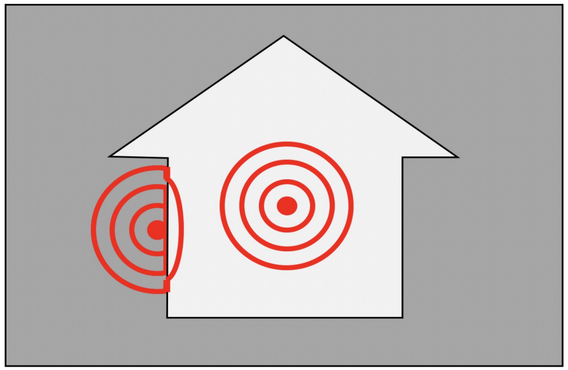
unit step $\tau = 1$

*Spatial Derivative: Graph Rewiring?*



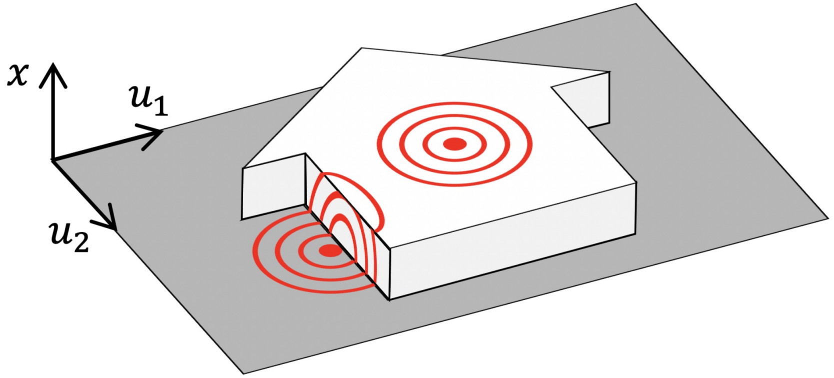
Different discretisations of 2D Laplacian

## *Images as embedded manifolds*



$$\dot{\mathbf{X}} = -\mathrm{div}(a(\mathbf{X})\nabla\mathbf{X})$$

**Non-linear diffusion**




$$\dot{\mathbf{Z}} = \Delta_{\mathbf{G}}\mathbf{Z}$$

**Non-Euclidean diffusion**

Kimmel et al. 1997; Sochen et al. 1998

# *Beltrami flow*

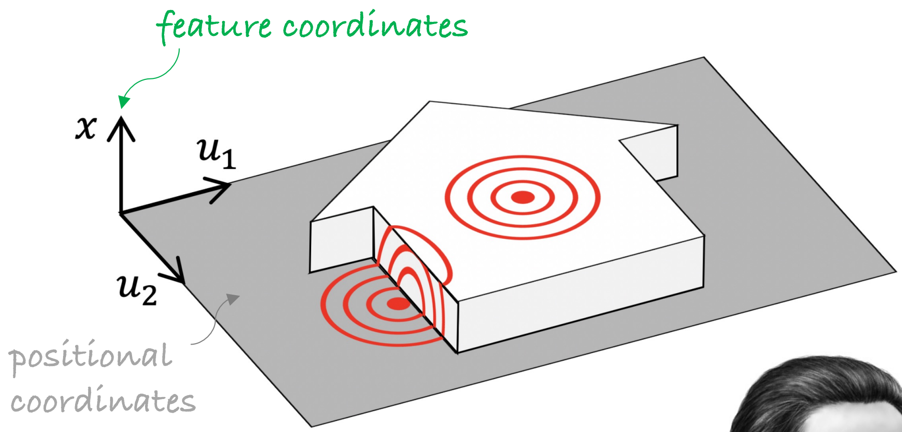
- Consider image as embedded 2-*manifold*

$$\mathbf{Z}(\mathbf{u}) = \big(\mathbf{u}, \alpha\mathbf{X}(\mathbf{u})\big)$$

- *Pullback metric:* 2×2 matrix

$$\mathbf{G} = \mathbf{I} + \alpha^2\big(\nabla_{\mathbf{u}}\mathbf{X}(\mathbf{u})\big)^\top \nabla_{\mathbf{u}}\mathbf{X}(\mathbf{u})$$

- *Beltrami flow* = gradient flow of the *Polyakov energy* (harmonic energy of the embedding used in string theory)




$$\dot{\mathbf{Z}} = \Delta_{\mathbf{G}}\mathbf{Z}$$

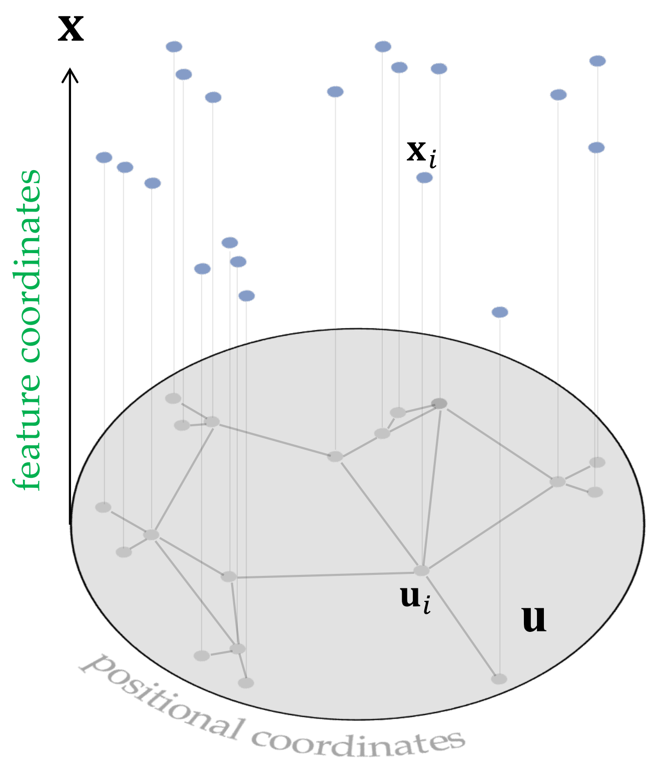
**E. Beltrami**

Kimmel et al. 1997; Sochen et al. 1998

## *Graph Beltrami flow*

- Graph with positional and feature node coordinates $\mathbf{z}_i = (\mathbf{u}_i, \mathbf{x}_i)$
- **Graph Beltrami flow**

$$\dot{\mathbf{z}}_i(t) = \sum_{j \in \mathcal{N}_i} a\left(\mathbf{z}_i(t), \mathbf{z}_j(t)\right)\left(\mathbf{z}_j(t) - \mathbf{z}_i(t)\right)$$



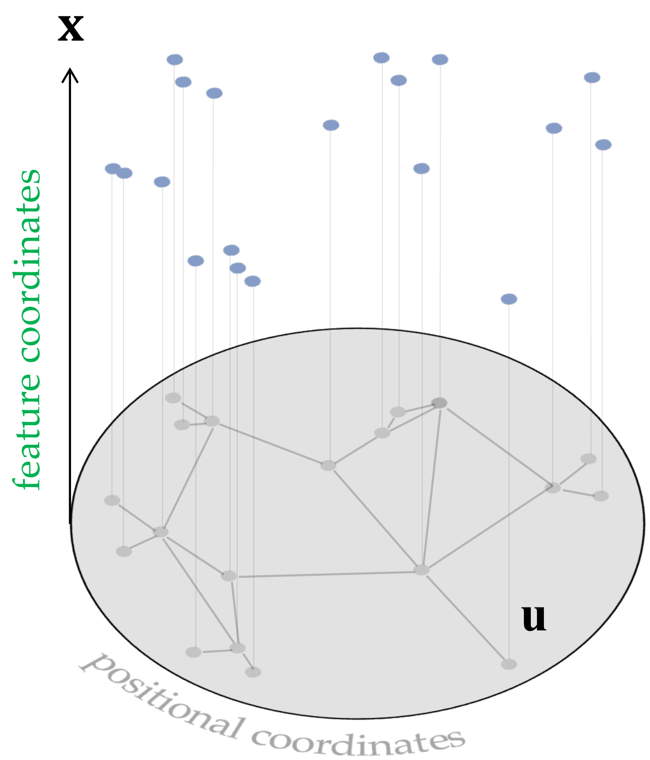

Chamberlain, Rowbottom, et B. 2021

# *Graph Beltrami flow*

- Graph with positional and feature node coordinates $\mathbf{z}_i = (\mathbf{u}_i, \mathbf{x}_i)$
- **Graph Beltrami flow**

$$\dot{\mathbf{z}}_i(t) = \sum_{j \in \mathcal{N}_i} a\Big(\mathbf{z}_i(t), \mathbf{z}_j(t)\Big)\Big(\mathbf{z}_j(t) - \mathbf{z}_i(t)\Big)$$

  - Evolution of $\mathbf{x}$ = feature diffusion



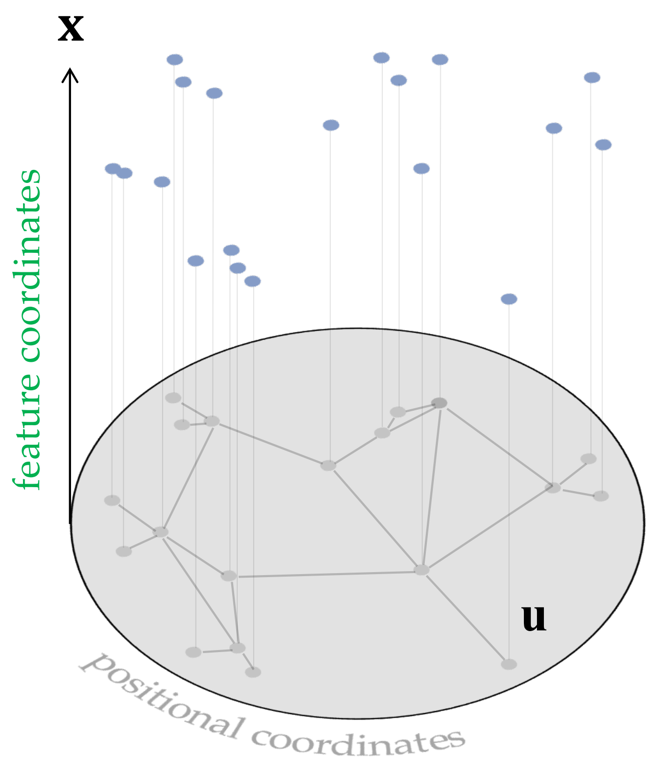

Chamberlain, Rowbottom, et B. 2021

# *Graph Beltrami flow*

- Graph with positional and feature node coordinates $\mathbf{z}_i = (\mathbf{u}_i, \mathbf{x}_i)$
- **Graph Beltrami flow**

$$\dot{\mathbf{z}}_i(t) = \sum_{j \in \mathcal{N}_i} a\Big(\mathbf{z}_i(t), \mathbf{z}_j(t)\Big)\Big(\mathbf{z}_j(t) - \mathbf{z}_i(t)\Big)$$

  - Evolution of $\mathbf{x}$ = feature diffusion
  - Evolution of $\mathbf{u}$ = graph rewiring




Chamberlain, Rowbottom, et B. 2021

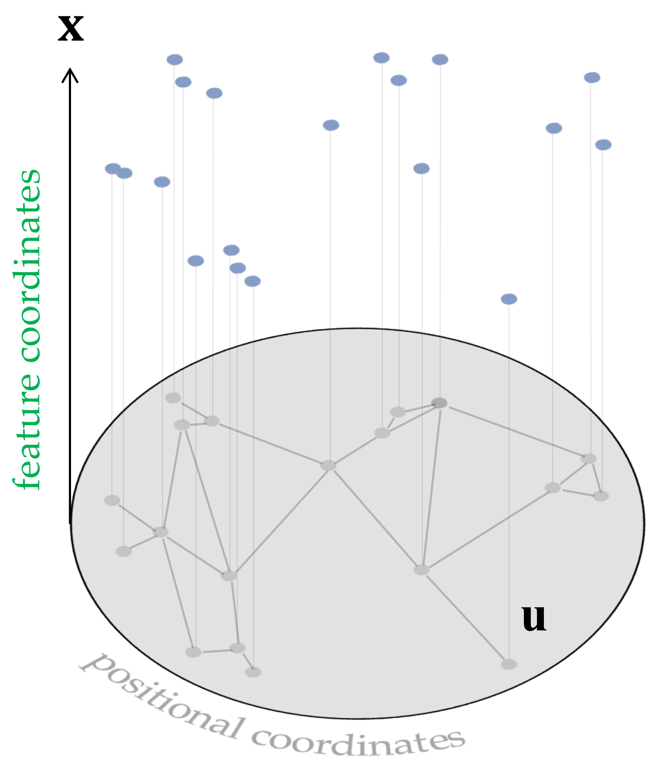
# *Graph Beltrami flow*

- Graph with positional and feature node coordinates $\mathbf{z}_i = (\mathbf{u}_i, \mathbf{x}_i)$
- **Graph Beltrami flow**

$$\dot{\mathbf{z}}_i(t) = \sum_{j \in \mathcal{N}'_i} a\left(\mathbf{z}_i(t), \mathbf{z}_j(t)\right)\left(\mathbf{z}_j(t) - \mathbf{z}_i(t)\right)$$

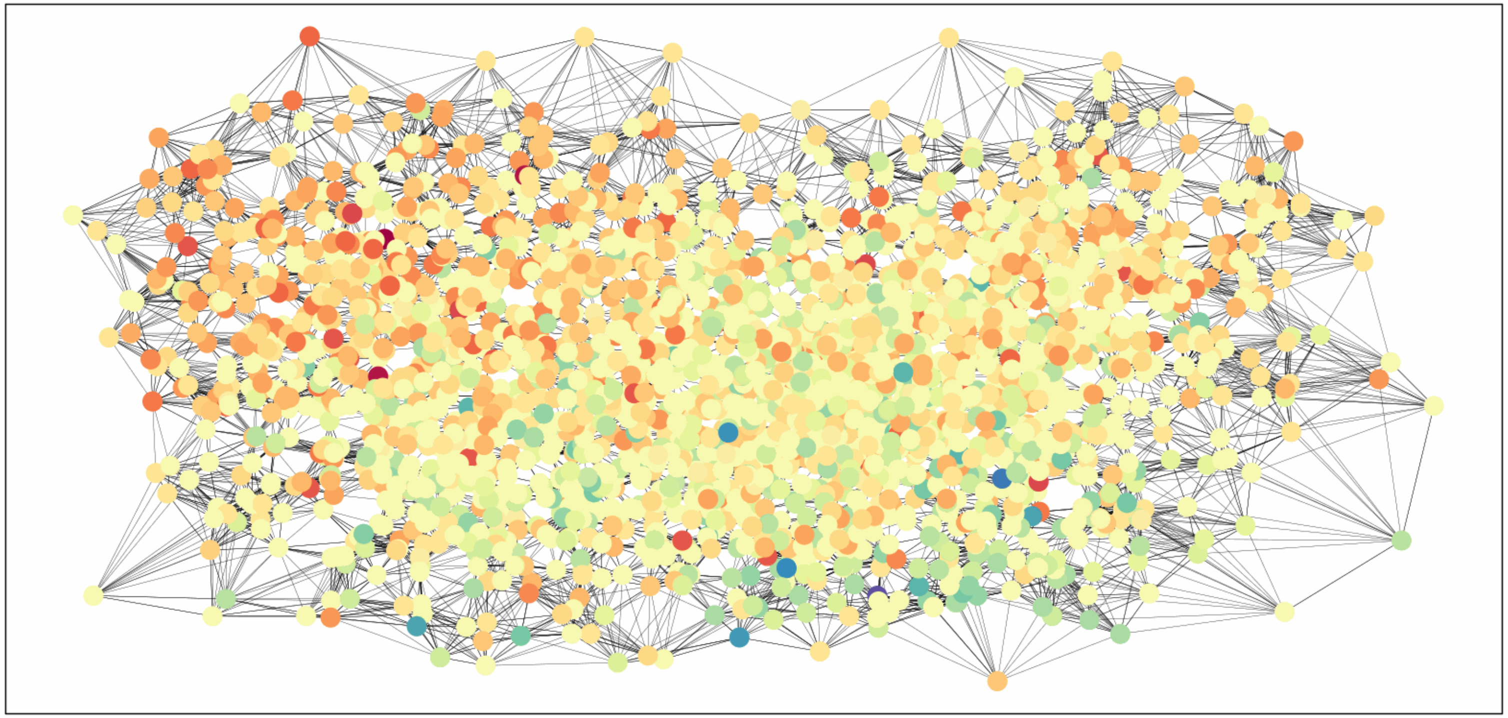
rewired graph

  - Evolution of $\mathbf{x}$ = feature diffusion
  - Evolution of $\mathbf{u}$ = graph rewiring



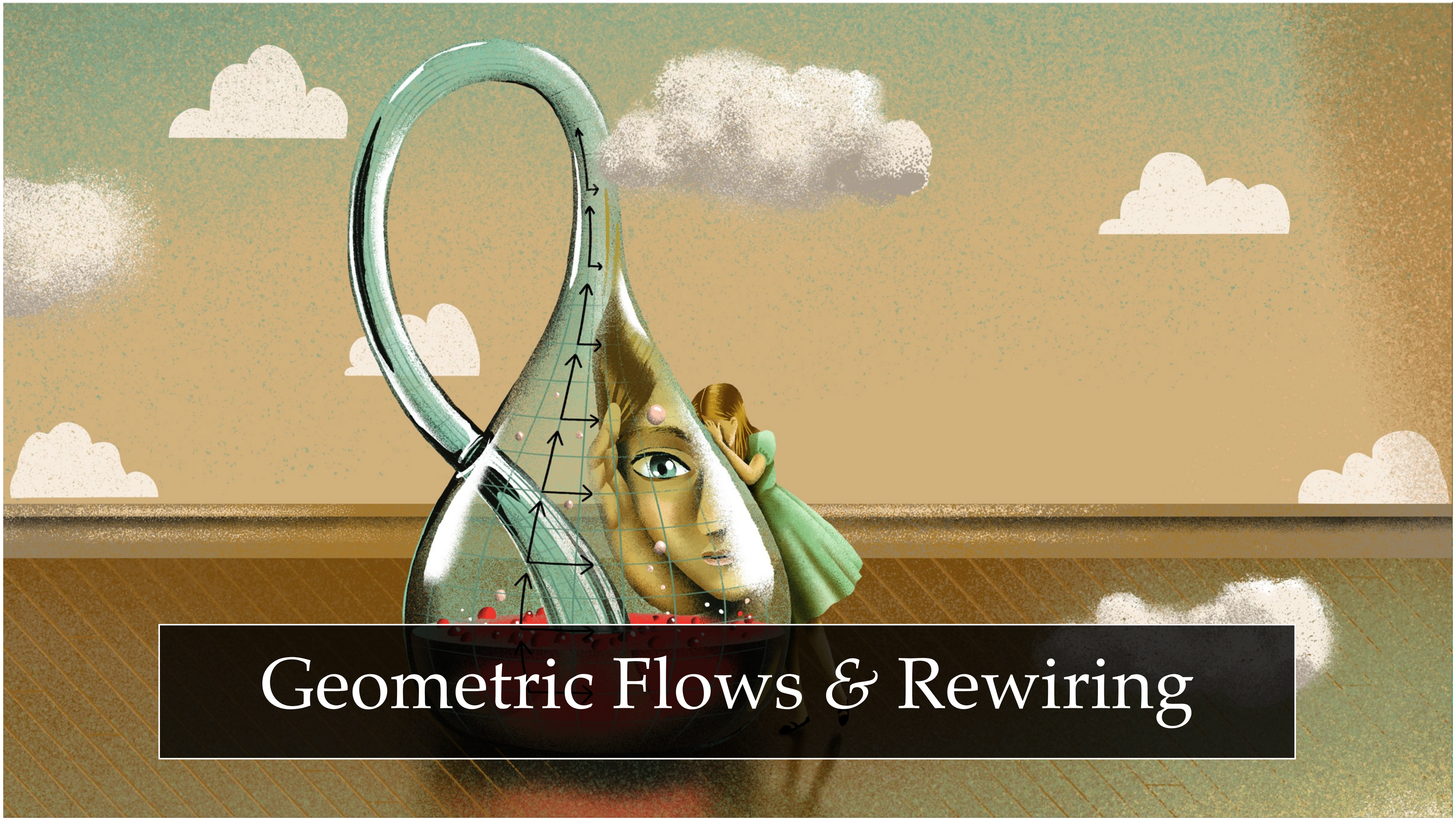

Chamberlain, Rowbottom, et B. 2021

# *Graph Beltrami flow*



Evolution of positional / feature components + rewiring of the Cora graph

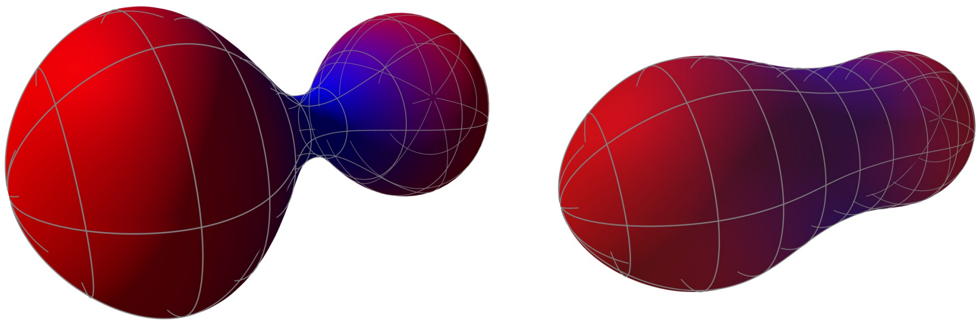
Chamberlain, Rowbottom, et B. 2021


Geometric Flows & Rewiring

# *Ricci flow*

- **Ricci flow:** "diffusion of the Riemannian metric"

$$\frac{\partial g_{ij}}{\partial t} = R_{ij}$$



Evolution of a manifold under Ricci flow

**G. Ricci-Curbastro**

**R. Hamilton**

Ricci 1903; Hamilton 1988;


Ricci 1903; Hamilton 1988; Perelman 2003

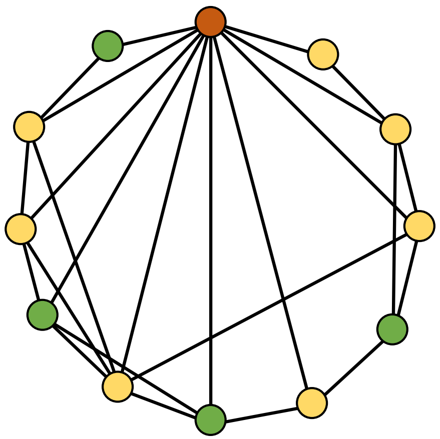
**G. Perelman**

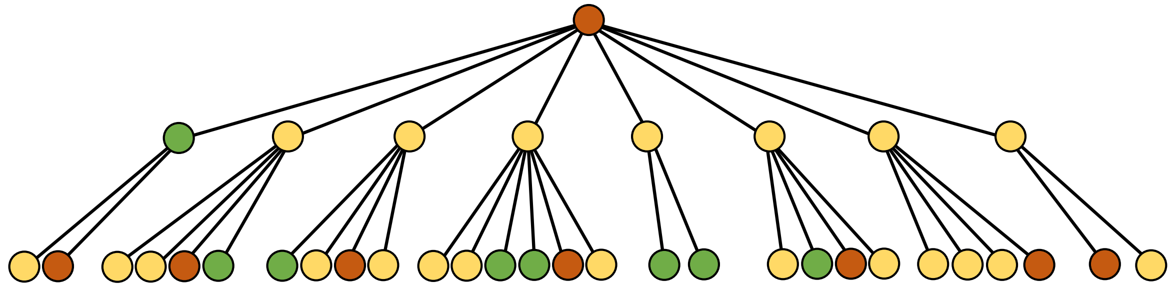
**G. Ricci-Curbastro**

**R. Hamilton**

**“Failure of Message Passing to propagate information on the graph”**

*Over-squashing & Bottlenecks*



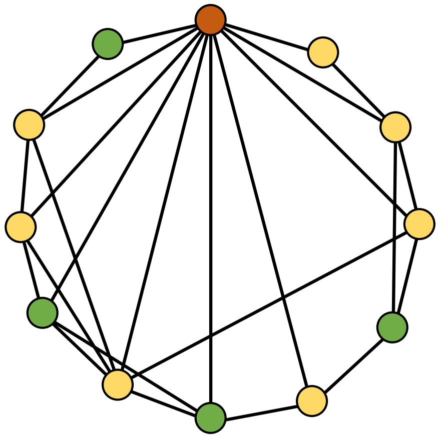


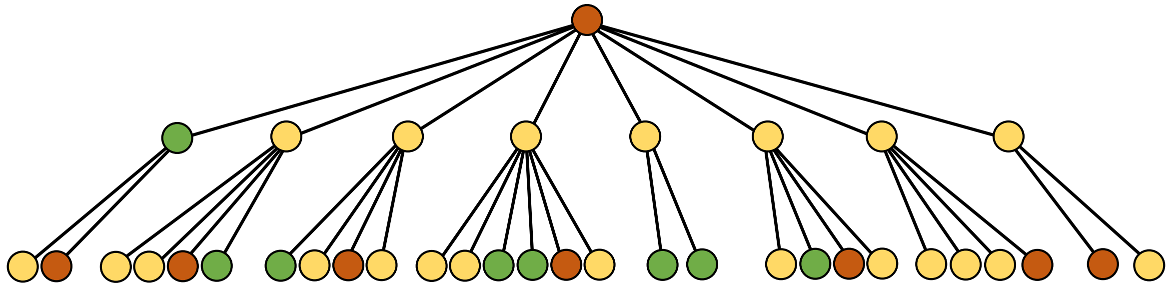
In some graphs metric ball volume grows exponentially with ball radius

Over-squashing = Fast volume growth
+ Long-range interactions

Alon, Yahav 2020

# *Over-squashing & Bottlenecks*





In some graphs metric ball volume grows exponentially with ball radius

graph topology

Over-squashing = Fast volume growth

\+ Long-range interactions

task

Alon, Yahav 2020

# *Over-squashing*

- Consider an MPNN of the form

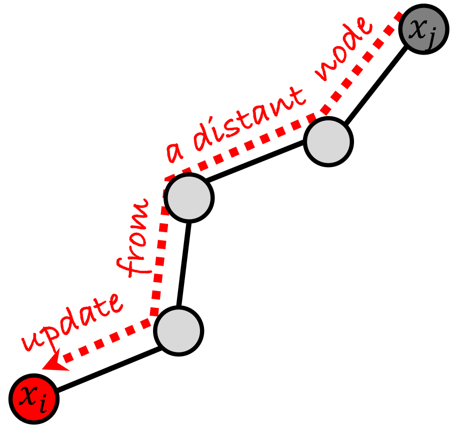
$$\mathbf{x}_i^{(k+1)} = \sigma\left(\mathbf{W}_1\mathbf{x}_i^{(k)} + \sum_j a_{ij}\,\mathbf{W}_2\mathbf{x}_j^{(k)}\right)$$

  - $L$ = *depth* (number of layers)
  - $p$ =*width* (hidden dimension)
  - Nonlinearity $\sigma$ is $c_\sigma$-*Lipschitz-continuous*
  - $w$ = maximum element of weight matrices $\mathbf{W}_1$, $\mathbf{W}_2$

**Theorem (Sensitivity bound):** For any $i, j$

$$\left\|\frac{\partial \mathbf{x}_i^{(L)}}{\partial \mathbf{x}_j^{(0)}}\right\|_1 \leq (c_\sigma w p)^L (\mathbf{I} + \mathbf{A})_{ij}^L$$

Topping, Di Giovanni et B 2021; Di Giovanni et B 2023




**Over-squashing:** small Jacobian $\left\|\partial \mathbf{x}_i^{(L)} / \partial \mathbf{x}_j^{(0)}\right\|$ indicates poor information propagation from input node

# *Over-squashing*

- Consider an MPNN of the form

$$\mathbf{x}_i^{(k+1)} = \sigma\left(\mathbf{W}_1 \mathbf{x}_i^{(k)} + \sum_j a_{ij}\, \mathbf{W}_2 \mathbf{x}_j^{(k)}\right)$$

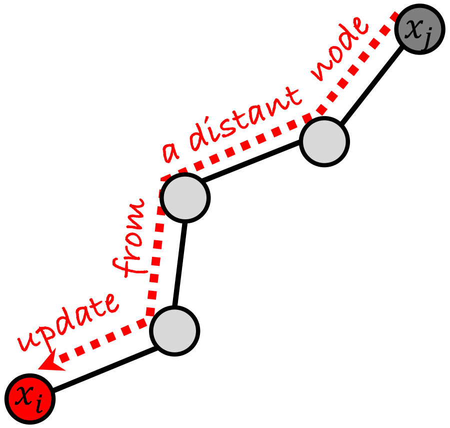
  - $L$ = *depth* (number of layers)
  - $p$ =*width* (hidden dimension)
  - Nonlinearity $\sigma$ is $c_\sigma$-*Lipschitz-continuous*
  - $w$ = maximum element of weight matrices $\mathbf{W}_1$, $\mathbf{W}_2$

**Theorem (Sensitivity bound):** For any $i, j$

$$\left\| \frac{\partial \mathbf{x}_i^{(L)}}{\partial \mathbf{x}_j^{(0)}} \right\|_1 \leq (c_\sigma w p)^L (\mathbf{I} + \mathbf{A})_{ij}^L$$

model topology

Topping, Di Giovanni et B 2021; Di Giovanni et B 2023

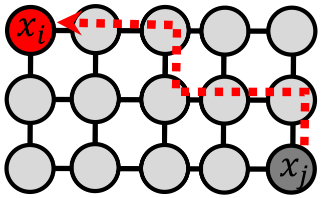


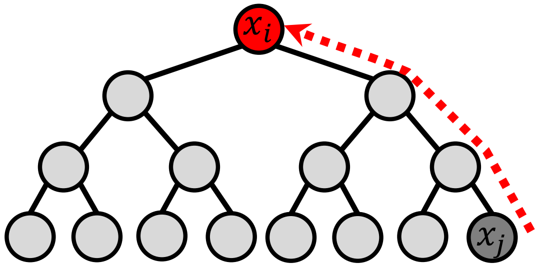

**Over-squashing:** small Jacobian $\left\|\partial \mathbf{x}_i^{(L)} / \partial \mathbf{x}_j^{(0)}\right\|$ indicates poor information propagation from input node

# *Preventing over-squashing*

$$\left\| \frac{\partial \mathbf{x}_i^{(L)}}{\partial \mathbf{x}_j^{(0)}} \right\|_1 \leq (c_\sigma w p)^L (\mathbf{I} + \mathbf{A})^L_{ij}$$

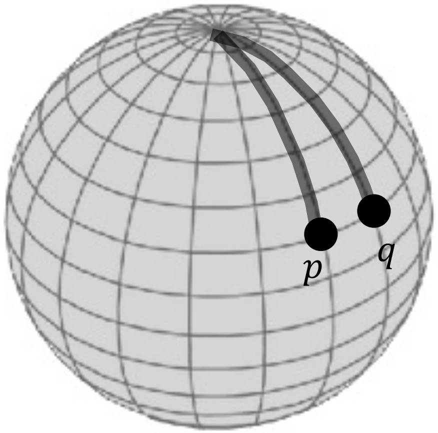
model topology



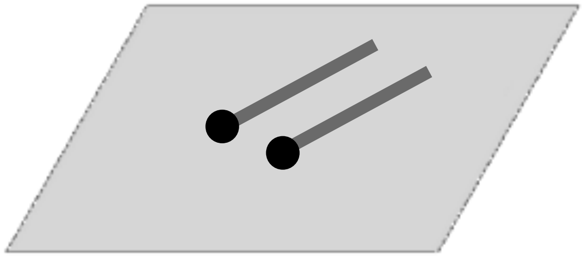




Di Giovanni et B 2023

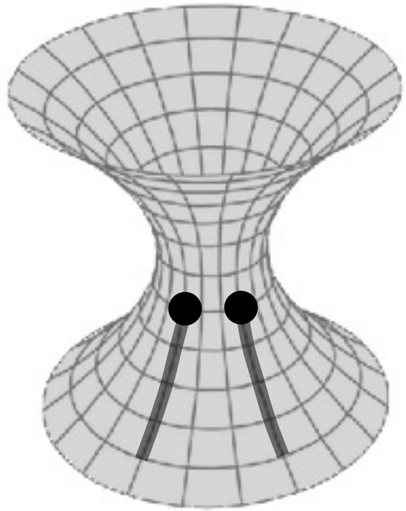
# *Ricci Curvature on Manifolds*




Spherical (>0)



Euclidean (=0)

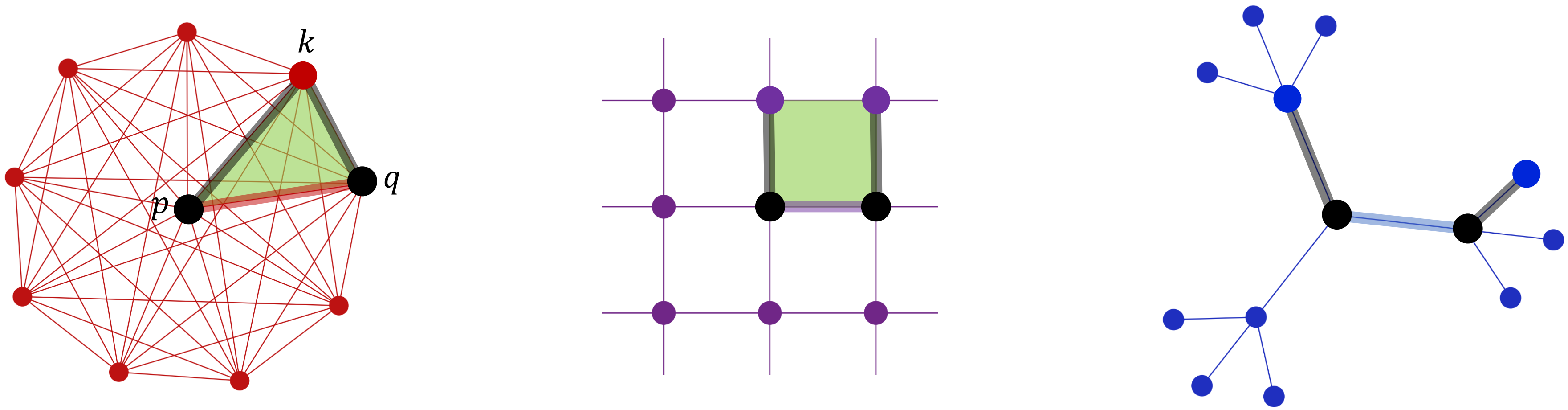


Hyperbolic (<0)

**"geodesic dispersion"**

Ricci 1903

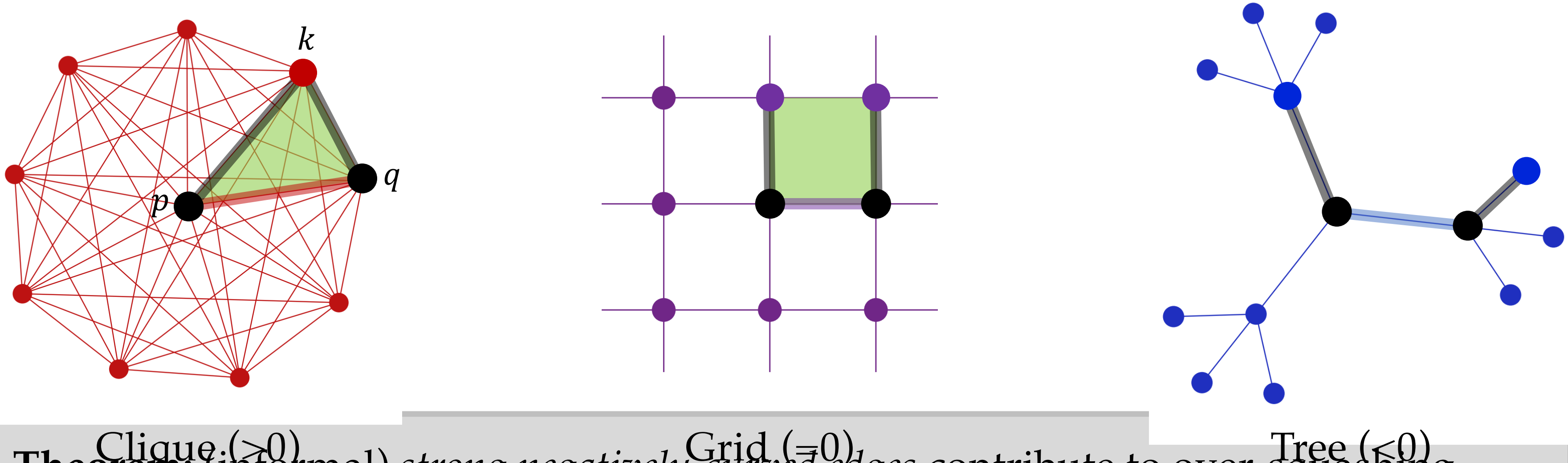
*Discrete Ricci Curvature on Graphs*




Clique (>0) Grid (=0) Tree (<0)

Topping, Di Giovanni et B 2021

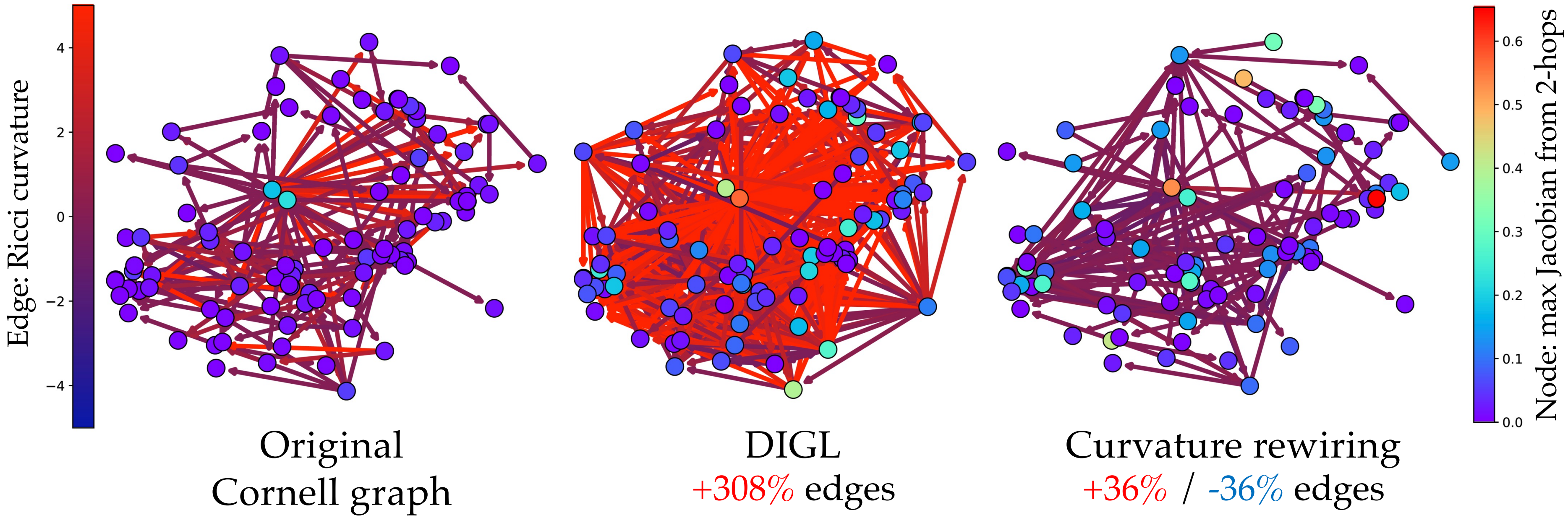
*What contributes to over-squashing?*




**Theorem:** (informal) *strong negatively-curved edges* contribute to over-squashing.

Topping, Di Giovanni et B 2021

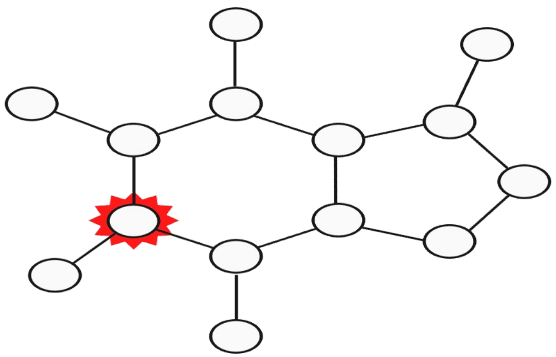
# *Curvature- vs Diffusion-based Rewiring*




**No relation to the task!**

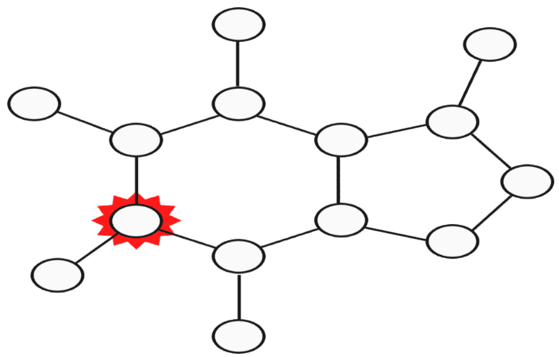
Topping, di Giovanni, et B. 2021; Klicpera et al. 2019 (DIGL)

*Why it is important to consider the task?*



**Van der Waals interactions**

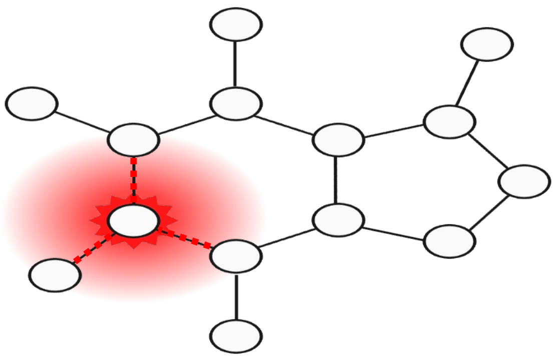
$\propto r^{-12}$



**Coulomb interactions**

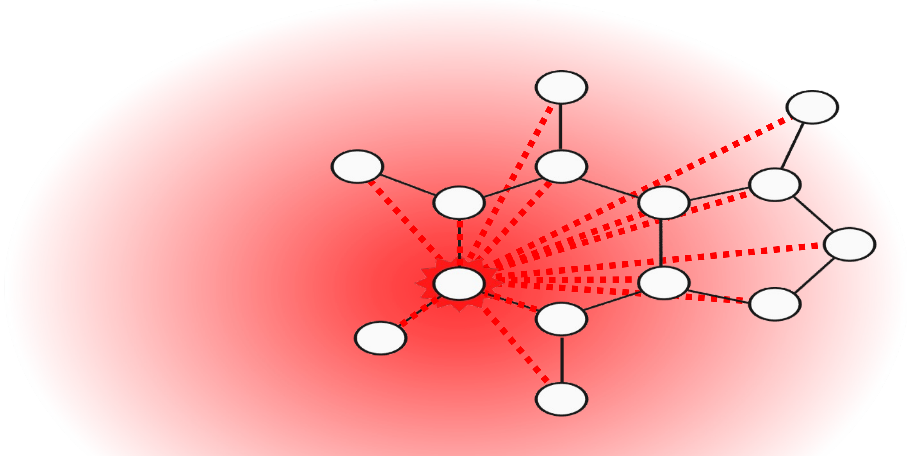
$\propto r^{-1}$

*Why it is important to consider the task?*



**Van der Waals interactions**

$\propto r^{-12}$



**Coulomb interactions**

$\propto r^{-1}$

**Same graph+features, different task**

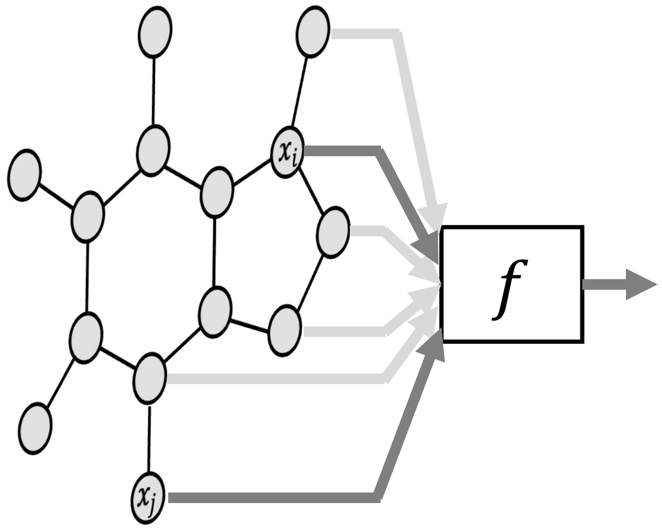
**Whether the graph is good depends on the task!**


Long-range interactions & Expressivity

# *Long-range interactions in graph tasks*

- *Task* = a function $f(\mathbf{X})$ on the node features of a graph $G$
- The interaction between features in nodes $i$ and $j$ required for the task is given by

$$\textbf{Mixing of } f: \quad \mathrm{mix}_f(i,j) = \max_{\mathbf{X}} \max_{1 \le \alpha,\beta \le d} \left| \frac{\partial^2 f(\mathbf{X})}{\partial x_i^\alpha \partial x_j^\beta} \right|$$

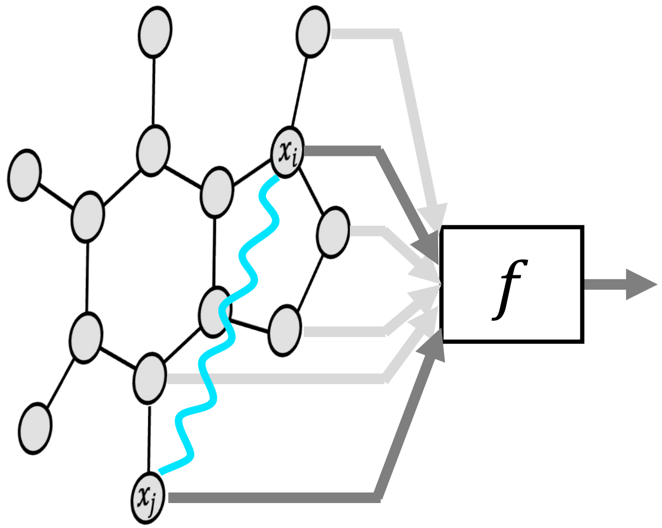



- $f(\mathbf{X}) = \phi(\mathbf{x}_i) + \phi(\mathbf{x}_j)$ is fully separable, thus $\mathrm{mix}_f(i,j) = 0$

Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

# *Long-range interactions in graph tasks*

- *Task* = a function $f(\mathbf{X})$ on the node features of a graph $G$
- The interaction between features in nodes $i$ and $j$ required for the task is given by




**Mixing** of $f$: $\quad \mathrm{mix}_f(i,j) = \max_{\mathbf{X}} \max_{1 \le \alpha,\beta \le d} \left| \frac{\partial^2 f(\mathbf{X})}{\partial x_i^\alpha \partial x_j^\beta} \right|$

- $f(\mathbf{X}) = \phi(\mathbf{x}_i) + \phi(\mathbf{x}_j)$ is fully separable, thus $\mathrm{mix}_f(i,j) = 0$
- $f(\mathbf{X}) = \phi(\langle \mathbf{x}_i, \mathbf{x}_j \rangle)$ mixing depends on how non-linear $\phi$ is

Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

*Capacity bounds*

$$\mathrm{mix}_f(i,j) \leq \sum_{k=1}^{L-1} (c_\sigma w)^{2L-k-1} \left( w \left( \mathbf{S}^{L-k} \right)^{\mathrm{T}} \mathrm{diag}\left(\mathbf{1}^{\mathrm{T}} \mathbf{S}^k\right) \mathbf{S}^{L-k} + C\mathbf{Q}_k \right)_{ij}$$

task model topology

**What is the capacity of MPNN required for a given task?**

Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

*Capacity bounds*

$$\mathrm{mix}_f(i,j) \leq \sum_{k=1}^{L-1} (c_\sigma \mathsf{w})^{2L-k-1} \left( \mathsf{w}\left(\mathbf{S}^{L-k}\right)^{\mathrm{T}} \mathrm{diag}\left(\mathbf{1}^{\mathrm{T}}\mathbf{S}^{k}\right)\mathbf{S}^{L-k} + c\mathbf{Q}_k \right)_{ij}$$

task — model — topology

**What is the capacity of MPNN required for a given task?**

model + topology — mixing

Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

# *Capacity bounds*

$$\text{mix}_f(i,j) \leq \sum_{k=1}^{L-1} (c_\sigma w)^{2L-k-1} \left( w \left(\mathbf{S}^{L-k}\right)^{\mathrm{T}} \text{diag}\left(\mathbf{1}^{\mathrm{T}} \mathbf{S}^k\right) \mathbf{S}^{L-k} + c\,\mathbf{Q}_k \right)_{ij}$$

**Bound on weights $w$**

$$w \geq \frac{d_{\min}}{c_2} \left( \frac{\text{mix}_f(i,j)}{q} \right)^{1/d(i,j)}$$

- $d_{\min}$ =min node degree
- Fixed depth $L = \lceil d(i,j)/2 \rceil$
- $q$ =number of paths of length $d(i,j)$ between $i$ and $j$

**Bound on depth $L$**

$$L \geq \frac{\tau(i,j)}{4c_2} + \frac{|E|}{\sqrt{d_i d_j}} \left( \alpha \text{mix}_f(i,j) - \beta \right)$$

- $d_i$ =degree of node $i$
- $\alpha, \beta$ =model-related constants
- $|E|$ =number of edges
- Bounded weights

Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

## *Capacity bounds*

$$\text{mix}_f(i,j) \leq \sum_{k=1}^{L-1} (c_\sigma w)^{2L-k-1} \left( w \left(\mathbf{S}^{L-k}\right)^{\mathrm{T}} \text{diag}\left(\mathbf{1}^{\mathrm{T}} \mathbf{S}^{k}\right) \mathbf{S}^{L-k} + C\mathbf{Q}_k \right)_{ij}$$

**Bound on weights** $w$

$$w \geq \frac{d_{\min}}{c_2} \left( \frac{\text{mix}_f(i,j)}{q} \right)^{1/d(i,j)}$$

*"weights need to be large enough to allow mixing"*

**Bound on depth** $L$

$$L \geq \frac{\tau(i,j)}{4c_2} + \frac{|E|}{\sqrt{d_i d_j}} \left( \alpha \text{mix}_f(i,j) - \beta \right)$$

- *Depth must be ~commute time* $\tau(i,j)$
- *Rewiring tries to improve* $\tau$
- $\tau$ *can be as large as* $O(n^3)$*, which implies impossibility statements*

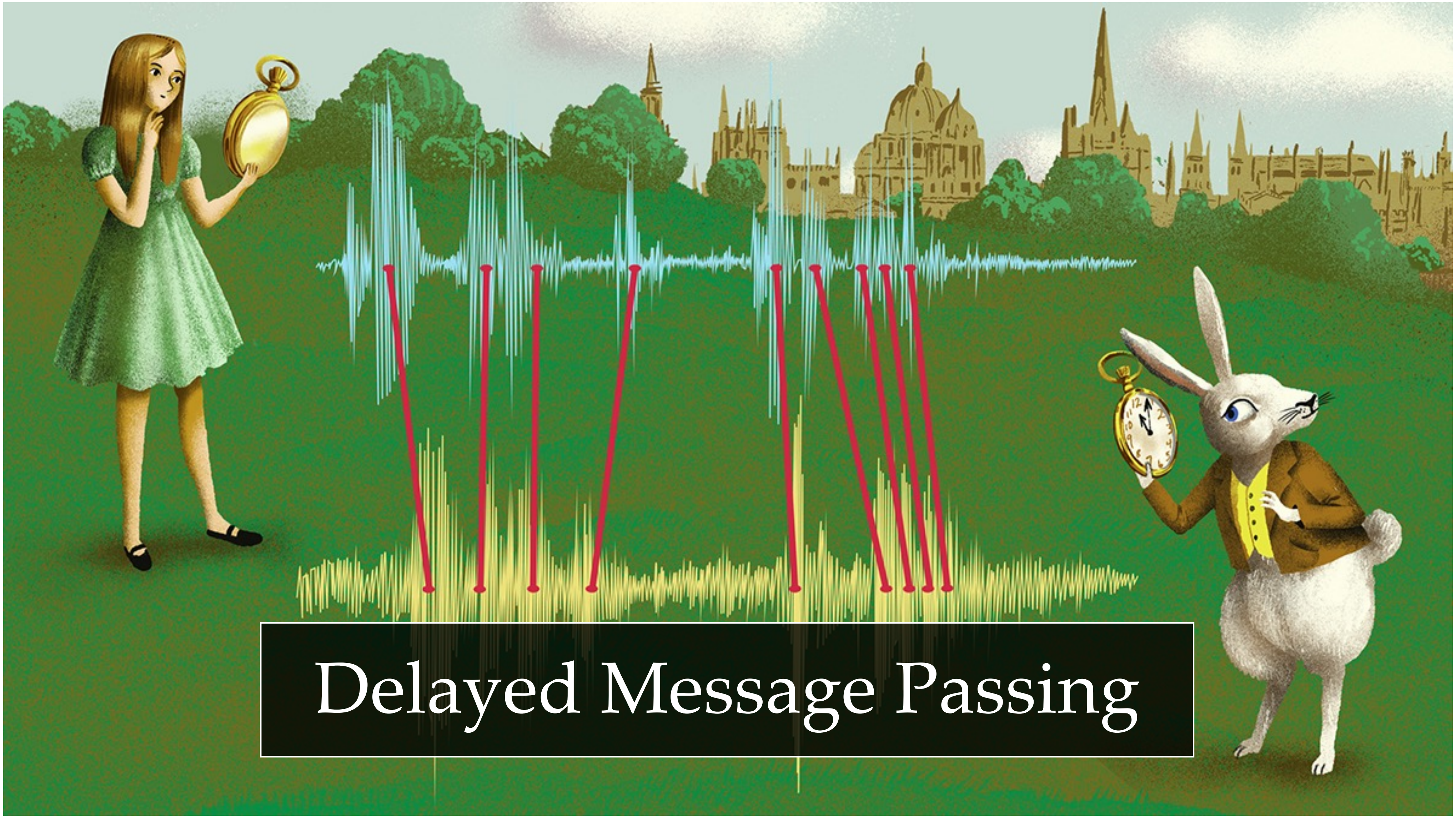
Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023

# *Expressive power beyond Weisfeiler-Lehman*

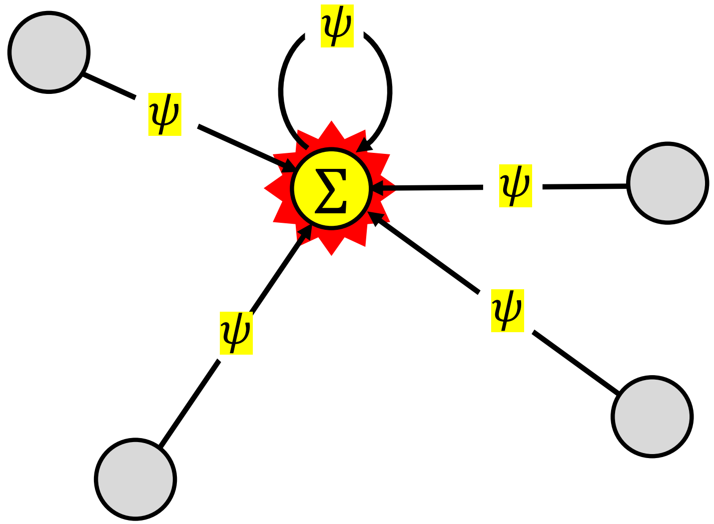
**Expressive power (informal):** MPNN with $L \leq n$ layers *cannot learn* tasks that require high mixing among features at nodes with large commute time.

**A. Lehman** **B. Weisfeiler**

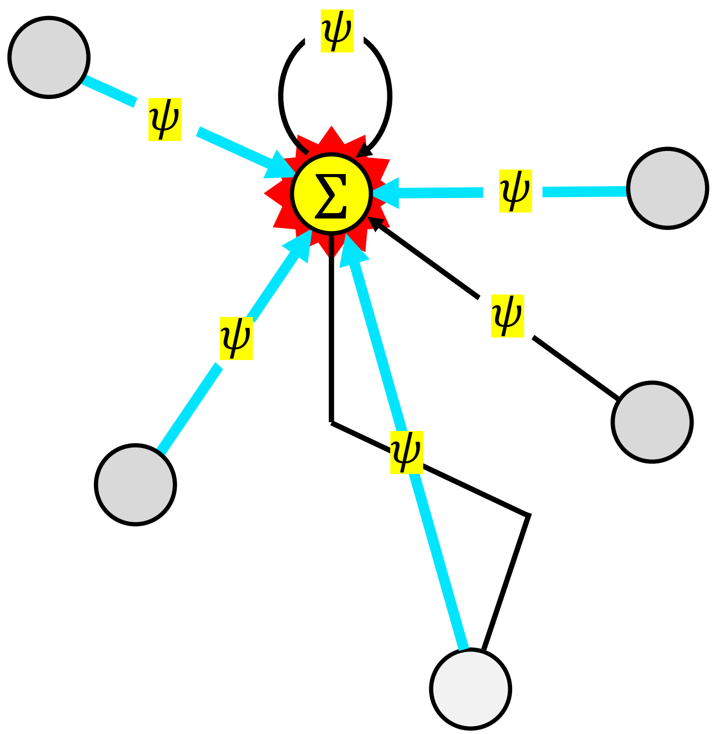
Di Giovanni, Rusch, B, Deac, Lackenby, Mishra, Veličković 2023


Delayed Message Passing

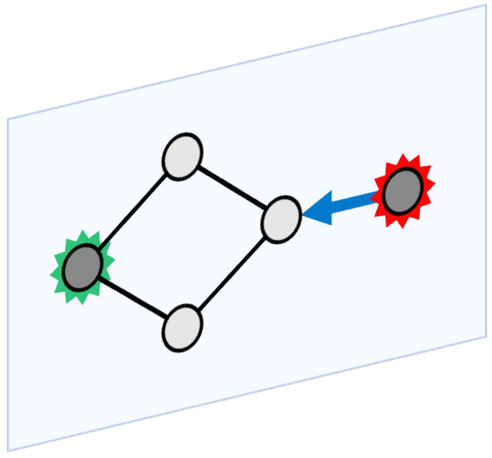
What



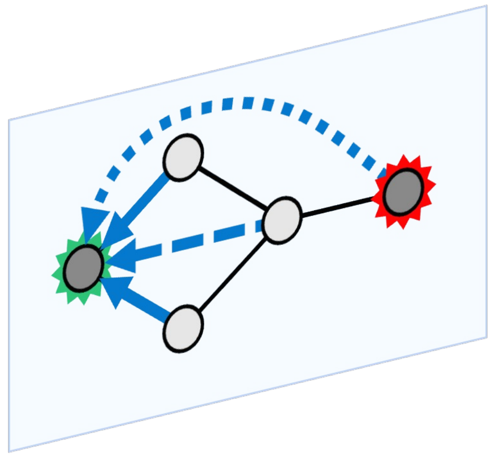
What + Where



What + Where + When

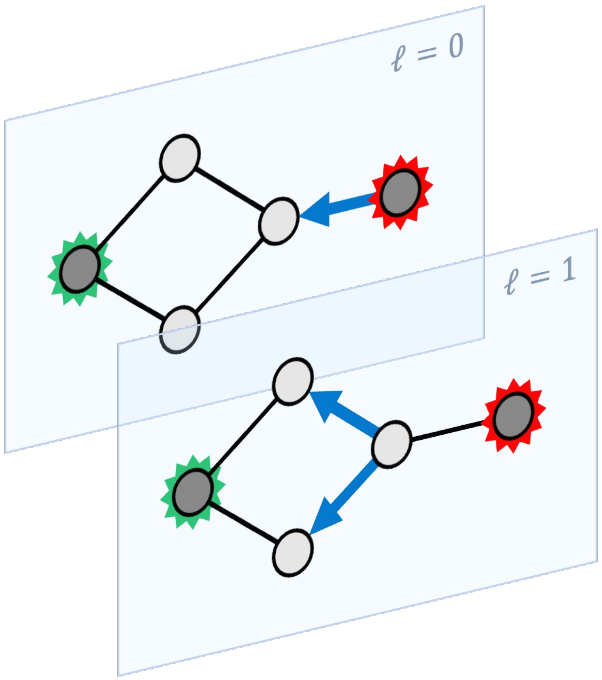


Classical MPNN

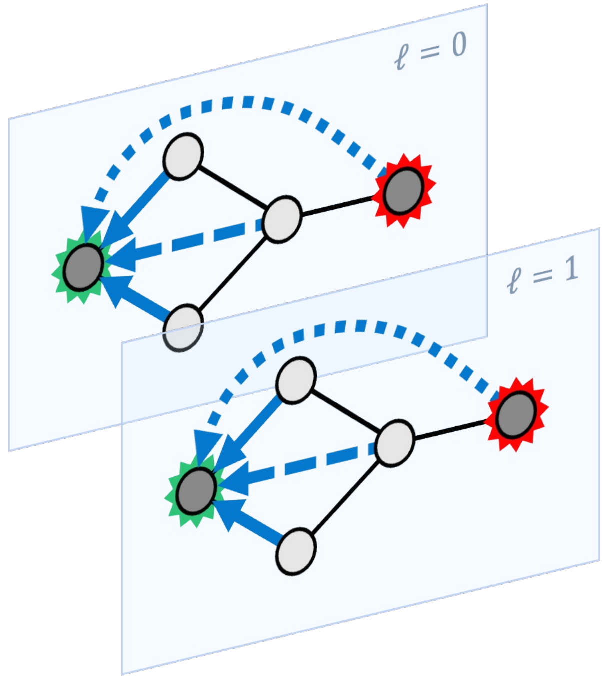


Graph Transformer

Gutteridge, Di Giovanni et B 2023

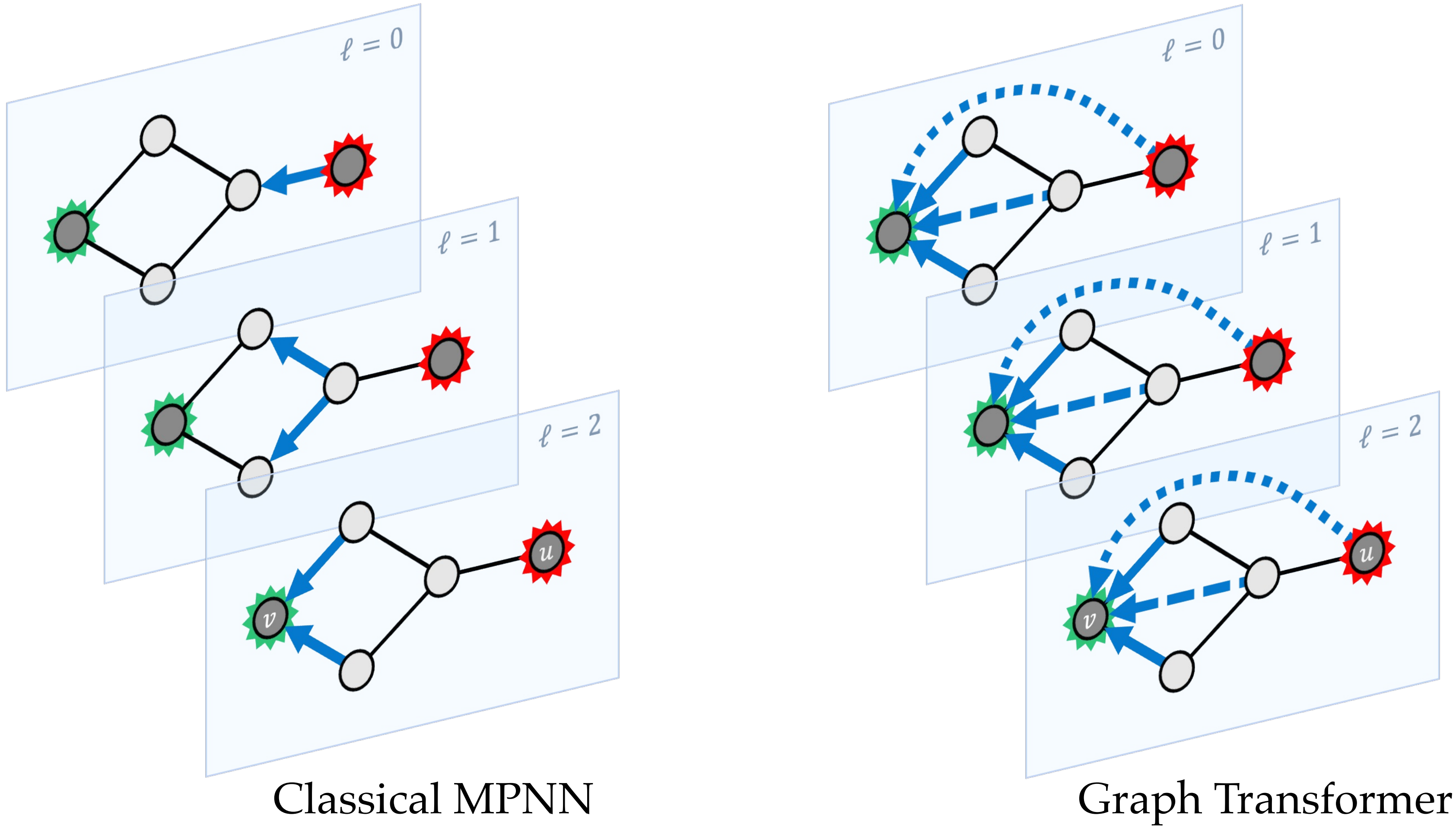



Classical MPNN

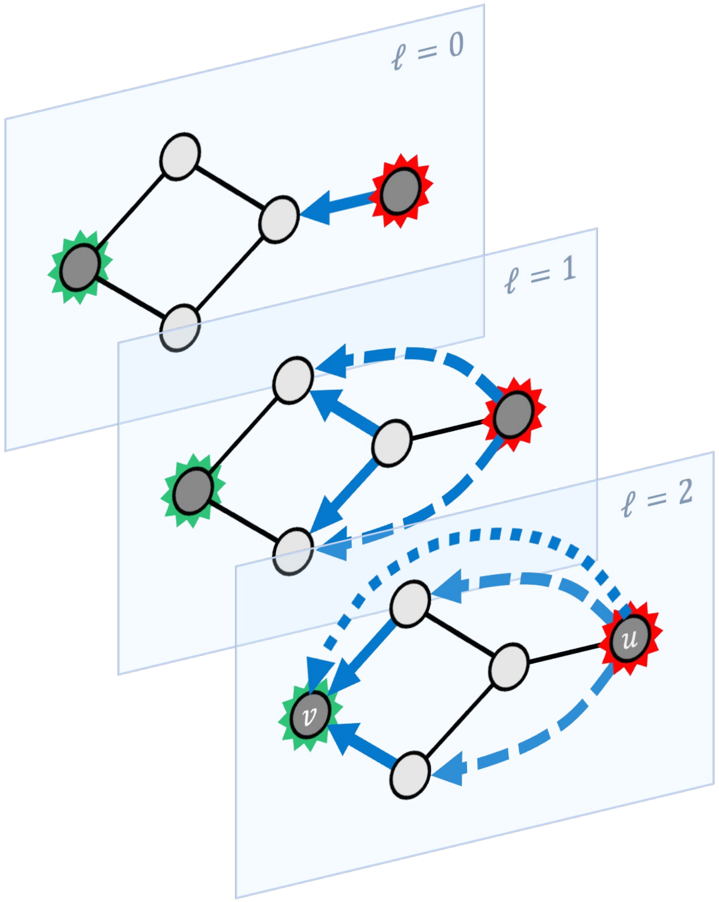



Graph Transformer

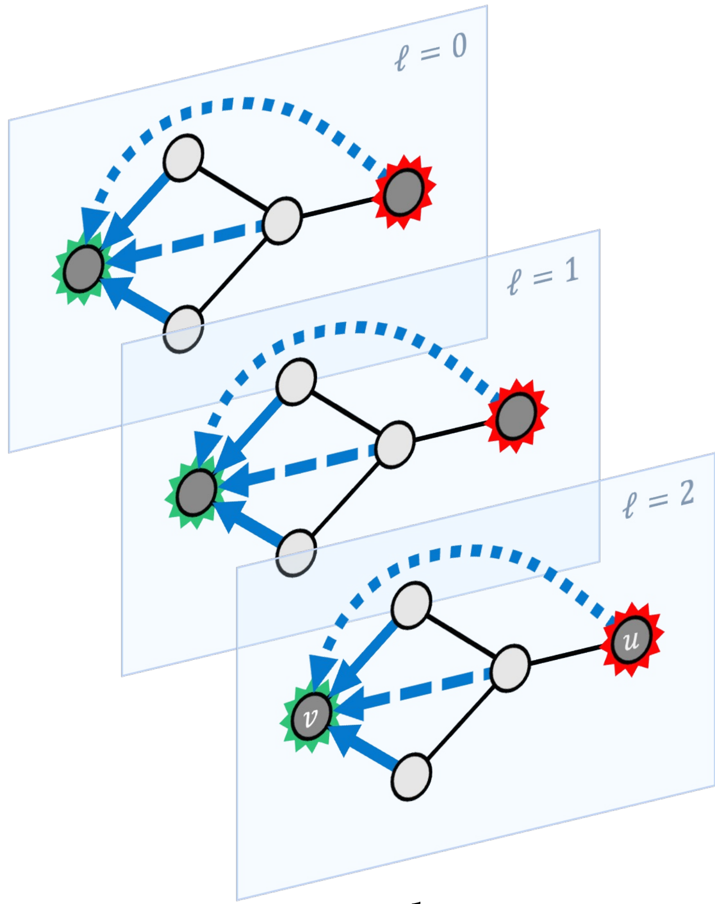
Gutteridge, Di Giovanni et B 2023




Gutteridge, Di Giovanni et B 2023

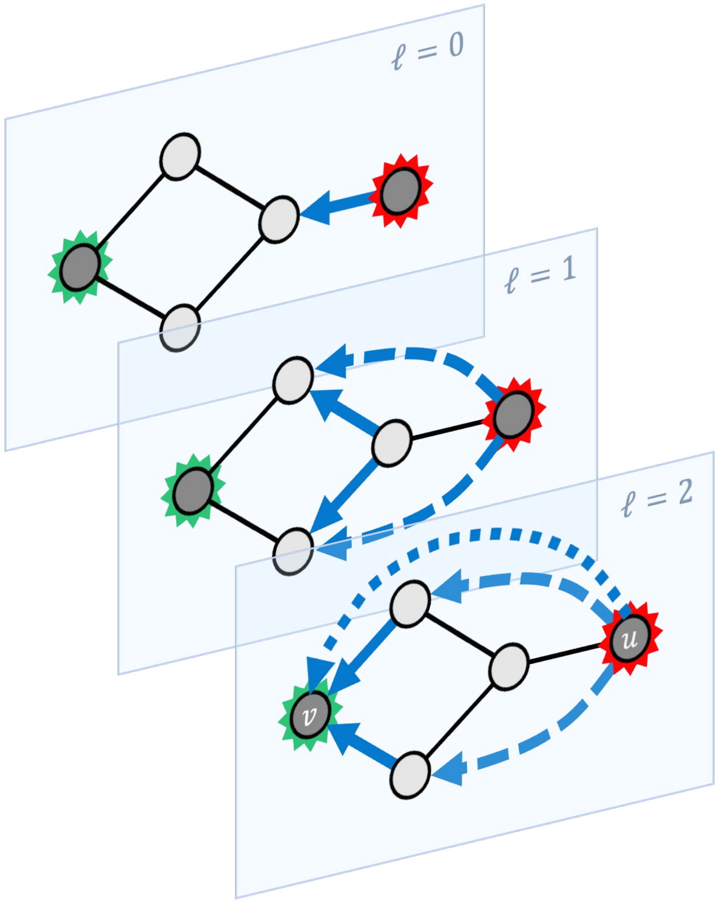



Dynamic Rewiring (DRew)



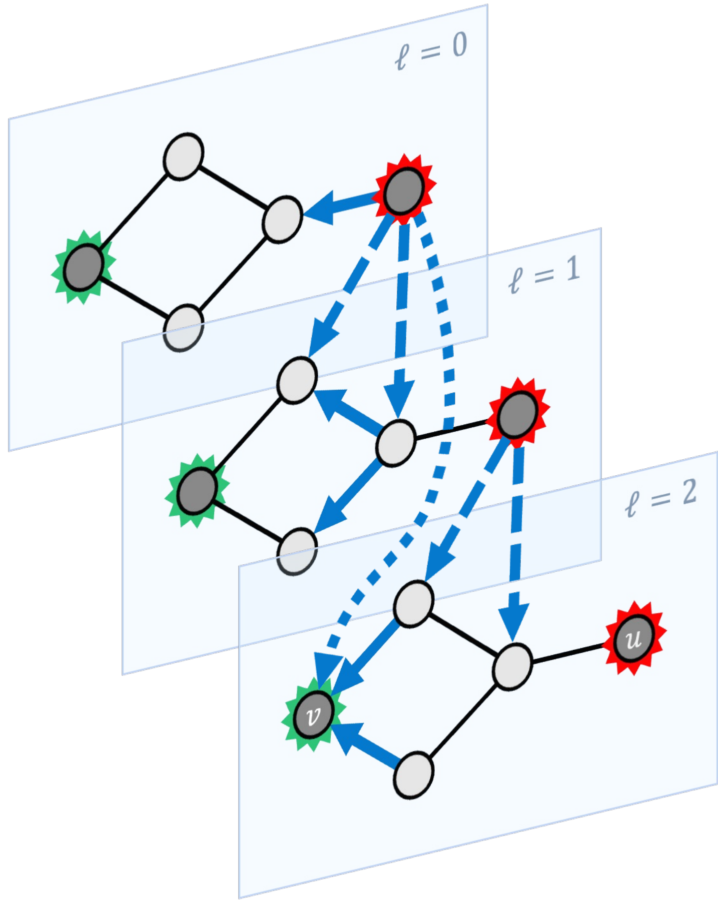

Graph Transformer

Gutteridge, Di Giovanni et B 2023



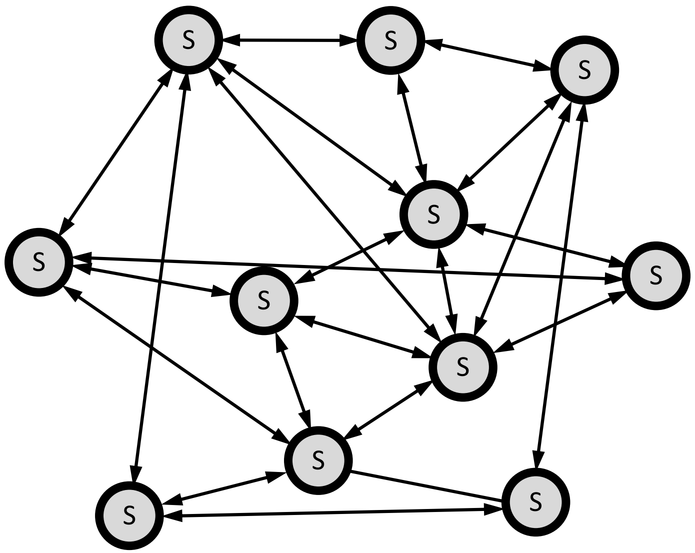

Dynamic Rewiring
(DRew)

Gutteridge, Di Giovanni et B 2023

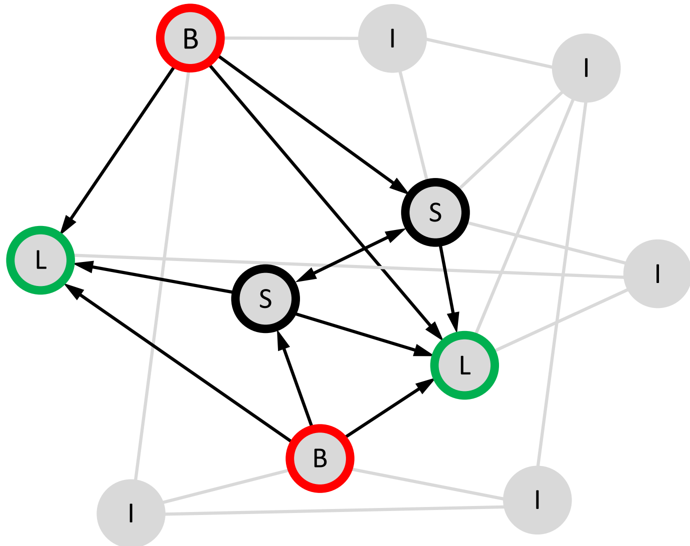



Dynamic Rewiring + delay
(**ν**DRew)

# *Cooperative Message Passing*



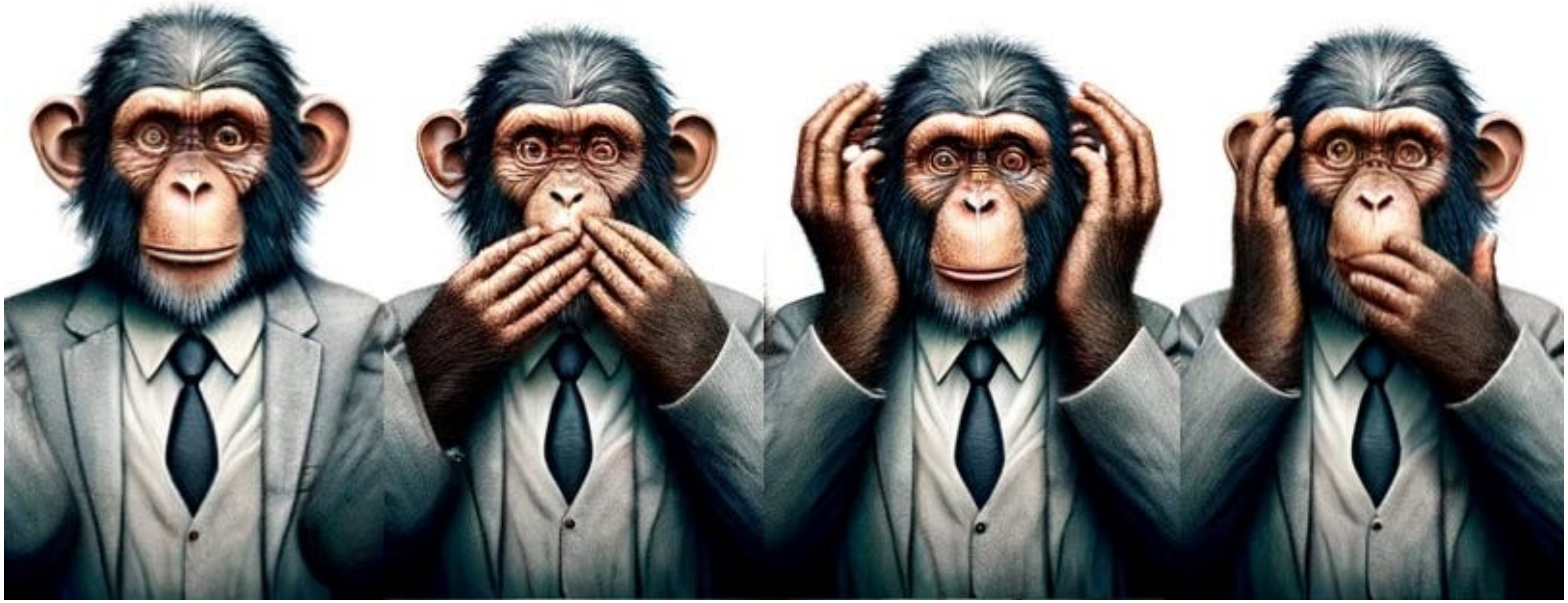

**Standard Message Passing**
each node Broadcasts & Listens




**Cooperative Message Passing**
each node individually decides

Finkelshtein, Huang, B, Ceylan 2023

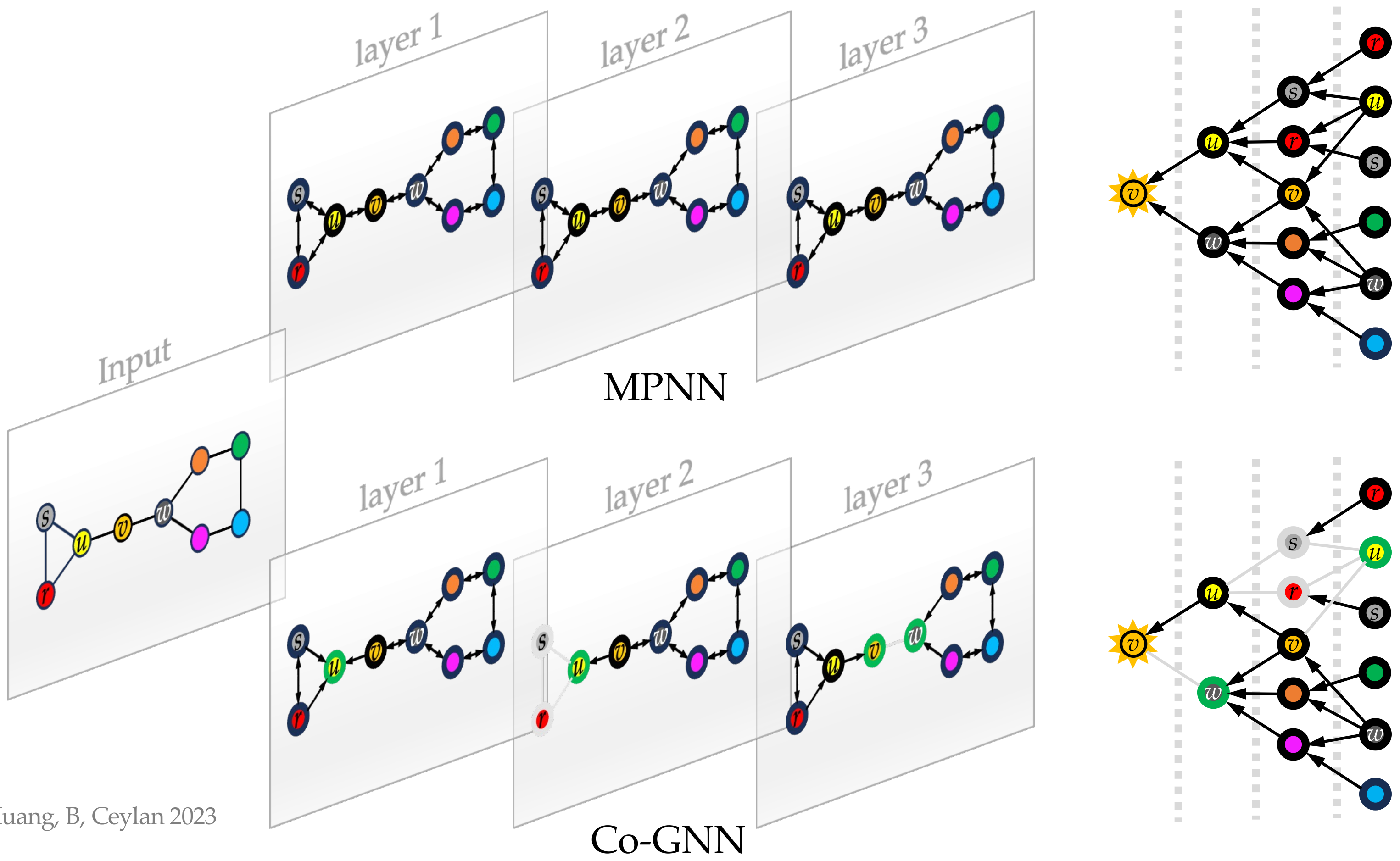
*Cooperative Message Passing*



**Broadcast & Listen** **Listen** **Broadcast** **Isolate**

Finkelshtein, Huang, B, Ceylan 2023; Illustration: DALL-E 3 (after a lot of effort)

# *Cooperative Message Passing*




Finkelshtein, Huang, B, Ceylan 2023

## *What do we gain from physics-inspired GNNs?*

- New perspectives on old problems (e.g. oversmoothing, bottlenecks, etc.)
- Explains old architectures & gives rise to new ones
- Principled architectural choices (residual connection, shared symmetric weights)
- Theoretical guarantees (e.g. stability, convergence, expressive power, etc.)
- Deep links to other fields less known in GNN literature (e.g. differential geometry & algebraic topology)
- In GNNs, the graph is both *input* and *computational device* – not all graphs are good!
- Rewiring tells *what* messages to send *where*
- Dynamic rewiring+delay adds control also *when*


Thank you!